

The Lifecycle of Representations: Constructing, Curating, and Communicating Generative Libraries

Dissertation

der Mathematisch-Naturwissenschaftlichen Fakultät
der Eberhard Karls Universität Tübingen
zur Erlangung des Grades eines
Doktors der Naturwissenschaften
(Dr. rer. nat.)

vorgelegt von
M.Sc. Hanqi Zhou
aus Jiangsu/China

Tübingen
2026

Gedruckt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der
Eberhard Karls Universität Tübingen.

Tag der mündlichen Qualifikation:

16.07.2026

Dekan:

Prof. Dr. Thilo Stehle

1. Berichterstatter:

Prof. Dr. Charley M. Wu

2. Berichterstatter:

Prof. Dr. Michael Franke

3. Berichterstatter:

Prof. Dr. Frank Jäkel

Supervisor: Prof. Dr. Charley M. Wu

Thesis Advisory Committee: Prof. Dr. Celestine Mendler-Dünner
Prof. Dr. Robert Bamler
Prof. Dr. Peter Dayan

To Haven Feng, who believed in me before I fully believed in myself.
I am endlessly fortunate to have found in you my closest friend, a companion in curiosity,
and someone who shares my wonder for the questions that guide us.

Acknowledgments

I am deeply grateful to my advisor, Charley Wu, for his support and guidance throughout my PhD. I thank him for his openness and patience, and for giving me the trust and freedom to explore and find my own path as a researcher. I also thank him for bringing together such a wonderful lab—a place where I could learn, think, struggle, and grow alongside great people.

I am also deeply grateful to Robert Bamler and Álvaro Tejero-Cantero for supervising me during my first year. Their technical depth taught me a great deal, and seeing how they did research shaped the way I think about scientific rigor.

I would like to thank Seth Axen, for being a fellow coffee enthusiast, a thoughtful mentor, and a good friend. Being extremely dedicated and knowledgeable while humble, he showed me something close to an ideal of what a researcher can be. I am also grateful for our many conversations about shows and movies of varying artistic merit, which made many days lighter.

I would like to thank Mani Hamidi for being such a kind and generous friend. I am grateful for our many scientific and personal conversations, including our early, curious, and often confused attempts to understand causal inference together.

I would like to thank David Nagy for being both a friend and an intellectual sounding board. I am grateful that I could turn to him for anything, from concrete technical advice to broader ways of thinking, and for our many enjoyable conversations about all kinds of “abstraction” in life.

I would like to thank Alex Witt and Turan Orujlu for sharing the office with me, and for the after-work drinks and dinners that gave our PhD days some much-needed warmth.

I am also grateful to Peter Dayan and Kou Murayama for always being approachable and insightful, and for guiding me along the way with such generosity. I thank Tobias Gerstenberg for hosting me for a wonderful semester, interleaved with game nights, drinks, and fun experiment designs.

I am grateful to all my friends and collaborators throughout this journey, including Alex, Britt, Bonan, Caroline, Chenfei, Eunice, Fei, Freddy, Maximilian, Ruiqi, Susu, Tim, Valkyrie, Yirong, Zeju, and many others who have supported me over the past four years. Their support came in ideas and feedback, conversations, shared meals, small kindnesses, and simply being there. All of it mattered more than I can fully say.

I am grateful to Sebastian Schwenk, Shyamala Subramanian, and Elena Sizana for always being there, for their care and help with all administrative matters. Their support made life beyond research much easier.

I am especially grateful to Ulrike von Luxburg and Heike Größl for their support during one of the most difficult periods of my PhD. Their encouragement and understanding helped me continue when it was not easy.

I thank my parents for their unconditional love. They have always stood by me and supported my choices in every way they could. Their love has been the quiet ground beneath this journey.

Finally, I would like to thank myself—for making it this far.

Hanqi Zhou
Tübingen, May 1, 2026

Abstract

Cognitive science is good at modeling how minds update beliefs within a representational vocabulary that is already in place. It is much weaker at modeling how that vocabulary itself changes. Symbolic systems hand-design their vocabularies and inherit no process by which experience could have produced them. Connectionist systems learn representations, but the usual analytic lens is convergence to a final code rather than the accumulation of reusable structure that reshapes later learning. Bayesian systems specify a hypothesis space and perform inference within it, leaving open how that space was carved up in the first place and how it should be revised as the agent encounters new tasks. The shared failure is not the absence of representation, but the absence of representational *dynamics*: a process-level account of how vocabulary is built, maintained, and shared, and of how each of these changes what becomes easy to learn next.

This thesis argues that these dynamics become tractable once a particular object is named: a *generative library* $\mathcal{L}$ of reusable, executable, compositional, and inspectable building blocks over which hypotheses are composed. The central claim is that intelligence is the resource-rational lifecycle management of $\mathcal{L}$. The claim has bite because the properties that make something a library impose three demands on any adequate account. First, library entries accumulate. What has already been compressed determines what can be represented cheaply next, so construction must be modeled as path-dependent rather than as convergence to a fixed code. Second, a library is finite, and the value of an entry depends on uses it has not yet seen. Curation therefore cannot be reduced to local utility; it requires a criterion defined over the agent's own representational state and the future computations that state makes possible. Third, library entries are inspectable and shareable. Communication is therefore not merely the transmission of information, but the alignment of representational resources; an explanation succeeds only when it can be compiled into action by a listener whose library may differ from the speaker's.

These demands are not three separable problems. A library that is cheap to build but undisciplined in what it keeps will drift. A library that is well curated but opaque will not survive contact with another agent. A library that is communicable but built without regard to learning history will encode the wrong things. The lifecycle is the constraint that construction, curation, and communication must be optimized together.

The thesis develops this argument through a common methodology: latent inference over generative models of agents, with compression and two-part description length as the cross-stage criterion of representational value. The empirical chapters span controlled human sequence learning, large-scale educational data, simulated agents driven by competing intrinsic motivations, and human studies of explanation under uncertainty. Each chapter is built so that its model and its evidence speak to one stage of the lifecycle using the same yardstick as the others. The contribution is therefore not a model of any single cognitive capacity, but a unified account of representational dynamics in which construction, curation, and communication are stages of the same process and become optimizable only together.

Zusammenfassung

Die Kognitionswissenschaft ist gut darin, zu modellieren, wie der Geist Überzeugungen innerhalb eines bereits bestehenden repräsentationalen Vokabulars aktualisiert. Deutlich schwächer ist sie darin, zu modellieren, wie sich dieses Vokabular selbst verändert. Symbolische Systeme entwerfen ihre Vokabulare von Hand und verfügen über keinen Prozess, durch den Erfahrung sie hätte hervorbringen können. Konnektionistische Systeme lernen Repräsentationen, doch der übliche analytische Blick richtet sich auf die Konvergenz zu einem endgültigen Code, nicht auf die Akkumulation wiederverwendbarer Struktur, die späteres Lernen umformt. Bayesianische Systeme spezifizieren einen Hypothesenraum und führen innerhalb dieses Raums Inferenz durch, lassen jedoch offen, wie dieser Raum ursprünglich gegliedert wurde und wie er überarbeitet werden sollte, wenn der Agent neuen Aufgaben begegnet. Das gemeinsame Defizit ist nicht das Fehlen von Repräsentation, sondern das Fehlen repräsentationaler *Dynamik*: eines prozessualen Berichts darüber, wie Vokabular aufgebaut, gepflegt und geteilt wird, und wie jeder dieser Vorgänge verändert, was als Nächstes leicht zu lernen ist.

Diese Arbeit argumentiert, dass diese Dynamiken handhabbar werden, sobald ein bestimmtes Objekt benannt wird: eine *generative Bibliothek* $\mathcal{L}$ aus wiederverwendbaren, ausführbaren, kompositionalen und inspizierbaren Bausteinen, über denen Hypothesen zusammengesetzt werden. Die zentrale These lautet, dass Intelligenz im ressourcenrationalen Lebenszyklusmanagement von $\mathcal{L}$ besteht. Diese These hat Substanz, weil die Eigenschaften, die etwas zu einer Bibliothek machen, drei Anforderungen an jede angemessene Theorie stellen. Erstens akkumulieren Bibliothekseinträge. Was bereits komprimiert wurde, bestimmt, was als Nächstes günstig repräsentiert werden kann; daher muss Konstruktion als pfadabhängig modelliert werden, nicht als Konvergenz zu einem festen Code. Zweitens ist eine Bibliothek endlich, und der Wert eines Eintrags hängt von Verwendungen ab, die noch nicht eingetreten sind. Kuratierung kann daher nicht auf lokalen Nutzen reduziert werden; sie erfordert ein Kriterium, das über den eigenen repräsentationalen Zustand des Agenten und die zukünftigen Berechnungen definiert ist, die dieser Zustand ermöglicht. Drittens sind Bibliothekseinträge inspizierbar und teilbar. Kommunikation ist daher nicht bloß die Übertragung von Information, sondern die Abstimmung repräsentationaler Ressourcen; eine Erklärung gelingt nur dann, wenn sie von einem Hörer, dessen Bibliothek sich von der des Sprechers unterscheiden kann, in Handlung übersetzt werden kann.

Diese Anforderungen sind keine drei voneinander trennbaren Probleme. Eine Bibliothek, die billig aufzubauen ist, aber undiszipliniert darin, was sie behält, wird driften. Eine Bibliothek, die gut kuratiert, aber opak ist, wird den Kontakt mit einem anderen Agenten nicht überstehen. Eine Bibliothek, die kommunizierbar ist, aber ohne Rücksicht auf die Lerngeschichte aufgebaut wurde, wird das Falsche kodieren. Der Lebenszyklus ist die Bedingung, dass Konstruktion, Kuratierung und Kommunikation gemeinsam optimiert werden müssen.

Die Arbeit entwickelt dieses Argument durch eine gemeinsame Methodologie: latente Inferenz über generative Modelle von Agenten, wobei Kompression und zweigeteilte Beschreibungslänge als phasenübergreifendes Kriterium repräsentationalen Werts dienen. Die empirischen Kapitel umfassen kontrolliertes menschliches Sequenzlernen, großskalige Bildungsdaten, simulierte Agenten, die von konkurrierenden intrinsischen Motivationen angetrieben werden, sowie Humanstudien zu Erklärungen unter Unsicherheit. Jedes Kapitel ist so aufgebaut, dass sein Modell und seine Evidenz eine Phase des Lebenszyklus adressieren und dabei denselben Maßstab verwenden wie die übrigen Kapitel. Der Beitrag ist daher kein Modell einer einzelnen kognitiven Fähigkeit, sondern eine einheitliche Theorie repräsentationaler Dynamik, in der Konstruktion, Kuratierung und Kommunikation Phasen desselben Prozesses sind und nur gemeinsam optimierbar werden.

Table of Contents

Acknowledgments	vii
Abstract	ix
Zusammenfassung	xi
Table of Contents	xiii
Notation	xvii
1. Overview	1
1.1. Introduction	1
1.2. Research objective and scope	3
1.3. Structure of this thesis	4
 I. THE PROBLEM BEHIND THE PROBLEM REPRESENTATION	 7
2. Principle: internal representation and its missing dynamics	9
2.1. Computable thinking	9
2.2. One shared blind spot	12
2.3. Why representational dynamics resist	18
2.4. The lifecycle of representational vocabulary	22
2.5. Summary	25
3. Formalism: from latent variables to scalable libraries	27
3.1. Generative models and latent inference	27
3.2. Agent-centric learning	29
3.3. Programs as structured latent representation	36
3.4. World models as learned latents	40
3.5. Other agents as latent systems	44
3.6. Compression as the evaluative principle	48
3.7. Summary	52
 II. CONSTRUCT: HOW STRUCTURED KNOWLEDGE IS ACQUIRED	 55
4. Learning leaves traces: path dependence in human knowledge	57
4.1. Introduction	57
4.2. Background	59
4.3. Methods	62
4.4. Simulations	66
4.5. Behavioral experiment	68
4.6. Curriculum effects	76
4.7. Discussion	78
4.8. Summary	80
5. What learners share: transferable structure across learners	81
5.1. Introduction	81
5.2. Background	82
5.3. Methods	84

5.4. Results	89
5.5. Discussion	94
5.6. Summary	95
III. CURATE: HOW STRUCTURED KNOWLEDGE IS SELECTED	97
6. Curiosity is not free: how exploration shapes representation	99
6.1. Introduction	100
6.2. Background	101
6.3. Methods	103
6.4. Experiment 1: Tabular Agent	110
6.5. Experiment 2: Dreamer Agent	114
6.6. Discussion	122
6.7. Summary	123
7. Not everything is worth pursuing: agent-centric representation curation	125
7.1. Introduction	125
7.2. Background	127
7.3. Methods	129
7.4. Experiment 1: Closed-vocabulary causal learning in humans	134
7.5. Experiment 2: Matched simulations	137
7.6. Experiment 3: Open-vocabulary symbolic construction	138
7.7. Discussion	141
7.8. Summary	143
IV. COMMUNICATE: HOW STRUCTURED KNOWLEDGE IS SHARED	145
8. Explaining how: communication as action enablement	147
8.1. Introduction	148
8.2. Background	149
8.3. Methods	152
8.4. Experiment 1: Explanation Utility	155
8.5. Experiment 2: Explanation Generation	159
8.6. Discussion	162
8.7. Summary	163
V. CONCLUSION & FUTURE DIRECTIONS	165
9. General discussion	167
9.1. What the thesis established	167
9.2. Where current paradigms fall short	174
9.3. The future directions	176
9.4. Closing reflection	183
VI. APPENDIX	185
A. Additional Material for Chapter 4	187
A.1. Method details	187
A.2. Data details	193
A.3. Paradigm and spans of three observers	195
A.4. Supplemental simulation results	196

A.5. Supplemental behavioral analyses	199
A.6. Model fitting details	203
B. Additional Material for Chapter 5	209
B.1. ELBO of the hierarchical SSM	209
B.2. Baseline models and datasets	212
B.3. PSI-KT model architecture	216
B.4. Prediction and generalization experiments details	218
B.5. Learner-specific representations analysis	221
B.6. Graph inference analysis	227
B.7. Ablation study	233
C. Additional Material for Chapter 6	235
C.1. DreamerV3 architecture and implementation	235
C.2. Information gain variants	237
C.3. Nonstationarity of intrinsic rewards	240
C.4. Blahut-Arimoto and factorized mutual information	243
D. Additional Material for Chapter 7	247
D.1. Formal results for representation empowerment	247
D.2. Closed-vocabulary causal-learning environment	252
D.3. Human study supplementary results	254
D.4. Matched-simulation supplementary results	258
D.5. Open-vocabulary Curator-Actor implementation and results	261
E. Additional Material for Chapter 8	269
E.1. Environment design and map construction	269
E.2. Experiment 1: LLM compilation protocol	271
E.3. Experiment 1: Participant details and pre-registration	275
E.4. Utility-guided generation details	277
Bibliography	281

Notation

The notation is influenced by Goodfellow et al. (2016).

Tensors, Matrices, Vectors, Numbers

a	A scalar
$\mathbf{a}$	A column vector
$\mathbf{A}$	A matrix
$\mathbf{A}$	A tensor
a_i or $[\mathbf{a}]_i$	The i th entry of the vector $\mathbf{a}$
$A_{i,j}$ or $[\mathbf{A}]_{i,j}$	The (i, j) th entry of the matrix $\mathbf{A}$ (row i , column j)
$[\mathbf{A}]_{i,:}$ (or $[\mathbf{A}]_{:,j}$)	The i th row (or j th column) of the matrix $\mathbf{A}$
$A_{i,j,k}$ or $[\mathbf{A}]_{i,j,k}$	The (i, j, k) th entry of the tensor $\mathbf{A}$
$\text{vec}(\mathbf{A}), \text{vec}(\mathbf{A})$	Matrix/tensor flattened into a vector; convention implies $\text{vec}(\mathbf{ABC}) = (\mathbf{C}^\top \otimes \mathbf{A}) \text{vec}(\mathbf{B})$
$\text{diag}(\mathbf{a})$	The square matrix with vector $\mathbf{a}$ on the diagonal and zeros elsewhere
$\text{diag}(\mathbf{A})$	The vector containing the diagonal elements of the matrix $\mathbf{A}$
$\text{diag}(\mathbf{A}_1, \dots, \mathbf{A}_L)$	A block-diagonal matrix with diagonal blocks given by square matrices $\mathbf{A}_1, \dots, \mathbf{A}_L$
$\text{Tr}(\mathbf{A}), \det(\mathbf{A})$	Trace and determinant of a matrix $\mathbf{A}$
$\ \mathbf{a}\ _2$	L_2 norm of vector $\mathbf{a}$, i.e. $\ \mathbf{a}\ _2^2 = \mathbf{a}^\top \mathbf{a}$
$\text{eig}(\mathbf{A}) := \{(\lambda_k, \mathbf{e}_k)\}_k$	Eigendecomposition of the matrix $\mathbf{A}$, eigenpairs $(\lambda_k, \mathbf{e}_k)$ satisfy $\mathbf{A}\mathbf{e}_k = \lambda_k \mathbf{e}_k$
$(\lambda_k(\mathbf{A}), \mathbf{e}_k(\mathbf{A}))$	k th eigenpair (eigenvalue, eigenvector) of matrix $\mathbf{A}$
$\mathbf{A} \otimes \mathbf{B}$	Kronecker product of two matrices, For two vectors $\mathbf{a}, \mathbf{b}$, one has $\mathbf{a} \otimes \mathbf{b}^\top = \mathbf{a}\mathbf{b}^\top$
$\mathbf{A} \odot \mathbf{B}, \mathbf{A} \oslash \mathbf{B}, \mathbf{a} \odot \mathbf{b}$	Elementwise multiplication (Hadamard product) of two tensors, matrices, vectors
$\mathbf{A} \oslash \mathbf{B}, \mathbf{A} \oslash \mathbf{B}, \mathbf{a} \oslash \mathbf{b}$	Elementwise division (Hadamard division) of two tensors, matrices, vectors
$\mathbf{A}^{\odot 2}, \mathbf{A}^{\odot 2}, \mathbf{a}^{\odot 2}$	Elementwise square of a tensor, matrix, vector
$\mathbf{A}^{\odot 1/2}, \mathbf{A}^{\odot 1/2}, \mathbf{a}^{\odot 1/2}$	Elementwise square root of a tensor, matrix, vector

Neural Networks

The layer number is indicated by parenthesized superscripts (l) .

L	Total number of layers
$d^{(l)}$	Number of parameters in layer l ; total number of parameters is $D = \sum_{l=1}^L d^{(l)}$
$\boldsymbol{\theta}^{(l)} \in \mathbb{R}^{d^{(l)}}$	Parameter vector of layer l , potentially empty for parameter-free layers like activations
$h^{(l-1)}$	Number of (hidden) inputs fed into layer l
$M := h^{(0)}, C := h^{(L)}$	Input feature dimension, output dimension (number of classes for classification)
$\mathbf{z}^{(l-1)} \in \mathbb{R}^{h^{(l-1)}}$	(Hidden) features fed into layer l (output of layer $l-1$)
$\mathbf{x} := \mathbf{z}^{(0)}, \mathbf{f} := \mathbf{z}^{(L)}$	Input to the neural network, and its prediction for input $\mathbf{x}$
$f_{\boldsymbol{\theta}^{(l)}}^{(l)}$	Layer l parameterized by $\boldsymbol{\theta}^{(l)}$, mapping input $\mathbf{z}^{(l-1)}$ to output $\mathbf{z}^{(l)}$
$f_{\boldsymbol{\theta}} := f_{\boldsymbol{\theta}^{(L)}}^{(L)} \circ \dots \circ f_{\boldsymbol{\theta}^{(1)}}^{(1)}$	Sequential feedforward neural network parameterized by $\boldsymbol{\theta}$, maps input $\mathbf{x}$ to output $\mathbf{f}$
$\boldsymbol{\theta}$	Parameter vector, concatenation of parameters over layers $\boldsymbol{\theta} := (\boldsymbol{\theta}^{(1)\top}, \dots, \boldsymbol{\theta}^{(L)\top})^\top$

Statistics

$\mathcal{U}(\{1, \dots, N\})$	Uniform distribution over $\{1, \dots, N\}$
--------------------------------	---

$\mathcal{N}(x \mid \mu, \sigma^2)$	Uni-variate normal/Gaussian distribution of random variable x , with mean μ , positive variance σ^2 , and density $\mathcal{N}(x \mid \mu, \sigma^2) = 1/\sigma\sqrt{2\pi} \exp[-1/2((x-\mu)/\sigma)^2]$
$\mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma})$	Multi-variate normal/Gaussian distribution of random vector $\mathbf{x}$ with mean vector $\boldsymbol{\mu}$, PSD covariance matrix $\boldsymbol{\Sigma}$, and density $\mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) = 1/(\sqrt{2\pi} \det \boldsymbol{\Sigma}) \exp[-1/2(\mathbf{x} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x} - \boldsymbol{\mu})]$
$\text{Cat}(c \mid \mathbf{p})$	Multinomial/Categorical distribution with probabilities $\mathbf{p}$ for categories c

Miscellaneous

$\log$	The natural logarithm (base e, <i>i.e.</i> $\log(e) = 1$)
$\text{onehot}(c)$	One-hot vector of class c with $\text{onehot}(c) = \delta_{i,c}$
$\text{softmax}(\mathbf{a})$	Softmax probabilities of the logits $\mathbf{a}$, $[\text{softmax}(\mathbf{a})]_c = \exp(a_c) / \sum_{i=1} \exp(a_i)$
$\delta_{i,j}, \delta(\mathbf{x} - \mathbf{a})$	Kronecker delta ($\delta_{i,i} = 1$ and $\delta_{i,j \neq i} = 0$), Dirac delta distribution
$(\mathbb{X} \rightarrow \mathbb{Y})$	Signature of a function that maps between $\mathbb{X}$ and $\mathbb{Y}$
$\{\mathbf{x}_n\}$ or $\{\mathbf{x}_n\}_n$	A set/collection of vectors $\mathbf{x}_1, \mathbf{x}_2, \dots$ over the index set implied by n
$\hat{\mathbf{e}}_i$	Unit vector in direction i , <i>i.e.</i> $\hat{\mathbf{e}}_i = \text{onehot}(i)$
$\mathbf{1}_m$	An m -dimensional vector containing ones everywhere
$\log(\mathbf{a}), \exp(\mathbf{a})$	Elementwise natural logarithm and exponential function of a vector
$m_{\theta_t}(\boldsymbol{\theta})$	Local approximation of the loss in around $\boldsymbol{\theta}_t$

Acronyms & Abbreviations

<i>E.g.</i> or <i>e.g.</i>	For example (<i>exempli gratia</i>)
<i>Etc.</i> or <i>etc.</i>	And so on (<i>et cetera</i>)
<i>I.e.</i> or <i>i.e.</i>	That is (<i>id est</i>)
<i>I.i.d.</i> or <i>i.i.d.</i>	Independent and identically distributed
<i>W.r.t.</i> or <i>w.r.t.</i>	With respect to
API	Application Programming Interface
CNN	Convolutional neural network
CPU	Central processing unit
DNN	Deep neural network
GPU	Graphics processing unit
MAP	Maximum a posteriori (estimation)
MC	Monte Carlo
ML	Machine learning
MLE	Maximum likelihood estimation
MLP	Multi-layer perceptron
NGD	Natural gradient descent
ResNet	Residual (neural) network

1.1 Introduction

1.1 Introduction	1
1.2 Research objective and scope	3
1.3 Structure of this thesis . . .	4

Cognitive science began with an audacious wager: that thinking can be explained without mysticism, as the rule-governed manipulation of internal representations—an organized form of information processing. The wager has been productive, giving us formal models of perception, language, reasoning, and learning; philosophical analyses of meaning and content; and empirical psychological models of how minds actually behave. But from its earliest days, the program this wager opened generated two different and often competing questions.

One question asks what is possible: what forms of computation can transform evidence into knowledge, perception into action, and experience into generalization? This is the question that invites formalization—algorithms, complexity classes, inference procedures, analyses of when computation succeeds or fails. It makes a mind look like a machine, and Turing’s work was its turning point, because it made “a procedure” a mathematically precise object. A long lineage followed, from physical symbol systems through connectionist learning to Bayesian models of induction and probabilistic programs. Each proposed a different computational story about how minds process information.

The other question asks what makes any of this content-bearing in the first place: what makes an internal state a representation *of* something, what makes a procedure a rule rather than a mere regularity, what makes a purported internal symbol more than an inscription. This is the question that philosophy, particularly in pragmatist traditions, has kept alive: rules do not carry their own applications on their sleeves, and representations do not interpret themselves; both require practices of use, training, correction, and coordination. On this view, cognition is the capacity to stabilize ways of doing and interpreting, often socially and often tacitly. An expert does not merely possess more information than a novice; she possesses a different way of seeing, one built through years of practice that has reorganized how she represents problems.

Much of cognitive science can be read as an attempt to keep both questions in view at once, and the frameworks that have most moved the field—Marr’s levels, resource-rational analysis—refuse to let either dissolve the other. They also make salient a constraint that neither question alone foregrounds: cognition is the work of a finite agent operating under limits of time, memory, and data. This thesis argues that one place these pressures meet most acutely, and where current frameworks have least to say, is in *the dynamics of representation*: how representational vocabulary is built, maintained, and shared over time by an agent who cannot afford to represent everything.

Every major tradition in cognitive science has had to take a stance on representation: on the format of knowledge, its organization, and the operations it affords. Much progress has come from answering that question in comparatively static terms. Once a representational scheme is given, symbolic systems can reason over it, neural networks can tune it, and Bayesian models can infer within it. The harder question is how

such representations themselves come into being and how they persist as resources for future thought.

This harder question inherits all three pressures. It is about formal possibility, because we want to know which kinds of representational change can in principle succeed; about content, because what is constructed must be usable as a representation of something; and about resource bounds, because no finite agent can entertain every distinction, retain every abstraction, or communicate its internal structure in full. A learner must instead manage representational resources selectively—deciding what to construct, what to reuse, what to revise, what to forget, and what to make explicit for others. Existing frameworks illuminate parts of this process, but none yet gives a *process-level account of representational vocabulary* as something accumulated, curated, and socially coordinated under resource constraints. This thesis takes that missing dynamic as its object: the construction and maintenance of representational structure, and the way that structure reshapes what can subsequently be learned, computed, understood, and shared.

The claim. This thesis argues that the missing account centers on a specific kind of theoretical object: a generative library. A library, as the term is used here, is a repertoire of reusable building blocks—programs, abstractions, subroutines, schemas—over which new hypotheses, actions, and interpretations can be composed. It is the representational vocabulary that stands between raw experience, which is too unstructured to support efficient generalization, and a monolithic world model, which is too unwieldy to inspect, curate, or communicate.

Cognitive science has already repeatedly introduced constructs that play the role of library entries, e.g., schemas, frames, scripts, chunks, cognitive maps, and learned abstractions. Nor are questions about the acquisition, revision, or reuse of such entries wholly absent from these traditions. What is missing is a unified account of the *library-level problem*: how a repertoire of reusable representational resources is built, kept useful, and coordinated across bounded agents so that it can support ongoing thought and action. The novelty, then, is not the claim that cognitive systems use reusable structure. It is the claim that the resource-rational management of such structure—its construction, curation, and communication—is itself a central computational problem for intelligence.

The guiding claim of this thesis is:

Intelligence depends on the resource-rational lifecycle management of a generative library: a repertoire of reusable representational resources constructed from experience, curated over time, and communicated across agents so that it can be exploited in thought and action.

Three properties make libraries the right unit of analysis. First, library entries are *generative*: they are, beyond labels or feature vectors, executable procedures that can be run to produce predictions, simulate outcomes, compose with other entries, and guide action. Second, libraries are *compositional*: complex hypotheses can be built by combining simpler entries, so a small library can support a large space of thoughts and behaviors. Third, libraries are *inspectable and communicable*: unlike the weights of a neural network, a library entry can in principle be examined, refactored, taught, or explained. These properties are exactly what is needed to study how representational vocabulary changes over time.

A unifying lens across fields. The library framework redescribes problems often studied in isolation as questions about constructing, curating, and communicating reusable representational resources. It treats development, expertise, machine learning, reinforcement learning, and pedagogy in terms of what entries learners have, how they acquire them, when they reuse or revise them, and how they align them with others.

In this lens:

- ▶ **Cognitive science** asks how learners construct new library entries from experience, why learning is path-dependent once earlier entries shape later hypothesis spaces, why experts and novices differ in the representational resources they can bring to bear, and why structured errors reveal the organization of an internal library.
- ▶ **Machine learning and reinforcement learning** ask how learned representations, options, hierarchical policies, and world-model components become reusable entries in a computational library; how such entries compress search and action spaces; and how they should be retained, revised, or discarded as tasks change.
- ▶ **Education and pedagogy** ask how teaching can guide library construction in another agent: how curricula sequence the acquisition of reusable entries, how explanation links new material to the learner's existing repertoire, and how instruction aligns the teacher's and learner's representational libraries.

The thesis instantiates this lens through a series of computational models and empirical studies, each targeting a different stage of the library lifecycle.

1.2 Research objective and scope

The thesis develops an account of how representational vocabulary is (i) constructed, (ii) actively curated, and (iii) communicated between agents. Across these domains, the core research question is:

How do resource-bounded agents construct, select, and share libraries of reusable abstractions that make learning, planning, generalization, and communication tractable?

Methodological approach. The common formal structure across the thesis is latent inference over generative models of agents. We observe choices, trajectories, and learning curves; posit hidden structure—knowledge states, programs, world models, communicative intent—and recover that structure from behavior using probabilistic or variational methods. What changes across chapters is the nature of the latent structure (i.e., the entries of the generative library) and the formal machinery appropriate to it. The thesis therefore combines conceptual analysis with computational modeling, large-scale behavioral data, controlled laboratory experiments, simulated agents, and human studies of explanation.

Construct: how are libraries built? When an agent learns by building reusable structure—compressing recurring patterns into abstractions, or chunking frequent sequences into subroutines—what it learns first changes the landscape for later learning. Early abstractions become the building blocks of later hypotheses. What is already in the library determines what representations are short, natural, or easy to search. This creates path dependence, which means different curricula induce different libraries, and those libraries support different patterns of generalization.

The thesis studies this in two settings: controlled laboratory sequence learning, and large-scale educational data with prerequisite structure.

Curate: how are libraries maintained? Construction alone is insufficient; a library must also be managed. Representations that were useful in one regime may become misleading in another, and a finite agent cannot keep every abstraction indefinitely. This raises a curation problem: what should be explored, what should be retained, and what should be discarded? The thesis treats intrinsic motivation as a signal about representational value, then argues that a satisfactory curation criterion must be defined over internal representations rather than external states alone.

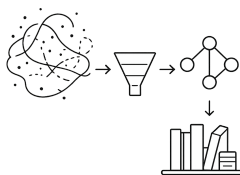
Communicate: how are libraries shared? If representations are shaped by learning history, then different agents do not merely know different facts; they may inhabit different representational vocabularies. Communication is therefore not just the transmission of propositions, but the alignment of representational resources. The thesis studies this most directly in pedagogical explanation: how an informed speaker helps a partially informed listener act effectively, first when the listener is known and then when the listener must be inferred.

1.3 Structure of this thesis

The thesis is organized around three substantive parts, framed by an overview and an appendix. **Part I** develops the conceptual and formal vocabulary of the thesis. **Part II** studies how structured knowledge is acquired. **Part III** studies how structured knowledge is selected and maintained. **Part IV** studies how structured knowledge is shared. An extensive appendix (**Part VI**) complements the main text with additional methodological and empirical detail.

Part I contains two background chapters. The first makes the conceptual case for the representational dynamics gap. Every major computational tradition presupposes a representational vocabulary that it cannot itself explain, and the thesis argues that this gap becomes tractable when studied through the lifecycle of a generative library. The second introduces methods used throughout the thesis: generative models and latent inference, reinforcement learning, program induction, multi-agent inference, and compression as a resource-rational principle. Together, these chapters provide the conceptual motivation and formal vocabulary for the empirical work that follows.

Part II: how do agents build structured internal representations from experience, and why does learning order matter?



Part II addresses the construction stage of the lifecycle in two complementary settings. **Chapter 4** studies controlled human sequence learning and shows that behavior is best explained by hierarchical, reusable programs learned under memory and search constraints. Its central claim is that library construction is path-dependent: what a learner has already compressed determines what they can represent efficiently next. **Chapter 5** shifts to naturalistic educational data and asks whether traces of such structured construction remain recoverable at scale. It shows that learner performance is better explained by a generative model with latent knowledge states, interpretable learner traits, and a shared prerequisite graph than by flat skill models. Together, the two chapters show the same construction logic in two regimes, one tightly controlled and one ecologically large-scale.

Remark 1.1 Chapter 4 is based on the following peer-reviewed papers:

H. Zhou, D. G. Nagy, and C. M. Wu (2024b). “Harmonizing Program Induction with Rate-Distortion Theory”. *Proceedings of the 46th Annual Conference of the Cognitive Science Society*. Edited by S. Frank, M. Toneva, A. Mackey, and E. Hazeltine. *Cognitive Science Society*

H. Zhou, D. G. Nagy, P. Dayan, and C. M. Wu (2026b). “Path-dependent program induction under resource constraints explains human sequence learning”. *arXiv preprint arXiv:2606.20623*

Contributions: **HZ:** project conceptualization, model implementation, data analysis, literature review, manuscript writing and editing. **DN:** project conceptualization, manuscript writing and editing. **PD:** methodology feedback, manuscript writing and editing. **CW:** project conceptualization, supervision, manuscript writing and editing.

Remark 1.2 Chapter 5 is based on the following peer-reviewed paper:

H. Zhou, R. Bamler, C. M. Wu, and A. Tejero-Cantero (2024a). “Predictive, scalable and interpretable knowledge tracing on structured domains”. *The Twelfth International Conference on Learning Representations*

Contributions: **HZ:** project conceptualization, theoretical development, model implementation, results analysis, literature review, manuscript writing and editing. **RB:** theoretical development, manuscript writing and editing. **CW*:** project conceptualization, supervision, manuscript writing and editing. **AT*:** project conceptualization, supervision, manuscript writing and editing. *equal contribution.

Part III turns from construction to active management. **Chapter 6** asks how different intrinsic motivation signals shape exploration and representation learning, comparing novelty, information gain, and empowerment in both fixed and learned representational settings. Its main result is that the choice of intrinsic drive has direct representational consequences, and that these consequences depend on the feedback loop between motivation and the learned world model. **Chapter 7** then proposes a normative criterion for curation itself: a library should be evaluated by how much diverse and controllable future structure it makes available. This chapter formalizes that criterion and shows how it can guide the admission and pruning of abstractions. Together, the curation chapters move from descriptive contrasts between exploration drives to a principled account of representational value.

Part III: how do agents decide what is worth representing, keeping, and revising?



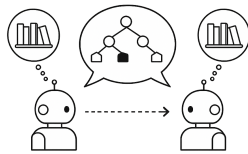
Remark 1.3 Chapter 6 is based on the following peer-reviewed paper:

F. Mantiuk*, H. Zhou*, and C. M. Wu (2025). “From Curiosity to Competence: How World Models Interact with the Dynamics of Exploration”. *Proceedings of the 47th Annual Conference of the Cognitive Science Society*. Edited by A. Ruggeri, D. Barner, C. Walker, and N. Bramley. San Francisco, CA: *Cognitive Science Society*

Contributions: **FM*:** project conceptualization, model implementation, results analysis, literature review, manuscript writing and editing. **HZ*:** project conceptualization, model implementation, results analysis, literature review, manuscript writing and editing. **CW:** project conceptualization, supervision, manuscript writing and editing. *equal contribution.

Remark 1.4 Chapter 7 is based on the following peer-reviewed papers:

Part IV: how do agents communicate internal structure so that another agent can act with it?



H. Zhou, F. Mantiuk, D. G. Nagy, and C. M. Wu (2025). “Agent-centric learning: from external reward maximization to internal knowledge curation”. *Reinforcement Learning Conference: Finding the Frame Workshop*

F. Dai*, H. Zhou*, and A. Gopnik (2025). “Hierarchical World Models with Causal Curation for Generalizing Agents”. *NeurIPS 2025 Workshop on Bridging Language, Agent, and World Models for Reasoning and Planning*

H. Zhou*, F. Dai*, C. M. Wu and A. Gopnik (Under Review). “What Is Worth Representing? Representation Empowerment for Continual Model Construction”.

Contributions: **FD***: project conceptualization, theoretical development, model implementation, results analysis, literature review, manuscript writing and editing. **HZ***: project conceptualization, theoretical development, model implementation, results analysis, literature review, manuscript writing and editing. **FM**: project discussion. **DN**: project conceptualization, theoretical development, manuscript editing. **CW**: supervision, manuscript editing. **AG**: supervision, manuscript editing. *equal contribution.

Part IV studies communication as representational alignment. The procedural explanation chapter treats explanation as action enablement: an explanation is good insofar as it can be compiled into executable guidance for a partially informed listener. It develops a utility-based model of explanation quality and tests that model against human judgments and navigation behavior, then uses the same utility as a target for explanation generation. Together, this chapter argue that communication is not a separate problem from representation, but one more stage in the lifecycle of the library.

Remark 1.5 Chapter 8 is based on the following peer-reviewed paper:

H. Zhou, B. Besch, C. M. Wu, and T. Gerstenberg (2026a). “Effective Explanations Support Planning Under Uncertainty”. *Proceedings of the 48th Annual Conference of the Cognitive Science Society*

Contributions: **HZ**: project conceptualization, model implementation, results analysis, literature review, manuscript writing and editing. **BB**: model implementation. **CW**: project conceptualization, manuscript editing. **TG**: project conceptualization, supervision, manuscript editing.

The thesis concludes by drawing these threads together. Its final chapter revisits the central questions through the single lens of the library lifecycle, argues that construction, curation, and communication form a coupled system rather than three independent topics, and identifies the open problem that remains: how to optimize the full representational lifecycle jointly rather than one stage at a time.

Part I.

The problem behind the problem
representation

Principle: internal representation and its missing dynamics

2.

Every framework that made thinking computable did so by assuming a representational vocabulary whose origin it did not explain. Turing needed symbols on a tape; Newell and Simon needed production rules; connectionist networks need an architecture; Bayesian models need a hypothesis space. In each case, the vocabulary is treated as already available for the purposes of the model. Where it comes from, how it changes, and what happens when it stops working—these questions have been deferred at every turn.

In [Chapter 1](#), I called this the lifecycle of representational resources and organized it into three stages: construct, curate, communicate. This chapter makes the case that such a lifecycle is genuinely needed. We trace the gap from its philosophical origins through three computational traditions, each of which defers the same question. Drawing on pragmatism and resource rationality, we explain why the gap persists and what a solution requires. The chapter then proposes generative libraries as a tractable theoretical object. Its aim is to show that the problem is real across traditions, structurally deep in its conditions for representation, and consequential for adaptation when existing vocabularies fail.

2.1 Computable thinking . . .	9
2.2 One shared blind spot . . .	12
2.3 Why representational dynamics resist	18
2.4 The lifecycle of representa- tional vocabulary	22
2.5 Summary	25

2.1 Computable thinking

2.1.1 The dual commitments in modeling the mind

Every theory of mind must answer two questions. The first is about process: by what mechanism does thinking proceed? The second is about substance: what does thinking operate over, and what are the basic units of thought? These cannot be answered independently, since a process is defined only over the contents it can manipulate, and those contents become cognitively useful only through the processes that operate on them. The shape any answer to one question can take is constrained by the answer to the other. Pre-computational philosophy can be read, in large part, as the attempt to characterize this mutual dependence, and the tension that effort exposed organizes the problems this thesis takes up.

The empiricist tradition, from Locke (1690) through Hume (1739), gave an answer to both questions. It treated sensory impressions as the basic material of thought and association as the process that connects them through contiguity, resemblance, and causality. In modern terms, this view treats cognition as the extraction of statistical regularities from experience (Rescorla, 2014; Saffran et al., 1996; Shanks, 1995). The framework's appeal is that it answers both questions at once with a process that is plainly observable: agents do form links between things that co-occur. However, associative learning alone does not specify how learned regularities become the kinds of reusable structures that support systematic generalization¹. That requires some explanation of how elements can be recombined and applied beyond the learned contexts (Lake and Baroni, 2023). The point, then, is not that empiricism cannot allow structure to develop, but that it makes this development of

1: For example, learning that fire co-occurs with heat is not yet an account of how an agent can form and use the novel concept of a fire-resistant material.

2: By *nativist*, I mean the view that some structure of the mind is innate rather than learned entirely from experience.

3: Kant's basic concepts are the *Kategorien*, and his organizing forms are the *Anschauungsformen*; see Kant (1908).

4: We use representational vocabulary to refer to the set of primitives and their combinatorial grammar—the building blocks from which beliefs are assembled.

5: Turing's formalization and its proven equivalence to Church's λ -calculus (Church, 1932) established that "computable" has a unique, robust definition independent of the particular formalism chosen. See Turing et al. (1936).

6: "I propose to consider the question, 'Can machines think?'" Turing (1987, p. 433).

7: "Instead of trying to produce a programme to simulate the adult mind, why not rather try to produce one which simulates the child's?" Turing (1987, p. 456).

representations its central burden: it must explain how initially simple experiences become resources for general thought.

Kant, and the nativist tradition more broadly², approached the same dependence from the opposite direction: experience is possible only because the mind contributes basic concepts and organizing forms in advance (Kant, 1908)³. Learning therefore depends on structure the mind already brings to the world. In contemporary cognitive science, this idea reappears in core-knowledge accounts (Gopnik and Meltzoff, 1997; Kinzler and Spelke, 2007; Lake et al., 2017), and in Simon's account of intelligent action as a fit between internal organization and task environment (Simon, 2014). The common commitment is that learning does not begin from an unstructured intake of experience. It depends on a representational vocabulary⁴ that determines, at least initially, what can be distinguished, grouped, and generalized.

Yet this starting condition does not remove the developmental problem. Even if some structure precedes learning, one must still explain how it becomes adequate to new domains, is extended through use, and is revised when it fails. The contrast, then, is not between learning and structure: both empiricism and nativism must explain how representational vocabularies come to have the form they do. Empiricism places the burden on construction from experience; nativism on the specification of initial structure. The shared gap concerns how the right structure becomes available, remains useful, and can be revised or shared as an agent's problems change. To make this gap precise, we need the other half of the dual commitment: a formal account of process. The theory of computation supplied such an account by defining effective procedure, but thereby sharpened rather than solved the representational question—what vocabulary do procedures operate over, where does it come from, and how does it change through use?

2.1.2 The computational turn

Turing's (1936) formalization gave "procedure" a precise meaning: rule-governed manipulation of discrete symbols⁵. Any process describable as a definite procedure can, in principle, be mechanized, and the mind, Turing argued in 1950, might be such a machine⁶.

Two features of Turing's framework matter for what follows. First, Turing's *universal machine* showed that the description of a procedure can itself be encoded as symbols and manipulated by another procedure. An agent can, in principle, inspect and reorganize its own computational building blocks. In modern terms, programs can be treated as data: stored, read, composed, and executed. This makes it formally possible for a system to operate not only on first-order inputs, but also on descriptions of its own procedures. This possibility will matter later in **Part II**, where construction is treated as the process by which new computational and representational vocabulary are assembled from existing ones. Second, Turing envisioned what he called a "child machine": a system whose initial state is simple but whose internal organization changes through experience, acquiring new operations that reshape what becomes easy to compute⁷. The child machine matters for this thesis because it gives an early computational formulation of the developmental problem introduced above: cognition may depend not only on computing over a fixed vocabulary, but also on acquiring new resources through use. New building blocks change what the agent can efficiently compute, and those changed computations create

opportunities for further restructuring. Turing glimpsed this recursive picture but did not provide a theory of its dynamics.

The formal precision that Turing provided converged, in the 1950s and 1960s, with several other intellectual currents to produce the cognitive revolution. The core bet was that cognition is *computation* over internal *representations*—mental states are data structures, and thinking is the algorithmic manipulation (Fodor, 1975; Newell and Simon, 2007)⁸. The Physical Symbol System Hypothesis (PSSH) in Newell and Simon (2007)—A physical symbol system has the necessary and sufficient means for general intelligent action—gave this view a canonical formulation by treating intelligent action as the operation of internal symbols and symbol-manipulating processes. Its importance here is that it made internal representations explicit theoretical objects while treating their format as already fixed. It organized work around a shared vocabulary—production rules, problem spaces, search trees, goals, and subgoals—and asked what could be computed over it.

8: Shannon’s information theory provided a language for encoding and transmission (Shannon, 1948); Chomsky’s formal linguistics demonstrated compositional structure in natural language (Chomsky, 1956); and cybernetics modeled goal-directed behavior through internal state.

Frameworks after the PSSH fixed a representational vocabulary and then studied what it made computable. Turing’s child machine points to a different problem: vocabulary itself changes. When a child discovers counting, arithmetic becomes a stable way to state previously unavailable problems. It then enables new tasks—division, fractions, estimation—which generate further vocabulary. Earlier approaches leave this development undertheorized. They explain cognition within a vocabulary, but not how experience builds new vocabulary, how it reorganizes later learning, or how agents adapt when old vocabulary fails. Section 2.2 shows that this omission recurs across major computational traditions, and Section 2.3 connects it to the problems of construction, curation, and communication that organize the thesis.

2.1.3 The representational dynamics gap

We define the *representational dynamics gap* as the absence of a process-level account of how an agent’s representational vocabulary and the computational processes that operate over it jointly evolve through experience. Because agents have limited time, memory, attention, and computational capacity, this evolution is not simply a search for better representations: it is a resource-rational problem of what can be constructed, maintained, and revised at acceptable cost.

The two sides of this gap—representation and computation—cannot be separated. An agent’s representational vocabulary determines what computations are cheap, including which hypotheses can be concisely expressed, which inferences can be tractably performed, which plans can be efficiently composed. But building new representational structure is itself a computation (searching for compressible patterns, abstracting recurring procedures, refactoring obsolete schemas), and that computation is constrained by the current vocabulary. Representation shapes computation; computation reshapes representation. The gap concerns this coupled dynamics with the feedback loop between them.

In the standard form of each major framework, learning changes performance within a representational format whose basic structure is already fixed. Deep networks adjust weights within a fixed architecture (Goodfellow et al., 2016). Bayesian models update posteriors within a fixed hypothesis space (Tenenbaum et al., 2011). Reinforcement-learning agents revise values or policies within a fixed state representation (Botvinick

9: Each framework has extensions that partially address this—learned features, nonparametric priors, option discovery—while Section 2.2 examines these cases in detail.

et al., 2019; Dayan and Daw, 2008; Sutton, Barto, et al., 1998)⁹. In each case, computation operates within a representational format that is given in advance and does not change through the agent’s experience. Whether this separation is a convenient simplification or a structural feature of representation itself is the question that drives the rest of this chapter.

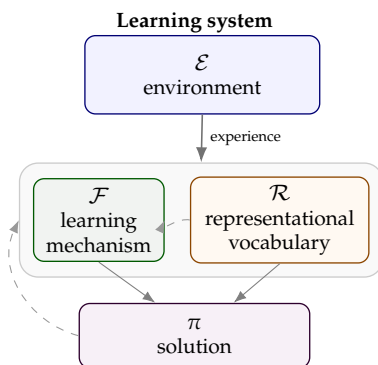
We will argue that the gap decomposes into three structural problems—*construction* (how representational vocabulary is built, and why the process is path-dependent), *curation* (how it is maintained as task demands shift, which is itself a computational process that consumes the resources it manages), and *communication* (how agents with different representational histories align their vocabulary to enable coordination)—and that this decomposition follows from what representations are under resource constraints, not merely from what cognitive science has failed to model. Section 2.2 first establishes that the gap persists across three independent computational traditions. Section 2.3 then derives the three-part decomposition from first principles, drawing on resource rationality and the pragmatist philosophical tradition.

2.2 One shared blind spot

The preceding Section 2.1 defined the representational dynamics gap as the missing process-level account of how representational vocabulary and the computations that operate over it jointly evolve through experience under resource constraints. We now test whether the gap persists across three traditions that have each proposed a different answer to the question of what representations are and how they are acquired: symbolic AI, which emphasizes hand-designed compositional structure; connectionism and deep learning, which learns distributed representations from data; and Bayesian cognitive science, which formalizes learning as probabilistic inference over structured hypothesis spaces.

Each tradition has produced substantial explanatory achievements. Yet, through the internal logic of their own theoretical commitments, each tradition defers the same coupled question on representational dynamics. These traditions differ in formalism and intellectual origin, yet each stops at the same point. This pattern suggests that the problem reflects a deep feature of the relationship between representation and learning, rather than an oversight of any one research program.

To make the comparison more precise, we describe every learning system as a triplet $(\mathcal{R}, \mathcal{F}, \mathcal{E})$:



Definition 2.1 (Learning system) A **learning system** is a triple $(\mathcal{R}, \mathcal{F}, \mathcal{E})$ where $\mathcal{R}$ is the *representational vocabulary* (the space of internal states the system can occupy), $\mathcal{F}$ is the *learning mechanism* (the process by which internal states change in response to data), and $\mathcal{E}$ is the *task environment* (the distribution of problems the agent faces). This decomposition is related to, but distinct from, Marr’s three levels of analysis (Marr, 2010): $\mathcal{E}$ specifies the computational-level problem (what the agent must do), while $\mathcal{R}$ and $\mathcal{F}$ together constitute the algorithmic level (how it does so). The key difference is that Marr’s framework treats each level as a static description of a fixed system; the representational dynamics gap concerns precisely the case where $\mathcal{R}$ and $\mathcal{F}$ jointly change through experience in $\mathcal{E}$.

Table 2.1: How the feedback-loop terms are instantiated across traditions. The table gives the working interpretation of $(\mathcal{R}, \mathcal{F}, \mathcal{E}, \pi)$ used in the sections that follow and indicates where each tradition typically leaves the abstraction step outside its standard theory.

Tradition	$\mathcal{R}$	$\mathcal{F}$	$\mathcal{E}$	π	Abstraction
Symbolic	A fixed signature and the space of well-formed symbolic expressions over it	Search over symbolic structures	Problems, examples, or rule-governed tasks	A proof, plan, or symbolic program	Usually left to the modeler through hand design or later knowledge engineering
Connectionist	An architecture together with its distributed activation space	Gradient-based parameter update	Input-output pairs, trajectories, or reward signals	A fitted parameter state and the behaviors it supports	Partially approximated by learned features, but architecture and training operators are largely fixed in advance
Bayesian	A hypothesis space with priors and compositional assumptions	Probabilistic inference over that space	Observations sampled from the environment	A posterior distribution or maximum a posteriori hypothesis	Sometimes approximated by model revision, but usually fixed by the initial hypothesis space

The representational dynamics gap, stated in these terms, asks: how are $\mathcal{R}$ and $\mathcal{F}$ jointly shaped by experience? When a learner acquires new representational structure, the format $\mathcal{R}$ changes—but this change also alters what $\mathcal{F}$ can efficiently compute, because the cost of searching, storing, and composing hypotheses depends on what building blocks are available. Each tradition has a well-developed account of $\mathcal{F}$ operating within a given $\mathcal{R}$. The gap concerns their coupled dynamics—the feedback loop in which experience produces new reusable structure that changes both $\mathcal{R}$ and the computational landscape of $\mathcal{F}$:

$$\mathcal{E} \xrightarrow{\mathcal{R}, \mathcal{F}} \pi \xrightarrow{\text{abstract}} (\mathcal{R}', \mathcal{F}') \xrightarrow{\mathcal{E}} \pi' \xrightarrow{\dots} \quad (2.1)$$

Here, $\mathcal{E} \rightarrow \pi$ denotes the learner acquiring a solution π (e.g., a hypothesis, a set of weights, or a program) from experience in the current environment. The *abstract* step extracts reusable structure from π and incorporates it into the representational format and learning mechanism, yielding $(\mathcal{R}', \mathcal{F}')$. The updated system then faces further experience, and the cycle continues. This loop, unfolding under resource constraints (Section 2.3.2) that make the order and cost of each step matter, is the specific mechanism we test each tradition against. Can the tradition instantiate every arrow in this diagram as part of its standard theoretical apparatus?

2.2.1 The symbolic tradition

The symbolic tradition, rooted in Newell and Simon (1972), treats intelligence as the manipulation of discrete symbols governed by explicit rules¹⁰. Inheriting from Turing’s account of symbol manipulation and from logical formalisms such as first-order logic and production systems, it encodes knowledge in structured, inspectable forms—semantic networks, scripts, or production rules—and explains inference as explicit rule application.

Formally, a symbolic system operates over a fixed *signature*: a set of sorts (types of objects the system can reason about), function symbols (operations that produce objects from objects), and relation symbols (predicates that hold between objects)¹¹. The representational format $\mathcal{R}$ is the space of well-formed expressions over this signature—the set of all statements, rules, and data structures the system can construct by combining its primitive vocabulary according to its grammar. The learning mechanism $\mathcal{F}$ is search, i.e., heuristic exploration of the space of well-formed expressions (e.g. means-ends analysis, backward chaining, or analogical retrieval), guided by goals and constrained by resources.

10: “A physical symbol system has the necessary and sufficient means for general intelligent action.” — Newell and Simon (1972, p. 116).

11: For example, a blocks-world planner might define sorts $\{Block, Location\}$, function symbols $\{stack(Block, Block), move(Block, Location)\}$, and relation symbols $\{on(Block, Block), clear(Block)\}$. The system can reason about any configuration of blocks, but cannot represent a concept like “fragile” unless the modeler adds it to the signature.

12: An agent that understands “John loves Mary” can, in principle, also understand “Mary loves John.”

Because symbolic systems are grounded in logic, they inherit several properties directly from their formal foundations: *compositionality* (complex expressions are built from simpler parts with predictable semantics)¹², and *inspectability* (every internal state is a readable expression). These properties make symbolic representations stable, readable, and reusable. But in the feedback loop of Equation (2.1), this strength also marks its limitation where it leaves largely unspecified the abstraction step by which a solution π is transformed into revised representational resources $(\mathcal{R}', \mathcal{F}')$. In other words, symbolic systems can manipulate and recombine an explicit vocabulary, but their standard theory does not explain how experience reorganizes that vocabulary itself.”

Symbolic vocabulary extension from reuse and induction. When both $\mathcal{R}$ (the signature) and $\mathcal{F}$ (the search strategy) are specified by the model designer, symbolic systems can exploit explicit primitives, rules, and search biases to reason compositionally. This is a strength when the relevant primitives for a domain can be specified in advance, as the Language of Thought hypothesis assumes (Fodor, 1975). It becomes a limitation when the environment demands distinctions or abstractions not anticipated by the initial vocabulary.

The symbolic tradition does include mechanisms for extending its effective repertoire. Explanation-based learning generalizes from an example by extracting a rule from the proof that it satisfies a goal concept (DeJong and Mooney, 1986; Mitchell et al., 1986). SOAR’s chunking mechanism abstracts successful subgoal traces into new production rules, creating shortcuts for future problem solving (Laird et al., 2012; Laird et al., 1986). Inductive logic programming infers symbolic rules from examples and background knowledge (Muggleton, 1991, 1994). These methods show that symbolic systems need not be static: they can add rules, compress derivations, and extend the procedures available to the agent.

Their limitation is not that representational change is impossible, but that it is usually constrained by a prespecified logical language, background theory, or hypothesis bias. EBL and chunking derive new rules from proofs or traces over already available predicates and sorts; ILP searches a clause space shaped by given predicates, modes, types, and background knowledge. Symbolic learning can therefore modify parts of $\mathcal{R}$, but typically leaves the governing format and search regime $\mathcal{F}$ fixed or externally specified. In the feedback loop of Equation (2.1), symbolic learning mechanisms therefore address important parts of $\mathcal{E} \xrightarrow{\mathcal{R}, \mathcal{F}} \pi$: feedback from successful problem solving can produce new symbolic shortcuts or rules. What remains less specified is how systems retain, revise, discard, or reorganize these resources as they accumulate, and how search should adapt to the expanded resource set. Connectionism responds by replacing explicit search over expanding symbolic structures with fixed optimization procedures that continuously reshape distributed representations.

2.2.2 The connectionist and deep learning tradition

Connectionism originated from multiple motivations, e.g., dissatisfaction with hand-designed symbolic vocabularies, inspiration from neural computation in the brain (Rosenblatt, 1958; Rumelhart et al., 1986), and the desire for systems that learn representations from data rather than having them specified in advance. Modern deep learning (Goodfellow et al., 2016), the most successful descendant of this tradition, trains

neural networks with billions of parameters through gradient-based optimization. Formally, the representational format $\mathcal{R}$ is determined jointly by the network architecture (i.e., number of layers, connectivity pattern, and activation functions) together with the parameter space $\theta \in \Theta \subseteq \mathbb{R}^d$. The learning mechanism $\mathcal{F}$ is typically gradient descent, $\theta_{t+1} = \theta_t - \eta \nabla_{\theta} \mathcal{L}(\theta_t)$ where ∇_{θ} is the gradient with respect to the parameters and $\mathcal{L}$ is the training objective. As θ changes, the representation changes with it—deep networks discover hierarchical features for vision, language, and control that would be difficult to specify by hand.

The limitation in this tradition has two forms, a *dynamics* problem and an *opacity* problem. (i) The dynamics problem is that the theory is strongest within a fixed task. Standard training theory studies the trajectory $\theta_0, \theta_1, \theta_2, \dots$ toward an end-state θ^* and does not explain how structure learned on one task becomes a resource for the next. The information bottleneck (Tishby et al., 2000) comes closest to studying representational change by characterizing a compression-then-fitting trajectory during a single training run (Remark 2.1). But this remains a within-task account, and these dynamics arise as a by-product of optimization rather than as a deliberately controlled descriptive model of human cognition. (ii) The opacity problem is that learned representations are hard to inspect. A learned vector $z \in \mathbb{R}^k$ encodes useful information, but it is difficult for a human—or another agent—to identify what that encoding contains, to refactor it when the task changes, or to communicate it in parts. Researchers therefore often impose additional structure when they want representations to be more separable and inspectable, for example through attention mechanisms (Vaswani et al., 2017a), modular networks (Andreas et al., 2016; Meunier et al., 2010), or disentangling objectives such as β -VAEs (Higgins et al., 2017). The dynamics problem would persist even for fully interpretable neural networks. Opacity compounds it by making curation and communication—which require access to individual representational components—difficult.

Continual learning as cross-task retention. Continual learning relaxes the single-task picture by asking how learned structure persists and transfers across tasks. It therefore moves closer to the feedback loop of Equation (2.1), but repairs only part of the problem.

Continual learning (Kirkpatrick et al., 2017; Rusu et al., 2016) aims to address one consequence of the dynamics problem, namely catastrophic forgetting across tasks. When new tasks arrive, previously learned representations should not be overwritten. But retention is not yet curation (Part III). Existing methods preserve past knowledge without isolating reusable components for later compositional reuse. Meta-learning (Finn et al., 2017) fixes it by learning an initialization θ_0 from which a small number of gradient steps yields good performance on a new task. This is a powerful form of learning-to-learn, but θ_0 still lies in a fixed parameter space Θ whose architecture is specified in advance. Meta-learning improves adaptation within a given representational family. It does not expand that family by adding new reusable structure.

In the feedback loop, the connectionist tradition keeps $\mathcal{F}$ fixed, and it handles $\mathcal{E} \xrightarrow{\mathcal{R}, \mathcal{F}} \pi$, in part, $\pi \xrightarrow{\text{abstract}} \mathcal{R}'$ well even when $\mathcal{R}$ is large. Continual learning further improves $\mathcal{F}$ so that it can handle a sequence of tasks or environments. These methods show that learned structure can persist and shape later computation. But what is carried forward is usually encoded in distributed weights or activations rather than in explicit reusable elements. As a cognitive model, this makes it difficult to explain how an agent isolates a previously learned substructure, treats it

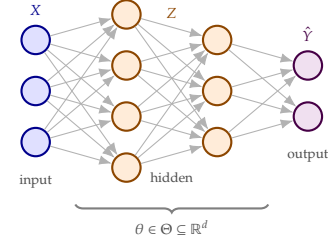


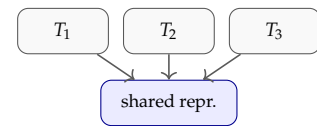
Figure 2.1: A feedforward neural network. Input X is transformed through hidden layers with learned parameters θ to produce output $\hat{Y}$. The intermediate representation Z is a distributed vector whose entries lack predefined semantic roles.

Remark 2.1 (The information bottleneck) Given input X , target Y , and hidden representation Z , the information bottleneck (Tishby et al., 2000) seeks a representation that minimizes:

$$\mathcal{J}_{\text{IB}} = \text{MI}(X; Z) - \beta \text{MI}(Z; Y),$$

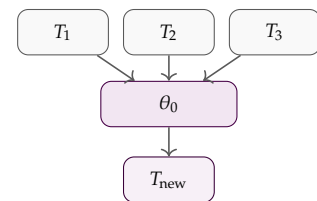
where $\text{MI}(X; Z)$ is the *rate* (information retained about the input) and $\text{MI}(Z; Y)$ is the *relevance* (information preserved about the target). During training, $\text{MI}(X; Z)$ first increases (fitting), then decreases (compression), while $\text{MI}(Z; Y)$ increases monotonically.

Continual learning



goal: preserve prior knowledge

Meta-learning



goal: adapt quickly to new tasks

Figure 2.2: Two strategies for multi-task representation. Top, continual learning preserves a shared representation across sequential tasks. Bottom, meta-learning distills tasks into an initialization θ_0 for rapid adaptation. Neither expands the representational vocabulary itself.

as a reusable unit, and deliberately revises or communicates it in a new setting. The unresolved gap is therefore not whether prior learning affects later learning, but how what is learned becomes an explicit resource for curation and communication.

2.2.3 Bayesian cognitive modeling

The contrast with the previous two traditions is now clear. Symbolic models make structure explicit, but give learning a limited role. Connectionist models make learning central, but often encode what is learned in forms that are difficult to isolate, revise, or communicate. Bayesian cognitive science (Griffiths et al., 2010; Tenenbaum et al., 2011) starts from a different question, namely how an agent should update its beliefs in light of evidence. This matters for the thesis because it combines a learning-theoretic perspective with more explicit objects of learning.

The Bayesian tradition frames cognition as inference under uncertainty. The agent maintains beliefs over a structured hypothesis space $\mathcal{H}$, which may consist of causal graphs (Griffiths and Tenenbaum, 2009), grammars (Goodman et al., 2008), logical theories (Tenenbaum et al., 2006), or programs (Lake et al., 2015), and updates those beliefs in light of data D via Bayes' rule:

$$p(h \mid D) \propto p(D \mid h) p(h), \quad h \in \mathcal{H}. \quad (2.2)$$

In the $(\mathcal{R}, \mathcal{F}, \mathcal{E})$ lens (Definition 2.1), $\mathcal{R} = \mathcal{H}$ defines the space of possible hypotheses, $\mathcal{E}$ supplies the observations or task structure to be explained, the prior $p(h)$ encodes inductive biases over $\mathcal{H}$, and $\mathcal{F}$ is Bayesian updating, which uses the likelihood of those observations under each hypothesis to obtain the posterior $p(h \mid D)$.

Unlike the symbolic and connectionist traditions, Bayesian cognitive science does not commit to a specific representational format. It operates primarily at Marr's computational level (Marr, 2010) by specifying what problem the agent solves (optimal inference given a generative model), while remaining agnostic about the representation. This generality is the tradition's strength, because it applies across many kinds of hypothesis spaces. However, the hypothesis space $\mathcal{H}$ is usually given, and so is the approximation scheme once exact inference becomes intractable. The next two paragraphs show why both assumptions matter.

Deferral 1: the hypothesis space is always given. Bayesian cognitive science has contributed in explaining important phenomena including few-shot generalization, causal learning, and compositional inference (Goodman et al., 2008; Griffiths and Tenenbaum, 2009; Tenenbaum et al., 2011). As a computational-level theory, it can strongly constrain what a process-level theory must explain but only works within a hypothesis space $\mathcal{H}$ that is already in place. Bayes' rule redistributes probability mass within $\mathcal{H}$. It does not explain how the agent acquires the primitives and structured hypotheses that make inference possible in the first place¹³. The question is displaced to a meta-space of possible hypothesis spaces, and the same gap reappears there.

Deferral 2: approximation is treated as an implementation detail. In practice, every Bayesian cognitive model relies on an approximation, e.g., MCMC sampling, variational inference, or amortized recognition networks, as exact Bayesian inference over any structured $\mathcal{H}$ is intractable (Dasgupta et al., 2018; Sanborn, 2017). Most Bayesian cognitive models treat this as an implementation detail, i.e., from Marr's

13: When $\mathcal{H}$ consists of programs, the model can represent extremely rich generative structure (Goodman et al., 2008; Lake et al., 2015). But the model still depends on a fixed set of primitives and composition rules, and the richer the language is, the more the explanation depends on that initial choice.

computation to algorithmic level. But the choice of approximation has direct consequences for learning dynamics, because it determines what structure the agent actually recovers from data.

Consider two cases that differ in whether the proposal distribution changes over time: importance sampling with a fixed proposal $q(h)$, and sequential Monte Carlo (SMC) (Doucet et al., 2001), where proposals adapt as inference unfolds¹⁴. With fixed-proposal importance sampling, inference episodes are independent: past successes and failures do not alter future proposals, so the feedback loop in Equation (2.1) has no abstraction step from π back to $(\mathcal{R}', \mathcal{F}')$. In SMC, past inference does shape future proposals by concentrating particles in promising regions. Yet only the distribution over hypotheses changes; the representational blocks and update format remain fixed.

Thus even adaptive approximations mainly redistribute computation within a pre-specified scheme. They explain belief change within that scheme, but not how experience builds new representational resources or inference procedures. The deeper issue is that asking which approximation an agent should use under finite time and memory turns Bayesian inference into *resource allocation*: which regions of $\mathcal{H}$ or $\mathcal{R}$ deserve effort, and which approximations are worth their cost? These are sequential decision problems. Resource-rational analysis (Lieder and Griffiths, 2020) begins to bridge this gap, but the algorithmic question—how agents jointly manage representational resources and inference budgets—remains open.

2.2.4 The remaining gap

The three traditions differ in formalism, intellectual history, and representational commitments, yet they converge on deferring the same step $\pi \xrightarrow{\text{abstract}} (\mathcal{R}', \mathcal{F}')$ in Equation (2.1) by which a solution becomes a revised representation vocabulary for the next cycle. Symbolic systems supply an explicit, inspectable vocabulary but no mechanism for extending it beyond recombination. Connectionist systems make representation a product of learning, but what is learned is distributed across weights rather than isolated into units that can be curated or communicated. Bayesian cognitive science gives a normative account of inference over a structured hypothesis space, but the space is given, and the approximations that make inference tractable are treated as implementation details rather than as learnable resources in their own right.

No tradition denies that the loop matters—symbolic AI acknowledges that experts and novices differ in production systems, deep learning researchers observe that pretrained representations shape downstream performance, and Bayesian modelers recognize that the choice of hypothesis space determines what can be learned. That three otherwise-dissimilar programs defer the same step is itself the argument that jointly updating $(\mathcal{R}, \mathcal{F})$ from experience is the right target for a new research programme. Section 2.3 asks why this is a gap and why it is difficult to solve, drawing on pragmatism and resource rationality to derive three constraints that any adequate account must satisfy. Section 2.4 then introduces the concrete object through which this thesis studies them.

14: A population of weighted particles approximates the posterior. At each step, low-weight particles are discarded and high-weight ones replicated, concentrating computation on high-posterior regions. See Figure 2.3.

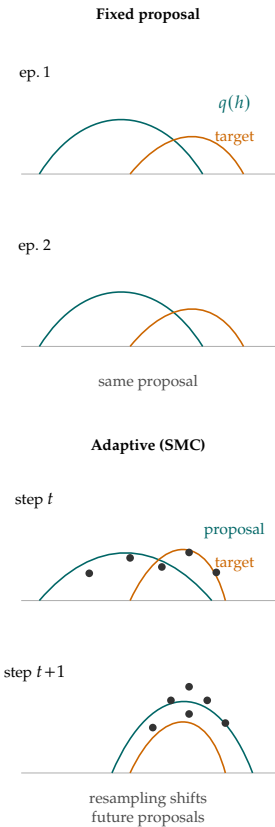


Figure 2.3: Fixed vs. adaptive inference. **Top:** importance sampling with a fixed proposal $q(h)$; episodes are independent. **Bottom:** SMC resampling concentrates particles on high-weight regions, adapting the proposal at each step.

2.3 Why representational dynamics resist

The preceding section established that the representational dynamics gap persists across different computational traditions. This section asks: why does it matter, and why is it so hard?

Representations are tools, and tools derive their meaning from practices of use (Gigerenzer, 1991). Consider a chess expert's chunk `KINGSIDE-FIANCHETTO` groups six pieces into one recognizable unit, compressing many move-evaluations into a single retrieval (Chase and Simon, 1973). However, transplanted to Chess960, where starting positions are randomized, the same chunk misfires because it silently presupposed regularities of standard opening theory. The meaning of any $\ell \in \mathcal{R}$ is therefore partly constituted by the history of its deployment in $\mathcal{E}$. This entails a formal consequence for the $(\mathcal{R}, \mathcal{F}, \mathcal{E})$ triplet, namely, that $\mathcal{R}$ and $\mathcal{F}$ cannot be characterized independently of $\mathcal{E}$. This is because the value of a representational resource is defined over the distribution of tasks it will be used on¹⁵. The pragmatist tradition identified this instability a century before computational cognitive science encountered it. Section 2.3.1 recasts its diagnosis in the $(\mathcal{R}, \mathcal{F}, \mathcal{E})$ terms used here.

Meaning instability alone would be manageable under infinite resources, where the agent could try every candidate representation and keep whichever earns its cost. A resource-bounded agent cannot. It must commit to a small set of representations in advance, and the value of any candidate is $\mathcal{E}$ -dependent—what a representation makes cheap (which hypotheses, which queries, which compositions) is a property of how it behaves, not of how it is written. The agent can therefore discover whether a representation pays off only by deploying it, consuming the same finite budget the representation was meant to conserve. Resource rationality (Lieder and Griffiths, 2020) formalises this trade-off but takes $\mathcal{R}$ as given and optimises $\mathcal{F}$ within it. How $\mathcal{R}$ and $\mathcal{F}$ should jointly change to make future tasks cheaper is qualitatively harder, for reasons Section 2.3.2 makes precise.

Together these two lines of argument derive three structural consequences—construction, curation, and communication—as necessary features of any resource-bounded agent operating with use-dependent representations, developed in Section 2.3.3.

2.3.1 Pragmatism: why $\mathcal{R}$ is $\mathcal{E}$ -dependent

The instability identified above has a long philosophical lineage within the pragmatist tradition. Wittgenstein (1953) gives a *constitutive* account, arguing that a rule's meaning is fixed by the practices of its application rather than by its form. Dewey (1938) gives an *evaluative* account, on which an idea's worth is the work it does in solving problems. The first explains why $\mathcal{R}$ cannot be characterised independently of $\mathcal{E}$ and the second supplies the criterion by which any $\ell \in \mathcal{R}$ earns or loses its place. Together they pin down the form of a value function over $\mathcal{R}$ that the remainder of this chapter builds on.

Meaning is use. Consider someone who has used “+” only on small numbers: their behavior is consistent both with ordinary addition and with a deviant rule that agrees on all past cases but returns 5 above an unseen threshold (Kripke, 1991). No further rule resolves the gap, since each clarification permits the same deviant reinterpretation. Likewise, any finite dataset $D = (x_1, y_1), \dots, (x_n, y_n)$ is consistent with infinitely

15: The No Free Lunch theorems (Wolpert and Macready, 2002) already establish that no learning algorithm $\mathcal{F}$ universally dominates across task distributions. The present claim extends this to the *representational vocabulary* $\mathcal{R}$: not just the choice of algorithm but the choice of representational vocabulary is $\mathcal{E}$ -dependent, prior to any optimization over $\mathcal{F}$.

many hypotheses in a sufficiently expressive $\mathcal{H}$. The Bayesian response is to introduce a prior $p(h)$ favoring simpler hypotheses, but this shifts the problem to what fixes $\mathcal{H}$ and the relevant simplicity metric. Wittgenstein's answer is that these are stabilized not by further rules, which invite regress, but by practices: training, correction, and agreement about what counts as correct use. Thus, for any $\ell \in \mathcal{R}$, syntactic definition underdetermines operational meaning. Meaning is fixed by the history of ℓ 's deployment in $\mathcal{E}$: the contexts where it has been used, the regularities it has tracked, and the failures that exposed its limits.

Value is instrumental. Dewey (1938) approached the same instability from an instrumentalist direction. If meaning is constituted by use, then the relevant question about any ℓ is not whether it is true but whether it is useful for the tasks the agent faces. Representations, on this view, are tools, and usefulness is a property of the pairing $(\ell, \mathcal{E})$ rather than of ℓ alone—the same tool is effective for some distributions of tasks and misleading for others. Dewey's instrumentalism thus inherits Wittgenstein's $\mathcal{E}$ -dependence and turns it into a normative framing that ℓ earns its place by the work it does, not by its form or by correspondence to any task-independent reality.

A pragmatist value function. The two pragmatist premises already constrain the shape of any value function over $\mathcal{R}$. Wittgenstein requires the content of ℓ to be assessed through its use across a deployment distribution $\mathcal{E}$; Dewey requires that use to be evaluated instrumentally, by what it helps the agent do. Since constructing and retaining ℓ are themselves costly, the pragmatist value of ℓ is its expected savings across $\mathcal{E}$, net of upkeep:

$$\text{value}(\ell) = \underbrace{\mathbb{E}_{\mathcal{E}} [\Delta_{\text{cost}}(\ell)]}_{\text{amortized computational savings}} - \underbrace{C_{\text{maintain}}(\ell)}_{\text{storage, update, risk of misapplication}} \quad (2.3)$$

where $\Delta_{\text{cost}}(\ell)$ is the reduction in inference or planning cost that ℓ provides and $C_{\text{maintain}}(\ell)$ is the ongoing cost of keeping ℓ available and aligned with reality¹⁶. Both terms depend on $\mathcal{E}$: $\Delta_{\text{cost}}(\ell)$ reflects the savings ℓ provides across the tasks to which it is applied, while $C_{\text{maintain}}(\ell)$ reflects the cost of keeping ℓ aligned as tasks, background conditions, or standards of successful use change. Both are ultimately answerable to deployment, even if agents can estimate them indirectly from similar resources, prior experience, or structural cues. The pragmatist diagnosis therefore explains why representational management is difficult: if the content and usefulness of $\mathcal{R}$ are constituted through use in $\mathcal{E}$, then they cannot be read off from syntax alone or fixed once and for all independently of future use. The next subsection shows that resource rationality provides a framework for analyzing such trade-offs under bounded time, memory, and computation, but that many resource-rational models hold $\mathcal{R}$ fixed and so leave the deployment-dependence of representational resources only partially addressed.

16: Barsalou (1999)'s perceptual symbol theory converges on the similar criterion: concepts are *simulators* (i.e., learned capacities to re-enact task-relevant states), which are valuable because they can be run.

2.3.2 Resource rationality: why $(\mathcal{R}, \mathcal{F})$ cannot be fixed

The pragmatist value function in Equation (2.3) defines what a bounded agent should maximize over $\mathcal{R}$ but offers no procedure for doing so. Resource rationality (Lieder and Griffiths, 2020) is the standard framework for such trade-offs. Building on Simon's (Simon, 1955) satisficing, it takes as its starting point the fact that optimal solutions are typically

out of reach for bounded agents, and recasts the rational choice as the *approximation* that best trades solution quality against its computational cost. Therefore, the resource-rational agent selects the algorithm $\pi^* \in \Pi$ that maximizes the trade-off between solution quality and cost:

$$\pi^* = \arg \max_{\pi \in \Pi} \mathbb{E}[U(f(x), \pi(x))] - \lambda \cdot C(\pi). \quad (2.4)$$

Here f is the function the agent aims to compute, U measures the quality of the approximation $\pi(x)$ relative to $f(x)$, and $C(\pi)$ is the computational cost of running π ¹⁷. Systematic deviations from Bayesian or expected-utility optimality (e.g., anchoring, probability matching) can be reframed as rational compromises. Anchoring emerges when only a small number of MCMC samples are affordable, while probability matching becomes optimal when identifying the best action is too expensive relative to the benefit (Griffiths et al., 2015; Lieder and Griffiths, 2020)¹⁸.

$\mathcal{R}$ and $\mathcal{F}$ are coupled, but resource rationality optimizes only $\mathcal{F}$. The optimization in Equation (2.4) ranges over algorithms $\pi \in \Pi$, but the set of available algorithms Π is itself determined by the representational format $\mathcal{R}$: what operations are cheap, what data structures are available, what hypotheses can be concisely expressed. Conversely, changing $\mathcal{R}$ —building a new abstraction, adding a subroutine—is itself a computation that $\mathcal{F}$ must perform using the current $\mathcal{R}$. Resource rationality takes $\mathcal{R}$ as given and optimizes $\mathcal{F}$ within it, which answers *given my representations, what should I compute?* but not the complementary question: *how should $\mathcal{R}$ and $\mathcal{F}$ jointly change to make future tasks cheaper?*

The achievable performance within a fixed $\mathcal{R}$ is bounded above by what $\mathcal{R}$ makes expressible and what it makes tractable. The classical expressivity illustration is Minsky et al. (1974). No training procedure operating over linear representations of the inputs can learn parity, because parity is not in the function class those representations can express. Adding a hidden layer and thus a richer $\mathcal{R}$ resolves the problem, not by finding a better algorithm but by changing what the algorithm is searching over (Rumelhart et al., 1986). The same pattern holds for tractability, where reformulating a planning problem’s state space can collapse exponential search into nearly trivial search with no change to the search procedure (Amarel, 1968). Changing $\mathcal{F}$ moves the agent within the ceiling $\mathcal{R}$ sets. Changing $\mathcal{R}$ moves the ceiling itself.

Why reshaping $(\mathcal{R}, \mathcal{F})$ is qualitatively harder. Resource rationality typically optimizes $\mathcal{F}$ within a fixed $\mathcal{R}$, but representational dynamics poses a different problem. For a lifetime agent, the central allocation decision is not which algorithm to use on the current task, but how to reshape $(\mathcal{R}, \mathcal{F})$ itself: which representations to build, which to retire, and how to adapt the computational repertoire as $\mathcal{E}$ changes. This meta-level problem is harder than algorithm selection for three reasons.

First, representational investments have *non-local payoff*. The payoff of ℓ is distributed across every future task that invokes ℓ , not concentrated on the task that prompted its construction. This is the logic of amortized analysis applied to representations. If ℓ is invoked on n future tasks, its construction cost amortizes to $C_{\text{construct}}(\ell)/n$, and its net value depends on the distribution of those tasks¹⁹. Evaluating ℓ therefore requires estimating a future task distribution the agent does not yet have.

Second, representational value is *uncomputable*. Even if the future task distribution were known, the value of ℓ could not be computed from ℓ ’s syntax. Usefulness depends on which computations ℓ makes cheap and

17: The framework subsumes several notions of “cost”: number of samples drawn, working-memory load, time, or any scarce resource consumed by computation.

18: Equation (2.4) and Equation (2.3) are the same trade-off at different levels of description: Equation (2.3) is the object-level case where ℓ is fixed and the question is whether to retain it; Equation (2.4) is the algorithm-selection problem given that $\mathcal{R}$ is already determined.

Example 2.1 (Representation dominates algorithm.) Roman numerals and positional notation represent the same integers, but long multiplication requires $O(n^2)$ symbol manipulations in Roman numerals and $O(n \log n)$ in positional notation. No choice of algorithm within the Roman numeral system closes this gap; the performance ceiling is set by $\mathcal{R}$, not $\mathcal{F}$.

19: In amortized analysis, an expensive operation is justified by savings across subsequent cheap ones; a representational abstraction follows the same logic.

which hypotheses ℓ makes expressible, and these are semantic properties of ℓ as a program. By Rice's theorem (Rice, 1953), every non-trivial semantic property of programs is undecidable²⁰. The agent therefore has no substitute for running ℓ and observing what it saves. Analysis cannot replace use, in principle and not merely in practice.

Third, the optimization is *self-referential*. Any improvement to $(\mathcal{R}, \mathcal{F})$ is itself a computation, and it must be carried out by the current $\mathcal{F}$ operating over the current $\mathcal{R}$. The machinery that proposes, evaluates, and installs a new abstraction is the same machinery the new abstraction is meant to alter. Self-improvement is nevertheless possible, since Turing's universal machine shows that programs can treat other programs as data, but two constraints follow. Each step must be small enough for the current vocabulary to execute, and each successful step reshapes the vocabulary that executes the next. The learning trajectory is therefore path-dependent: the same agent encountering the same tasks in a different order arrives at a different $(\mathcal{R}, \mathcal{F})$ ²¹.

These three features, non-locality, undecidability, and self-reference, are why representational dynamics cannot be captured by any optimization that holds $\mathcal{R}$ fixed, including the resource-rational one in Equation (2.4). They demand an ongoing, empirical, path-dependent process whose stages are derived, after one further line of argument, in Section 2.3.3.

2.3.3 Three structural consequences

From the pragmatist tradition: the meaning of any $\ell \in \mathcal{R}$ is constituted by its deployment history in $\mathcal{E}$, and its value is the work it does across $\mathcal{E}$. From resource rationality: a bounded agent must trade solution quality against computational cost under a fixed set of resources. Combining these gives the value function in Equation (2.3) and three properties of it that neither tradition made explicit. Its payoff is non-locally distributed across future tasks the agent has not yet seen. Its magnitude cannot be read off ℓ 's syntax and is undecidable. And any search for a better ℓ is performed by the current $(\mathcal{R}, \mathcal{F})$, making improvement path-dependent. Three structural consequences follow for any resource-bounded agent operating with use-dependent representations.

Construction is path-dependent. When an agent builds reusable structure, early solutions reshape the landscape for later learning. Once a primitive ℓ is available, it becomes a cheap building block, favoring hypotheses expressible in terms of ℓ ²². The order of experience is therefore a structural determinant of the eventual $\mathcal{R}$: different curricula over the same tasks yield different libraries, different hypothesis spaces $\mathcal{H}$, and different patterns of generalization (Bengio et al., 2009; Dekker et al., 2022; Elman, 1993). Path dependence is an inevitable consequence of incremental construction—an agent that reuses its current library when forming new abstractions will arrive at different representational states depending on what it encountered and when. Part II develops computational and empirical accounts of how representational structure is built incrementally under resource constraints and how the resulting library shapes subsequent learning.

Curation balances maintenance against obsolescence. A library calibrated to one task distribution $\mathcal{E}$ may become a liability when $\mathcal{E}$ changes. Primitives that once compressed useful structure can later bias inference toward hypotheses that no longer apply, a representational analogue of catastrophic interference (McCloskey and Cohen, 1989). The cost is

20: The halting problem is the most familiar instance; Rice's result generalizes it to every property that depends on what a program *does* rather than what it *looks like*.

Remark 2.2 (Rice's theorem (Rice, 1953).) Let ϕ_e denote the partial function computed by program e , and let P be any non-trivial set of partial functions (neither empty nor universal). Then $\{e : \phi_e \in P\}$ is undecidable. Usefulness is such a P : it depends on what ℓ computes across $\mathcal{E}$, not on its syntactic form.

21: Carey (2000) documents this in conceptual development: children's core primitives (object, agency, number) are the substrate from which all later abstractions are built, making them disproportionately influential.

22: Formally, under a description-length prior (Grünwald, 2007; Rissanen, 1986), a primitive that compresses a recurring pattern p shortens every hypothesis invoking p and thereby raises its posterior on future tasks.

therefore not merely storage: outdated entries can distort priors, slow search, or crowd out more useful abstractions. Maintaining $\mathcal{R}$ requires evaluating each ℓ against Equation (2.3): retaining it when its expected savings justify its maintenance cost, and revising or pruning it otherwise. Intrinsic motivation can be read in this light. Curiosity drives agents toward resources whose expected future value may exceed their construction cost (Oudeyer and Kaplan, 2007), while empowerment—an agent’s information-theoretic control over its future states (Klyubin et al., 2005)—offers one criterion for retaining representations that preserve flexible action. Part III develops this reinterpretation by connecting intrinsic-motivation signals to the curation stage of representational management.

Communication requires representational alignment. If $\mathcal{R}$ is $\mathcal{E}$ -dependent, agents with different histories arrive at genuinely different representational vocabularies (e.g., different defaults, different priors over $\mathcal{H}$). Transmitting knowledge across such agents requires more than propositional exchange; it requires aligning representational practices and establishing enough common ground (Clark and Brennan, 1991) for mutual inference to succeed. Teaching is the most deliberate form of this alignment: an effective teacher reasons about the gap between the learner’s current $\mathcal{R}$ and the one needed for competence, then designs an intervention—an explanation, an example, a curriculum—shaped to close that gap. Part IV formalizes this as resource-rational communication aimed at representational alignment.

One lifecycle. The three stages form a directed cycle. Construction determines what is available to curate—the library an agent has built determines which entries can become obsolete. Curation determines what is available to communicate—an agent that has pruned a once-useful abstraction can no longer teach it. Communication shapes what others construct—a learner who receives a well-chosen explanation inherits structure that reshapes $\mathcal{H}$ for future learning. Together, these stages form a self-modifying loop. Understanding how representational vocabulary is constructed, maintained, and shared under resource constraints is the central aim of the thesis.

2.4 The lifecycle of representational vocabulary

The preceding sections identified the representational dynamics gap (Section 2.2), explained why it persists (Section 2.3), and derived three structural consequences—construction, curation, communication—that any bounded agent operating with use-dependent representations must satisfy (Section 2.3.3). These consequences are stated at the level of the abstract triplet $(\mathcal{R}, \mathcal{F}, \mathcal{E})$. To study them empirically, we need a concrete commitment to what $\mathcal{R}$ and $\mathcal{F}$ are made of.

A useful example is DREAMCODER (Ellis et al., 2021). Its representational vocabulary $\mathcal{R}$ is a library of primitives defined in a domain-specific language (DSL), extended during a sleep phase in which recurring subroutines abstracted from recent solutions are added as new primitives (Hinton et al., 1995). A Bayesian description-length prior scores which abstractions are kept, while a neural recognition network, trained alongside, proposes programs more efficiently on future tasks. Once a new subroutine is stored, it shortens every program that invokes it, shifts the prior over hypotheses, and changes what the learner can

cheaply express next. For the first time in this chapter, the feedback path $\pi \xrightarrow{\text{abstract}} (\mathcal{R}', \mathcal{F}')$ is made explicit.

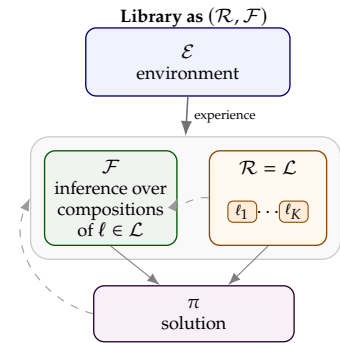
Read through the pragmatist lens of [Section 2.3.1](#), however, DREAMCODER's library is primarily cumulative rather than actively curated. DREAMCODER scores a primitive by how much it compresses prior solutions, but not by the instrumental payoff it provides across the future distribution of tasks the agent will face. After scoring, useful primitives are added to the library as learned resources; what remains less developed is an account of how such resources should later be revised, retired, or reorganized when their value changes. As a consequence, DREAMCODER scales the possibility of *construction* but leaves *curation* untheorized and does not attempt *communication*. The object this thesis proposes to study is the *lifecycle of libraries*, which treats the library $\mathcal{L}$ and its dynamics as the primary theoretical objects and recovers DREAMCODER's DSL as one instance. A library, defined by its entries and the possible computations over the entries, is continually reshaped as the agent encounters new tasks, as old abstractions stop earning their cost, and as other agents' libraries come into contact with its own. This shift from endpoint to lifecycle makes the three consequences of [Section 2.3.3](#) operational. Construction becomes the process by which entries enter the library, curation the process by which they are retained or pruned as $\mathcal{E}$ shifts, and communication the process by which entries are exchanged between agents.

2.4.1 The generative library as a target of analysis

A *library* $\mathcal{L} \in \mathbb{L}$ is a finite collection $\mathcal{L} = \{\ell_1, \ell_2, \dots, \ell_K\}$ of structured representations, sometimes we name it as *generative library* when we expect each entry is a generative model. Each entry $\ell \in \mathcal{L}$ satisfies a dual requirement. It is *data*, since it can be stored, inspected, compared with other entries, and transmitted to another agent. And it is a *computational tool*, since it can be applied to tasks, composed with other entries to form complex hypotheses, and evaluated through use.

The library as an instantiation of $(\mathcal{R}, \mathcal{F})$. The learning-system triplet $(\mathcal{R}, \mathcal{F}, \mathcal{E})$ in [Definition 2.1](#) leaves the representational format and the computation as abstract slots, compatible with any concrete choice. This thesis commits to a particular choice. For the purposes of modeling human learning, we take $\mathcal{R}$ to be a finite collection $\mathcal{L}$ of individually addressable, inspectable, deployable entries, and $\mathcal{F}$ to be inference that proceeds by composing those entries. The commitment is not that cognition must take this form. It is that this form is the one in which the three consequences of [Section 2.3.3](#) become concrete operations, and the one in which the lifecycle we want to study is visible.

The library concretizes the change of $(\mathcal{R}, \mathcal{F})$. Under this commitment, $\mathcal{L}$ and $(\mathcal{R}, \mathcal{F})$ are tied together directly²³. $\mathcal{R}$ is the set of well-formed compositions of entries in $\mathcal{L}$, and the description length $|h|_{\mathcal{L}}$ of any hypothesis h depends on which entries are available to form it. $\mathcal{F}$ is search or inference over those compositions, and its cost on any given hypothesis scales with the number of primitives that hypothesis invokes. An entry that compresses a recurring procedure into a single invocation therefore makes every hypothesis using it both shorter to describe and faster to find. Adding a new entry ℓ_{K+1} therefore modifies both $\mathcal{R}$ and $\mathcal{F}$ at once. The hypotheses prior shifts because what counts as “short” has changed, and every hypothesis that invokes ℓ_{K+1} becomes cheaper to both describe and find. This is the abstraction step $\pi \xrightarrow{\text{abstract}} (\mathcal{R}', \mathcal{F}')$



23: Suppose $\mathcal{L}$ contains a sort primitive. Under $\mathcal{R}$, the hypothesis “sort the list, then take the first element” is a two-primitive composition with a short description, and a description-length prior accordingly assigns it high probability. Under $\mathcal{F}$, a learner that searches by composing primitives can reach this hypothesis in two proposal steps. Remove sort from $\mathcal{L}$ and both change. The hypothesis must now be expressed as a full compare-and-swap routine, which lengthens its description and lowers its prior, and the learner must discover that longer routine by search, expanding the number of proposals required to find it.

in Equation (2.1). Whereas the traditions in Section 2.2 leave this arrow unspecified, here it acts on the library: incorporating a candidate entry into $\mathcal{L}$ makes the arrow concrete.

Relation to prior constructs. The commitment that cognition depends on a repertoire of reusable blocks is not new. Minsky’s *frames* (Minsky et al., 1974) and Schank and Abelson’s *scripts* (Schank and Abelson, 2013) formalized this as structured records with slots for recurring situations such as visiting a restaurant or recognizing a room. Anderson’s ACT-R *production rules* (Anderson, 1996) encode condition-action pairs that abstract procedural skill, and Chase and Simon’s *chunks* (Chase and Simon, 1973) account for the expert’s ability to compress familiar board positions into retrievable units. Tolman’s *cognitive maps* (Tolman, 1948) extended the commitment to spatial domains, positing internal representations of the environment that guide flexible behaviour. Each construct is a reusable representational building block that compresses recurring structure and supports use across contexts, and each has been influential precisely because it isolates a unit of representation that can be reasoned about on its own. What is new in the present framework is neither the entries nor their individual function, but the treatment of the collection $\mathcal{L}$ and its management over time as the central theoretical object—how entries enter the library, how they are retained or retired as the task distribution shifts, and how they travel between agents.

2.4.2 What the library framework makes tractable

The three consequences of Section 2.3.3, when concretized onto a named collection of discrete entries, become well-defined operations amenable to formal analysis.

Construction becomes a *compression problem*. The agent encounters solutions to tasks drawn from $\mathcal{E}$, identifies recurring structure across them, and abstracts that structure into a new entry ℓ_{k+1} . The criterion for a good abstraction is its effect on description length: an ℓ that compresses a recurring pattern p makes every hypothesis invoking p shorter under $\mathcal{L}$, increasing its prior probability. The order in which tasks are encountered determines which patterns get compressed first, making construction path-dependent in the precise sense established in Section 2.3.3. Part II develops this across two instantiations, programs in a compositional grammar (Chapter 4) and knowledge components in a prerequisite graph (Chapter 5).

Curation becomes an *action-decision problem*. As $\mathcal{E}$ shifts, each $\ell \in \mathcal{L}$ must be evaluated against Equation (2.3): does its amortized computational savings still exceed its maintenance cost? Entries whose value has dropped below their cost should be pruned or refactored, and whose scope has changed should be updated. The evaluation is inherently empirical and must be driven by signals that track ℓ ’s ongoing utility, or the agent’s internal model, which is the role that intrinsic motivation plays in Part III, through curiosity (Chapter 6) and empowerment (Chapter 7).

Communication becomes an *alignment problem*. Two agents with libraries $\mathcal{L}_A$ and $\mathcal{L}_B$, built through different histories in $\mathcal{E}$, will carve the domain at different joints and find different problems solvable. Coordination requires identifying which entries must be shared, adapting them to the receiving agent’s existing library, and establishing enough common ground for mutual inference (Clark and Brennan, 1991). The question is then whether an ℓ from one agent’s library can be absorbed

Table 2.2: Library entries across chapters. Each chapter instantiates ℓ as a different kind of structured representational resource. The lifecycle stage indicates which aspect of library management the chapter primarily studies.

Chapter	Stage	Entry type ℓ	Domain	Example entry
Chapter 4	Construction	Programs in a compositional grammar	Melody reproduction	A chunking rule that groups a repeated note sequence into a single callable motif
Chapter 5	Construction	Knowledge components in a prerequisite graph	Mathematics problems	A prerequisite link specifying that <i>addition</i> must be mastered before <i>multiplication</i>
Chapter 6	Curation	World-model components (transition models)	Single grid game	A learned transition model distinguishing deterministic walls from stochastic ice cells
Chapter 7	Curation	World-model components (state representations)	Sequential grid games	A state representation that tracks controllable degrees of freedom
Chapter 8	Communication	World model + value representations	Navigation explanation	A partial map and associated costs communicated to a listener to enable route planning

into another’s—a representational analogue of the alignment problem that the Rational Speech Act framework (Frank and Goodman, 2012; Goodman and Frank, 2016) addresses at the utterance level. Part IV takes this up in the context of explaining (Chapter 8), where the sender designs an intervention against a model of the receiver’s current library.

2.5 Summary

This chapter has argued, in three steps, that a particular object is the right target for studying how representations and the computations over them co-evolve under resource constraints. Section 2.2 located the representational dynamics gap empirically by tracking one deferral— $s \xrightarrow{\text{compile}} (\mathcal{R}', \mathcal{F}')$ —across symbolic, connectionist, and Bayesian cognitive science, showing that the gap is not an oversight of any one programme but a structural feature each tradition reaches and does not close. Section 2.3 derived the gap from first principles, using pragmatism to establish that the value of a representation is constituted by its deployment in $\mathcal{E}$ and using resource rationality to establish that such value is non-local, undecidable, and accessible only through self-referential search. Three structural consequences followed: *construction*, *curation*, and *communication*. Section 2.4 then committed to an object—the lifecycle of a generative library $\mathcal{L}$ —on which those three consequences become concrete operations. Construction is the abstraction of recurring structure into new entries. Curation is the evaluation of existing entries against shifting demands. Communication is the alignment of one agent’s library with another’s.

The three consequences are commitments about *what* the thesis studies, not methods for studying it. What this chapter has not done is specify how any of these operations is actually carried out. Chapter 3 introduces the computational tools the thesis uses to turn these commitments into tractable models, namely probabilistic program induction for construction, reinforcement learning with intrinsic rewards for curation, and Bayesian pedagogy under pragmatic inference for communication. Each tool is already standard in its home area; what is new is the argument, assembled across this chapter, that these three tools are the right ones because they correspond to the three operations any bounded agent with use-dependent representations must perform.

Formalism: from latent variables to scalable libraries

3.

The preceding chapter proposed generative libraries as the study object and derived three stages of their lifecycle (construction, curation, communication) from the coupled dynamics of representations and computation ($\mathcal{R}, \mathcal{F}$) under resource constraints. After establishing *what* to study, this chapter focuses on *how* to study it, providing the formal tools used throughout the thesis.

The common logic across these tools is latent inference over generative models of agents. From observed behavior (i.e., choices, trajectories, and learning curves), we posit hidden structure (i.e., knowledge states, programs, world models, and communicative intent), which allow us to recover similar behavior. What varies across the thesis is the form of the hidden variables and the machinery appropriate to each. The chapter assembles these tools in sequence, from the simplest latent-variable models through sequential decision-making, program induction, multi-agent inference, and compression, showing that each addresses a specific aspect of the lifecycle of a library.

3.1 Generative models and latent inference	27
3.2 Agent-centric learning . . .	29
3.3 Programs as structured latent representation . . .	36
3.4 World models as learned latents	40
3.5 Other agents as latent systems	44
3.6 Compression as the evaluative principle	48
3.7 Summary	52

3.1 Generative models and latent inference

Computational cognitive science begins from a basic observational asymmetry. We observe choices, judgments, trajectories, or utterances, but we do not directly observe the internal structures that make those behaviors possible. Methodologically, this can be solved by inverting the problem: positing hidden variables and then evaluating them based on how well they explain overt behavior. This section introduces that inferential stance and uses it to define the formal starting point of the thesis. The central claim is that agent¹ models can be treated, at first pass, as *latent-variable generative models*. What changes across modeling traditions is which hidden variables are taken to be explanatory, how richly they are structured, and how they evolve over time.

1: I use “agent” as the general modeling term for any system whose behavior is explained by latent structure. In the thesis’s empirical context, the primary target is human cognition, but the broader term is useful because the same formal tools also apply to idealized and artificial agents.

3.1.1 Latent-variable models of agents

The inverse problem identified above—inferring hidden cognitive structure from observable behavior—has a standard Bayesian formulation (L Griffiths et al., 2008; Tenenbaum et al., 2011). Let $\mathbf{x}_{1:T} = (x_1, \dots, x_T)$ denote behavior over T time steps (e.g. motor sequences, reaction times). A generative model introduces latent variables $\mathbf{z}$ that give rise to these observations:

$$\begin{aligned} \mathbf{z} &\sim p(\mathbf{z}), \\ \mathbf{x}_{1:T} &\sim p(\mathbf{x}_{1:T} \mid \mathbf{z}). \end{aligned} \tag{3.1}$$

Specifying such a model requires choosing both an *ontology of the latent $\mathbf{z}$* —the hidden object organizing behavior, such as a trait parameter or belief state—and a *generative process* $p(\mathbf{x}_{1:T} \mid \mathbf{z})$ linking it to data. These choices

2: A discriminative model learns $p(y | x)$ directly and can predict well, but cannot answer “what would happen if x were different?” because it has no model of how x is generated. A generative model commits to $p(x, z)$, which supports counterfactual queries by intervening on z .

set the model’s explanatory scope: scalar latents capture individual differences, whereas structured latents capture dependencies, abstractions, and compositional reuse, albeit at greater computational cost.

The value of committing to a generative model is that it supports derived inferences by specifying hidden causal structure that can be reused across tasks², such as prediction (Gershman, 2019), counterfactual reasoning (Byrne, 2016; Epstude and Roese, 2008), diagnosis (Baker et al., 2009), and explanation (Lombrozo, 2006). Given observations, inference inverts the model via Bayes’ rule,

$$p(z | x_{1:T}) = \frac{p(x_{1:T} | z) p(z)}{p(x_{1:T})} \quad \text{where} \quad p(x_{1:T}) = \int p(x_{1:T} | z) p(z) dz. \quad (3.2)$$

The prior $p(z)$ encodes which latent structures are plausible before data, while the likelihood $p(x_{1:T} | z)$ specifies their behavioral consequences. Because explanations depend on the choice of z , the same data can yield different posteriors under different representational vocabularies. This inferential stance recurs throughout the thesis, where z may denote knowledge states, prerequisite graphs, learned subroutines, or communicative intent.

So far, however, z is static: a fixed hidden cause generating the full observation sequence. Yet cognitive behavior unfolds over time, and the internal states producing it change as agents learn, adapt, or accumulate evidence. The next step is therefore to give z its own dynamics.

3.1.2 Passive dynamics: state-space models

3: State-space models originated in control theory (Kalman, 1960) for tracking physical systems (spacecraft trajectories, missile guidance). Their adoption in cognitive science applies the same mathematics to track internal states from behavioral observations.

Example 3.1 (Knowledge tracing as an SSM.) In knowledge tracing (Corbett and Anderson, 1994a), mastery of skill k is a hidden binary state $z_t^{(k)} \in \{0, 1\}$. Transition (absorbing):

$$\begin{aligned} p(z_{t+1}^{(k)} = 1 | z_t^{(k)} = 0) &= p_L \\ p(z_{t+1}^{(k)} = 1 | z_t^{(k)} = 1) &= 1 \end{aligned}$$

Emission (noisy):

$$\begin{aligned} p(x_t = \text{correct} | z_t^{(k)} = 1) &= 1 - p_S \\ p(x_t = \text{correct} | z_t^{(k)} = 0) &= p_G \end{aligned}$$

where p_L is the learn rate, p_S the slip rate, and p_G the guess rate. Filtering recovers $p(z_t^{(k)} = 1 | x_{1:t})$, estimating mastery after each practice trial.

The simplest extension gives z temporal structure while bracketing out actions: the agent’s internal state evolves over time, but the agent does not intervene on its own observation stream³. Examples include perceptual inference, passive sequence learning (Saffran et al., 1996), hidden-state tracking, and trial-by-trial belief updating (Daw, 2011).

A standard formalization is the state-space model,

$$\begin{aligned} z_1 &\sim p(z_1), & \text{(initial latent state)} \\ z_t &\sim p(z_t | z_{t-1}), & \text{(transition dynamics)} \\ x_t &\sim p(x_t | z_t). & \text{(emission / observation)} \end{aligned} \quad (3.3)$$

The latent state evolves through internal dynamics, with each observation generated from the current state. By the Markov property, z_t screens off past from future: any dependence of $x_{t+1:T}$ on $x_{1:t-1}$ is mediated by the current hidden state. Hidden Markov models, Kalman filters, and many trial-level cognitive process models instantiate this form (Smith et al., 2004). This formalization turns changing internal states into targets of inference. The transition model $p(z_t | z_{t-1})$ specifies *how* states change, the emission model $p(x_t | z_t)$ specifies their behavioral consequences, and the posterior $p(z_{1:T} | x_{1:T})$ recovers the latent trajectory from data.

However, passive state-space models are insufficient for agents that act on their environments. In Equation (3.3), observations are exogenous: the agent does not choose what to observe next. Such models capture how hidden states evolve under evidence, but not how actions shape which evidence arrives. When agents select tasks, explore, or communicate

strategically, the next observation depends on the current action. Modeling this coupling requires a formalism in which action, observation, and latent state evolve together, as introduced in [Section 3.2](#).

3.2 Agent-centric learning

The previous section argued that once agents act, latent inference and sequential interaction become inseparable. This section develops reinforcement learning (RL; Botvinick et al., 2019; Dayan and Daw, 2008; Sutton, Barto, et al., 1998) as the formal setting in which that coupling can be stated precisely. Our aim is to show how RL supplies a disciplined language for action under uncertainty, and how that language can be reinterpreted in agent-centric terms so that latent internal structure becomes the main object of explanation.

3.2.1 From exogenous observations to active interaction

Once an agent acts, future observations are no longer purely exogenous—the distribution of x_{t+1} depends on the action a_t taken at time t . The explanatory question shifts from “what hidden state generated this observation sequence?” to “what internal organization generated this trajectory of actions, observations, and feedback?”

The state-space model extends naturally to accommodate action. The environment state s_t plays the role of the SSM’s latent z_t , but its dynamics now depend on the agent’s choice a_t ,

$$\begin{aligned} a_t &\sim \pi(a_t \mid h_t), && \text{(policy)} \\ s_{t+1} &\sim T(s_{t+1} \mid s_t, a_t), && \text{(transition)} \\ o_{t+1} &\sim \Omega(o_{t+1} \mid s_{t+1}), && \text{(observation)} \\ r_t &= R(s_t, a_t). && \text{(reward)} \end{aligned} \tag{3.4}$$

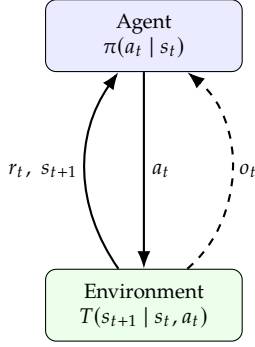
Here s_t is the environment state (often unobserved), o_t is the observation the agent receives, a_t is the action it selects, r_t is evaluative feedback, and $h_t = (o_1, a_1, \dots, o_t)$ is the agent’s history. When s_t is fully observed, this reduces to a Markov decision process (MDP); when it is hidden, the agent must maintain a belief over s_t from its history—a partially observable MDP (POMDP), which is itself a state-space model whose latent state is the agent’s belief⁴. The POMDP generalizes the state-space model of [Section 3.1.2](#) in two directions at once. The transition kernel T now depends on the agent’s action, closing the loop between inference and intervention; and an evaluative signal r_t is added, making the latent dynamics goal-directed rather than passive.

RL as closed-loop latent inference. Formally, sequential decision-making under partial observability is the POMDP,

$$\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{O}, T, \Omega, R, \gamma),$$

where $\mathcal{S}$ is the state space of the external environment, $\mathcal{A}$ the action space, $\mathcal{O}$ the observation space, $T(s' \mid s, a)$ the transition kernel, $\Omega(o \mid s)$ the observation model, $R(s, a)$ the reward function, and $\gamma \in [0, 1)$ a discount factor. At each time step the environment occupies a state $s_t \in \mathcal{S}$. The agent receives observation $o_t \sim \Omega(\cdot \mid s_t)$, selects action a_t , receives

4: The POMDP is the natural fusion of the SSM and decision-making. The SSM’s latent z_t becomes the environment state s_t ; the SSM’s emission becomes the observation function Ω ; and the transition now depends on the agent’s action a_t , closing the loop between inference and intervention.



5: Given the task model, the full history need not be retained explicitly once it has been compressed into the posterior b_t . Under exact Bayesian control, the belief state is a sufficient statistic for action.

Remark 3.1 (Predictive observation term.) The observation probability marginalizes over both current state uncertainty and action-conditioned dynamics,

$$p(o | b, a) = \sum_{s'} \Omega(o | s') \sum_s T(s' | s, a) b(s).$$

It is the probability that action a will lead to observation o when the agent's current belief over world states is b .

6: Useful actions need not produce immediate reward. In a dark room with two doors, opening one is a direct bid for the reward; turning on the light is not yet rewarding, but it is no less instrumental, because the information it provides is what lets the next action pay off.

reward $r_t = R(s_t, a_t)$, and the environment transitions to $s_{t+1} \sim T(\cdot | s_t, a_t)$.

The fully observable MDP is the special case in which each observation fully identifies the state, such that Ω is injective. For cognitive modelling, the POMDP is the more natural starting point because it makes explicit the distinction between what the world objectively is and what the agent can know about it. Here, the agent does not act with full knowledge of the true state, but relies on inferred states of the world constructed from its history.

Belief states and the value of information. Because the agent does not observe s_t directly, its information at time t is summarized by the *belief state*, which is a posterior distribution over environment states given the history $h_t = (o_{1:t}, a_{1:t-1})$,

$$b_t(s) \triangleq p(s_t = s | h_t). \quad (3.5)$$

After taking action a_t and receiving observation o_{t+1} , the belief updates via Bayes' rule⁵:

$$b_{t+1}(s') \propto \Omega(o_{t+1} | s') \sum_{s \in \mathcal{S}} T(s' | s, a_t) b_t(s). \quad (3.6)$$

This update is the active counterpart of the Bayesian inversion introduced in Section 3.1. At each step, the agent infers hidden world state from evidence whose distribution depends partly on its own previous intervention.

The belief is the first latent object in this chapter whose dynamics depend on the agent's own action. It is the active counterpart to the passive SSM of Section 3.1.2: where the SSM inferred a hidden state from evidence, the belief state infers a hidden state from evidence whose distribution the agent itself partly determines.

Any POMDP can be reformulated as a fully observed MDP over the belief simplex $\Delta(\mathcal{S})$, the *belief MDP*, where the state is the agent's posterior b_t and transitions follow the Bayesian update in Equation (3.6) (Kaelbling et al., 1998; Smallwood and Sondik, 1973). Thus, the hidden environment state is replaced by the agent's inferred belief state, so optimal action depends on the agent's current estimate rather than the true state.

Bellman equation in belief space. Given a policy π that maps belief states to actions, the value of a belief is the expected discounted return,

$$V^\pi(b) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t r_t \mid b_0 = b \right]. \quad (3.7)$$

The optimal value function satisfies⁶

$$V^*(b) = \max_{a \in \mathcal{A}} \left[\bar{R}(b, a) + \gamma \sum_{o \in \mathcal{O}} p(o | b, a) V^*(\tau(b, a, o)) \right], \quad (3.8)$$

where $\bar{R}(b, a) = \sum_s b(s) R(s, a)$ is the expected immediate reward under the current belief, $p(o | b, a)$ is the probability of observing o after acting, and $\tau(b, a, o)$ denotes the updated belief. An action can matter twice. It can alter the world through T , and it can alter the agent's future knowledge by changing which observations become available. Exploration is therefore not an add-on to the control problem. It is built

into the Bellman recursion itself because information has downstream decision value.

Aligned with the central topic of this thesis, RL converts the inverse problem from inferring hidden causes of observations to inferring hidden causes of actions, observations, and learning trajectories taken together.

3.2.2 Why standard RL is not yet a complete cognitive theory

RL is necessary for this chapter because the agent helps generate its own future data (Dayan and Daw, 2008). An action is thus an intervention on the world that changes what the agent can subsequently observe, predict, and learn. Planning (Sutton, 1991), delayed consequences, adaptive control (Åström, 1995), and strategic exploration (Auer et al., 2002) all require this closed-loop perspective, and the POMDP provides the minimal structure in which they can be stated precisely.

However standard RL is not sufficient on its own, because it is primarily *environment-centric* Figure 3.1d (Abel et al., 2024)⁷. The reward function R is specified externally, the state space $\mathcal{S}$ is fixed by the problem designer, and the transition dynamics T are treated as task structure to be known or estimated instrumentally. The agent’s internal computation is reduced to policy and value functions sufficient for control, but largely silent about the representational vocabulary enabling it.

An agent-centric analysis makes this shift explicit by separating what belongs to the task from what belongs to the agent. The task supplies the external structure: observations, actions, transitions, and rewards,

$$\mathcal{E} = (\mathcal{S}, T, \Omega, R), \quad (3.9)$$

which exists independently of any particular learner. The agent supplies an internal organization that mediates between them,

$$\begin{aligned} z_t &= \phi(h_t), & (\text{representation}) \\ \hat{z}_{t+1} &\sim m(z_{t+1} \mid z_t, a_t), & (\text{construal of dynamics}) \\ a_t &\sim \pi(a_t \mid z_t), & (\text{policy}) \\ \hat{V}(z_t) &= \mathbb{E}_\pi \left[\sum_{\tau \geq t} \gamma^{\tau-t} r_\tau \mid z_t \right]. & (\text{subjective value}) \end{aligned} \quad (3.10)$$

These objects live in the agent’s representational space instead of the environment’s (Figure 3.1e). The encoder ϕ compresses history into a representation that is generally non-Markovian with respect to reward; m is the agent’s construal of dynamics rather than the true transition kernel; and $\hat{V}$ is the value the agent ascribes to its own representations under its own policy and construal, not the optimal value of the underlying task.

This separation sharpens the inferential problem. The same external task can produce qualitatively different behavior under different internal organizations, and the same observed behavior can mask different ones⁸. RL supplies the formal syntax of sequential interaction—states, actions, and rewards—but leaves open which internal components explain observed behavior. The next subsection develops this question.

7: Abel et al. (2024) identify three dogmas of RL: (1) the *reward hypothesis*—all goals can be cast as reward maximization; (2) the *environment spotlight*—the environment is the primary object of modeling, not the agent; (3) the *single-lifetime assumption*—learning happens within one fixed task. Each has drawn objections on its own terms; I would add that each also deflects attention from the agent’s internal representational structure.

8: This underdetermination is the core inferential problem for cognitive science. Given observed behavior, what internal organization $(\phi, m, \pi, \hat{V})$ must the agent have in order to generate the pattern of action, exploration, and generalization that we observe?

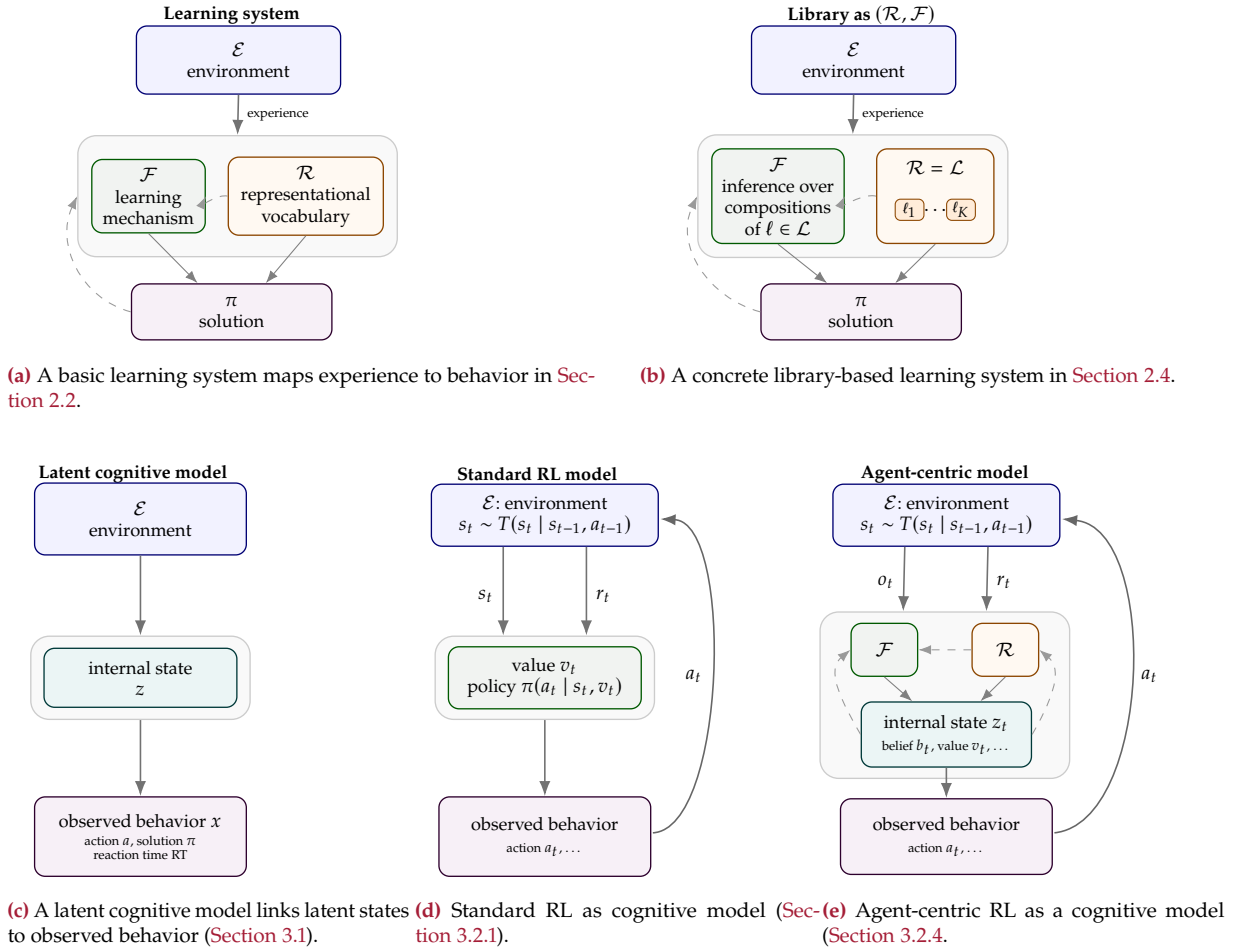


Figure 3.1: Five views of the agent-environment loop. (a) A generic learning system: a mechanism $\mathcal{F}$ acting over a representational vocabulary $\mathcal{R}$ produces a solution π from experience. (b) A library-based instance, where $\mathcal{R} = \mathcal{L}$ is a set of primitives and $\mathcal{F}$ is inference over their compositions. (c) A latent cognitive model: internal state z mediates between environment and observed behavior. (d) Standard RL collapses internal state into the environment state s_t . (e) Agent-centric RL reinstates an internal state z_t —belief, value, and other latent components—built by $\mathcal{F}$ over $\mathcal{R}$ from observation o_t and reward r_t .

3.2.3 The picture so far

It is worth pausing here to see that we have discussed three strands which are running in parallel. The first is the learning system from Chapter 2 (Figure 3.1a). A learning system, on this view, is any system that reacts to an environment $\mathcal{E}$ through a representational vocabulary $\mathcal{R}$ and a learning mechanism $\mathcal{F}$, producing an internal state s on which behavior depends. The dashed arrows are the feedback loop we argued is missing from most accounts of human learning: experience does not only produce behavior, it also reshapes $\mathcal{R}$ and $\mathcal{F}$ themselves, which in turn changes how future experience is organized. Section 2.4 made this concrete by committing to a specific object—the generative library $\mathcal{L}$ of reusable entries—as the content of $\mathcal{R}$, with $\mathcal{F}$ as inference over compositions of those entries (Figure 3.1b).

The second is the *latent cognitive model*, the standard empirical tools of cognitive science (Figure 3.1c, Section 3.1). A hidden variable z is posited as the cause of observable behavior x —responses, action sequences, or reaction times—and modeling the agent amounts to specifying a prior over z and a likelihood $p(x | z)$. This deliberately minimal formulation compresses the learning system into a single latent variable summarizing

the internal organization the modeler aims to infer. Classical cognitive models often operate at this level, with z representing, for example, a trait parameter, learning rate, belief distribution, or knowledge state.

This latent-variable view explains behavior by inferring hidden structure, but it is largely static unless extended with dynamics. The third framework, *reinforcement learning* (RL), adds those dynamics by modeling the agent's interaction with its environment over time (Figure 3.1d, Section 3.2.1). In RL, the environment has a state s_t that evolves under the agent's action a_t , while the agent maintains a value function v_t and policy $\pi(a_t | s_t, v_t)$ for selecting actions. This makes RL a natural formalism for cognitive phenomena such as exploration, planning, habit formation, and feedback-driven choice. However, standard RL typically takes s_t to be the externally specified environment state, designed into the task by the experimenter, with values and policies defined over that state space. This convention is useful for simple environments and parameter-fitting, but it assumes away the question raised in Chapter 2: what representational vocabulary does the agent use, and how does it change?

This is why this thesis adopts an *agent-centric* reformulation (Figure 3.1e and Section 3.2.4). The environment remains on the outside, specified by its transition dynamics T alone—reward is deliberately not part of its definition⁹. The agent gains its own internal compartment containing $\mathcal{R}$, $\mathcal{F}$, and an internal state z_t that bundles belief, value estimate, and any other structure it maintains. Observations enter from the environment and actions return to it; reward, where it appears, is a signal the agent receives and evaluates rather than a property of the world it acts in. The feedback loop from (a)—experience reshaping the library, the library reshaping computation—is now operative inside an acting agent. Panel (e) is panel (a) embedded in an acting-agent form, and it is the picture the remainder of the thesis works within.

9: Standard RL writes the environment as (S, A, T, R) , putting reward on the same footing as the dynamics — a property of the world, handed to the agent as ground truth. But a chess position is not intrinsically rewarding; it is rewarding *to* an agent that wants to win. Following Abel et al. (2024), this thesis keeps reward out of the environment: the world has dynamics, the agent has goals, and reward is what arises when the two meet.

What the latents are, in current cognitive models. Viewed through Figure 3.1e, reward parameters $\theta_{1:K}$ (Baker et al., 2009; Ng, Russell, et al., 2000) live inside the agent-compartment as the agent's model of what the environment pays out; value estimates $\hat{V}(z_t)$ (Daw, 2011) are the cached consequence of reward and transition structure, compressed for rapid decision; policy parameters such as the softmax temperature β (Wilson et al., 2014) govern how internal value is translated into action. Each of these is a latent in the sense of Figure 3.1(c)—a hidden cause of observed behavior to be recovered from data—and each sits in a specific slot of the agent-centric diagram. The next subsection examines them component by component, asking what each buys explanatorily and what classical experiments isolate it.

3.2.4 The agent's internal world model

The agent-centric decomposition in Equation (3.10) is not a set of independent modules. Together, the representation ϕ , reward structure R , value function V , and policy π form the agent's *internal world model*: the generative process by which it predicts, evaluates, and acts. For an observer, this model is latent; choices and outcomes are observed, but the structure producing them must be inferred. This applies the latent-inference logic of Section 3.1 to acting agents. Each component below is grounded in a cognitive experiment that isolates it.

Reward. The simplest internal component to infer is the agent's belief about reward, given that its goal—maximize reward—is stipulated by

10: Inferring *wants* from behavior alone is the harder problem of inverse RL or revealed preference (Ng, Russell, et al., 2000): choices are jointly compatible with many (goal, belief, decision rule) triples, so something must be fixed for the rest to be identified. Bandits fix the goal.

Example 3.2 (Contrast: category prototypes.) A standard category-learning model (Nosofsky, 1986) posits latent prototypes $\mu_{1:K} \in \mathbb{R}^D$ and attention weights $w \in \mathbb{R}_{\geq 0}^D$, with response model

$$p(y = k | s, \mathbf{z}) \propto \quad (3.11)$$

$$\exp(-c \sum_d w_d |s_d - \mu_{k,d}|). \quad (3.12)$$

This latent is behavior-generating (changing μ_k changes predictions) but not structurally legible in the relevant sense: prototypes do not compose, cannot be transmitted as discrete entries to another agent, and do not change $\mathcal{R}$ itself. They are latents *within* a fixed $\mathcal{R}$, not latents that reshape it.

11: Bisimulation metrics (Ferns et al., 2004) formalize this as a distance on states: s and s' are bisimilar if they yield the same reward and their successor distributions are close under the same metric. Information-theoretic bounds (Still and Precup, 2012) provide the complementary view: z_t is a good representation if $\text{MI}(z_t; r_{t+1}, z_{t+1} | a_t)$ is high relative to $\text{MI}(z_t; h_t)$ —the representation retains what matters for prediction while discarding what does not.

12: Tolman’s experiment illustrates *latent learning*: experience updates the representation ϕ without changing behavior until reward creates demand. Formally, ϕ changes while the policy $\pi(\cdot | \phi(h_t))$ remains uniform. Standard RL, which ties representation to reward, lacks a natural vocabulary for this dissociation.

13: Partially observed variants such as POMCP (Silver and Veness, 2010) extend the same idea to belief states.

the task¹⁰. The multi-armed bandit problem (Auer et al., 2002) isolates this inference because actions do not change the state of the world—the only hidden structure is the subject’s belief about reward and the rule by which it turns belief into choice. An agent faces K options, each yielding stochastic rewards with unknown parameters $\theta_{1:K}$. Watching the sequence of choices and rewards, the experimenter asks what internal organization could have produced it—a Bayesian observer reasoning about a (possibly) Bayesian subject (Daunizeau et al., 2010):

$$p(\theta_{1:K} | a_{1:t}, r_{1:t}) \propto p(\theta_{1:K}) \prod_{i=1}^t p(r_i | \theta_{a_i}). \quad (3.13)$$

If the subject’s decision rule is also uncertain — for instance a softmax $\pi(a | b) \propto \exp(\beta \hat{Q}(a; b))$ with unknown inverse temperature β — the experimenter’s latent expands to $(\theta_{1:K}, \beta)$ and the likelihood must account for both reward generation and action selection (Daw, 2011). The two-step task of Daw (2011) uses this structure to ask not what the subject wants (fixed) but how it computes:

$$p(\theta_{1:K}, \beta | a_{1:t}, r_{1:t}) \propto p(\theta_{1:K}) p(\beta) \prod_{i=1}^t \underbrace{p(r_i | \theta_{a_i})}_{\text{reward model}} \underbrace{\pi(a_i | b_i; \beta)}_{\text{choice model}}, \quad (3.14)$$

revealing whether the subject tracks reward through a model of transition structure or through cached action values. Even here, the experimenter’s inferential object is already a structured bundle — the subject’s belief about reward, and the rule that turns belief into choice.

Internal state structure. In the bandit, the only thing the subject had to maintain was a belief over a fixed set of arms. Once the task has sequential structure, the observer must additionally infer how the agent compresses its history—what it preserves, what it discards. The mapping $\phi : h_t \mapsto z_t$ produces an internal state intended to be sufficient for prediction and control, and the formal criterion is whether z_t is a *sufficient statistic* for the relevant downstream quantities¹¹. Different choices of ϕ induce different effective state spaces $\mathcal{Z} = \{\phi(h) : h \in \mathcal{H}\}$, different value landscapes $V : \mathcal{Z} \rightarrow \mathbb{R}$, and different patterns of generalization. Two agents can face the same task and yet inhabit different control problems if they compress the task differently.

Tolman (1948) demonstrated this point directly. Rats allowed to explore a maze without reward later took novel shortcuts as soon as a goal was introduced—behavior that is unintelligible without positing a spatial representation built *before* it was instrumentally useful. The representation had no behavioral signature during exploration; it became visible only when the task demanded it¹². For this thesis, internal state structure determines $\mathcal{R}$ in the learning-system triple $(\mathcal{R}, \mathcal{F}, \mathcal{E})$ introduced in Section 2.4, and it defines which distinctions are available for action, prediction, and reuse.

Value. A model-based agent computes value by simulating forward through its internal world model—exactly in small problems, but in general through approximations such as truncated rollouts, learned value bootstraps, or Monte Carlo tree search (Silver et al., 2017; Silver and Veness, 2010)¹³,

$$V^m(z_t) = \mathbb{E}_{\pi, m} \left[\sum_{\tau \geq t} \gamma^{\tau-t} R(z_\tau, a_\tau) \right], \quad (3.15)$$

where the expectation runs through the agent's own learned transition model $m(z_{t+1} | z_t, a_t)$. A model-free agent instead caches a value estimate $\hat{V}(z_t)$ updated incrementally from experienced returns via temporal-difference learning,

$$\hat{V}(z_t) \leftarrow \hat{V}(z_t) + \alpha [r_t + \gamma \hat{V}(z_{t+1}) - \hat{V}(z_t)]. \quad (3.16)$$

The two computations can produce identical behavior under a stable reward distribution. The reward devaluation paradigm developed by Dickinson (1985) isolates the difference. An animal is trained to press a lever for food; the food is then paired with illness to reduce its desirability. An agent using Equation (3.15) recomputes expected return through its model, and the devalued outcome changes the computation—the animal stops pressing. An agent using Equation (3.16) retains its cached estimate, which is insensitive to the change in reward—the animal persists. The two animals produced identical behavior before devaluation; only the probe exposes the different latent value structures¹⁴.

Policy. The policy $\pi(a_t | z_t)$ maps internal state to action. Although choices are observed as samples from π , the strategy generating them remains latent. In the same bandit problem, for example, a *directed* exploration policy selects actions to maximize expected information gain about reward parameters,

$$a_t = \arg \max_a \left[\hat{Q}(a; b_t) + \beta_{\text{info}} \sqrt{\text{Var}_{b_t}[\theta_a]} \right], \quad (3.17)$$

where the second term is an exploration bonus proportional to the agent's posterior uncertainty about option a . A *random* exploration policy adds noise to the value estimate,

$$a_t \sim \text{softmax}(\beta \hat{Q}(\cdot; b_t)), \quad (3.18)$$

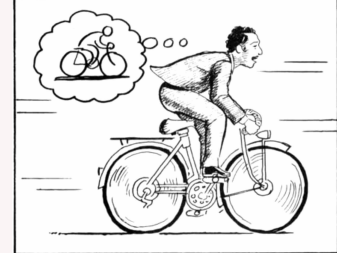
where the inverse temperature β controls the randomness. Both policies may produce the same observable action—choosing the uncertain arm—but for different reasons. Wilson et al. (2014) demonstrated this dissociation through a horizon manipulation. When the remaining horizon was long, participants preferentially sampled the unknown option in a manner consistent with Equation (3.17). When the horizon was short, behavior resembled Equation (3.18). Inferring policy from behavior therefore requires disentangling the choice rule from the representation and value function it operates over.

The coupled model. The four components above—reward, representation, value, and policy—form a coupled system: the agent's internal world model. Representation ϕ is foundational, defining the state space $\mathcal{Z}$ over which reward R , transition model m , value V , and policy π operate. The construal dynamics $m(z_{t+1} | z_t, a_t)$ predicts action consequences in this representational language; distinctions discarded by ϕ cannot be recovered downstream, while predictive regularities learned by m pressure ϕ to preserve them. Value integrates representation and reward over time, via forward simulation (Equation (3.15)) or cached updates (Equation (3.16)), and policy is the observable surface that inherits the structure of all components beneath it.

Different traditions foreground different parts of this coupled model. Inverse reinforcement learning (IRL) (Baker et al., 2009; Ng, Russell, et al., 2000) treats reward as the primary latent, asking which objective would rationalize observed behavior while taking representation and dynamics as given. Computational cognitive science and cognitive neuroscience (Daw,

14: This dissociation between model-based and model-free value is a major bridge between RL and cognitive neuroscience (Daw et al., 2005; Gläscher et al., 2010). Neuroimaging evidence links model-based values to prefrontal computation, cached values to dorsolateral striatum, and behavior to arbitration between the two systems (Lee et al., 2014).

Remark 3.2 (World models Ha and Schmidhuber (2018b).)



A world model decomposes the agent into three components: a *visual encoder* $\phi : o_t \mapsto z_t$ that compresses observations, a *memory model* $m : (z_t, a_t) \mapsto z_{t+1}$ that predicts latent dynamics, and a compact *controller* $\pi : z_t \mapsto a_t$. Once trained, the agent can plan entirely inside the learned latent space—“dreaming” trajectories without real-environment interaction. This architecture instantiates the decomposition in Equation (3.10): ϕ determines $\mathcal{R}$, m approximates $\mathcal{T}$, and π constitutes $\mathcal{F}$.

2011; Gläscher et al., 2010) emphasize value computation, using probes such as devaluation to distinguish model-based from model-free evaluation, typically within a fixed state space. Representation learning in deep RL (Lesort et al., 2018; Schwarzer et al., 2021) optimizes the encoder ϕ , but evaluates it instrumentally through downstream performance rather than as a behavioral latent. What remains underdeveloped is treating ϕ itself as the latent object of inference—like reward in IRL or value in cognitive modeling—from the patterns of action and generalization it produces. This is the gap the thesis addresses.

3.2.5 Mapping to the library lifecycle

This section links the agent-centric decomposition (Equation (3.10)) to the library framework: the RL components map onto $(\mathcal{R}, \mathcal{F}, \mathcal{E})$, the representational dynamics gap from the preceding chapter reappears in RL-specific form, and the rest of the chapter develops the tools to close it. The state representation ϕ fixes $\mathcal{R}$ —it determines which hypotheses can be concisely expressed and which distinctions are available for action and prediction. The policy, world model, and learning rules constitute $\mathcal{F}$, the processes operating over those resources. The task distribution is $\mathcal{E}$.

Standard RL fixes $\mathcal{R}$ —through a hand-crafted state space or a fixed architecture—and optimizes $\mathcal{F}$ within it. This sidesteps the question Section 2.3 identified as central: what happens when ϕ itself changes? When it does, the state space $\mathcal{Z}$ shifts under it, the world model, value, and policy are no longer defined over the same objects, and an entry in $\mathcal{L}$ that compressed a recurring pattern under the old ϕ may become redundant or misleading under the new one.

This thesis therefore treats ϕ as evolving by treating its substrate as latent: the library $\mathcal{L}$ structures what ϕ can express, so studying ϕ 's dynamics reduces to studying the lifecycle of $\mathcal{L}$ —how representational resources are constructed, curated, and communicated through experience. This is the representational dynamics gap in its RL-specific form, and it is logically prior to the gaps surveyed above: IRL, value-computation modeling, and representation learning each presuppose a stable ϕ , so progress on them depends on having a framework for the case where ϕ does not stand still.

The remaining sections develop the tools to close it. Section 3.3 treats representational formats and library learning as mechanisms for constructing ϕ and $\mathcal{L}$; Section 3.4 treats the coupling between representations and world models as curation under sequential interaction; Section 3.5 extends the framework to communication, where one agent must model how actions update another's internal state; Section 3.6 synthesizes these through compression and resource rationality, the thesis-wide principle for evaluating representational infrastructure as it changes.

3.3 Programs as structured latent representation

The previous section decomposed the agent's latent structure into reward, value, representation, and a model over that representation. We argued that reward and value inference are well studied, while the representation determining $\mathcal{R}$ —and the possible form of ϕ more generally—remains open. Because the generative library in Section 2.4 requires entries to be addressable, reusable, compositional, and extensible, not all latent

formats suffice. This section develops a progression of increasingly structured latent formats, culminating in *program induction* (Lake et al., 2015; Rule et al., 2018) as the formalism used here to construct such representations.

3.3.1 Structure of latent representations

The structure of the latent variable determines what inference can recover, which distinctions support prediction and control, and whether the representational vocabulary can change. Four progressively richer formats make this contrast clear.

Flat parameters. The simplest latent is $\mathbf{z} \in \mathbb{R}^d$, with fixed dimensionality and interpretation set by the modeler. In cognitive models, such parameters encode learning rates, inverse temperatures, forgetting rates, reward sensitivities, and related quantities (Daw et al., 2011; Halahakoon and Browning, 2023; Wilson and Collins, 2019). They are useful when the ontology is known and the goal is to explain graded variation across individuals or conditions. For example, in the Rescorla-Wagner model (Rescorla, 1972), the latent is a vector of associative strengths $v_1, \dots, v_K$, one per cue, updated by $\Delta v_i = \alpha(\lambda - \sum_j v_j)$ when cue i is present. Fitting the model recovers α and the associative weights, capturing responses to familiar cues. However, it does not recover organization among dimensions or introduce new primitives when the existing vocabulary is inadequate.

Structured but fixed decompositions. A richer form lets the latent decompose into components with explicit dependency structure (Koller and Friedman, 2009). A directed graphical model specifies latents $z_1, \dots, z_n$ and a graph of parent relations, giving

$$p(\mathbf{z}) = \prod_i p(z_i \mid \text{pa}(z_i)),$$

with analogous decompositions in undirected, factorial, and dynamical-state variants (Ghahramani, 2013; Ghahramani and Hinton, 2000; Jordan et al., 1999). In cognitive modeling, the graph itself is sometimes the latent to be inferred. Kemp and Tenenbaum (2008) cast the structural *form* of conceptual organization — tree, ring, chain, hierarchy — as a hypothesis to be learned from data, with the inferred graph serving as a generative model of the learner’s conceptual structure. Causal-Bayesian accounts of intuitive theories extend the move (Frosch et al., 2012; Gopnik, 2007; Griffiths and Tenenbaum, 2005). The explicit structure is what makes inference of this kind interpretable: one can ask which components depend on which, and which parts of the latent explain particular behavioral patterns (Kemp and Tenenbaum, 2009). But the representational primitives remain fixed. Inference recovers relations among given primitives rather than introducing new ones.

Compositional executable structure. Probabilistic generative programs (Goodman, 2013) introduce a further step. Let ρ be a program defined by a formal language L . The latent is now an executable symbolic object whose semantics are given by what it does when run. Two properties distinguish this format from fixed decompositions. First, programs are *compositional*: graph-based models recover relations among predefined units while program-like models recover how a complex object is assembled from reusable parts. Second, they are *executable*: a program can be run to produce predictions, actions, or plans, providing a direct behavioral semantics. This matters because the representation

no longer specifies only which variables depend on which others, but a procedure for generating, transforming, or controlling structured outcomes. Lake et al. (2015) showed human learners who saw a single example of a novel handwritten character could generate new examples, parse the character into strokes, and recognize it across variation. They reproduced this one-shot generalization with a probabilistic program that decomposed characters into stroke primitives, spatial relations, and motor routines.

Extensible libraries. Programs become most relevant for this thesis when composition is coupled with *abstraction*. Recurring subexpressions can be compiled into named procedures and added to a library, thereby extending the effective vocabulary of the hypothesis space itself¹⁵. This is the step that turns compositional representation into representational construction proper. Flat parameters vary values within a fixed basis. Structured decompositions recover relations over a fixed basis. Programs express procedures within a fixed language. Libraries change the basis itself by introducing new reusable building blocks. For that reason, extensible program-like libraries provide the cleanest formal setting for studying representational construction.

15: The connection to hierarchical RL is direct. Options, temporally extended skills, and macro-actions all introduce reusable control fragments (Sutton et al., 1999). Library learning generalizes the idea by allowing such reusable units to be discovered from data rather than fixed by design.

3.3.2 Program induction as inference over structured representations

A program ρ is a well-formed symbolic expression in a formal language L , and its meaning is given by what happens when it is executed. The language L defines a set of primitives and composition rules, fixing which procedures are expressible¹⁶. Program induction treats programs as latent hypotheses that explain observed data $\mathcal{D}$: inference asks which program in L best accounts for the data, with the language fixing the hypothesis space and execution providing the link between hypothesis and observation.

Bayesian formulation. Given observed data $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$, the posterior over programs is

$$p(\rho \mid \mathcal{D}) \propto p(\mathcal{D} \mid \rho) p(\rho), \quad (3.19)$$

where $p(\mathcal{D} \mid \rho)$ is typically obtained by executing ρ and evaluating how well its outputs account for the data. The prior commonly takes a description-length form,

$$p(\rho) \propto 2^{-|\rho|_L}, \quad (3.20)$$

where $|\rho|_L$ denotes program length in language L . Shorter programs receive exponentially higher prior probability. Via the coding theorem, description length approximates Kolmogorov complexity up to an additive constant (Li, Vitányi, et al., 2008), such that the prior in Equation (3.20) can be read as a computable approximation to a universal simplicity prior, relativized to the language L ¹⁷.

Intractability and approximation. The hypothesis space $\mathcal{H}_L = \{\rho : \rho \text{ is well-formed in } L\}$ is combinatorial and typically infinite. Program induction is therefore intractable for two separate reasons: (i) inference must search over a vast space of candidate programs, and (ii) each candidate must be executed and scored against the data. The marginal likelihood $p(\mathcal{D}) = \sum_{\rho \in \mathcal{H}_L} p(\mathcal{D} \mid \rho) p(\rho)$ is therefore generally not computable exactly. In practice, approximate inference is required. Different

16: A formal language can be defined recursively. For example, let the primitives be $\{a, b\}$. A program is either (i) a primitive, or (ii) $\text{concat}(\rho_1, \rho_2)$ where ρ_1 and ρ_2 are already programs. This generates expressions such as a , $\text{concat}(a, b)$, and $\text{concat}(a, \text{concat}(b, a))$.

17: The coding theorem states that a hypothesis's negative log-probability under a universal prior equals, up to an additive constant, its Kolmogorov complexity: the length of the shortest program generating it on a universal Turing machine. The description-length prior in Equation (3.20) inherits this normative motivation but remains computable because L is fixed.

methods approximate different parts of the problem. The principal strategies are enumeration for small domain-specific languages, stochastic search and MCMC over local program transformations (Goodman et al., 2008), amortized neural proposals $q_\phi(\rho \mid \mathcal{D})$ that learn data-to-program mappings (Devlin et al., 2017), and sequential Monte Carlo methods that build programs incrementally. Here, the choice of approximation interacts with the structure of L : richer languages can express more compact solutions but demand more sophisticated search.

Description length as state abstraction. The description-length prior makes a deep connection between program induction and the compression-map view of representation from Section 3.2. A short program that explains the data well is, in effect, a compact state abstraction (Li et al., 2006). It identifies the minimal structure in the data that is sufficient for prediction. The mutual information criterion introduced in the representation paragraph of Section 3.2.4—a good z_t is one where $\text{MI}(z_t; r_{t+1}, z_{t+1} \mid a_t)$ is high relative to $\text{MI}(z_t; h_t)$ —has a direct analogue here. A good program ρ is one where $p(\mathcal{D} \mid \rho)$ is high (the program predicts well) relative to $|\rho|_L$ (the program is short). The difference is that program induction searches over discrete, compositional, and interpretable compression maps rather than over continuous parameter spaces. Symbolic form does not guarantee transparency—long programs and opaque learned primitives can resist interpretation—but it does make compression auditable in a way continuous representations are not: the regularities captured, abstractions introduced, and distinctions drawn are at least addressable as objects, even when they resist easy reading.

3.3.3 Library induction as representational change

Inferring a program for a single task still does not fully capture the concept of construction that this thesis seeks to describe. Here, we are interested in how the construction of new representations (from reusable structures extracted from earlier solutions) can change the space of later solutions.

The generative library $\mathcal{L}$ introduced in Section 2.4 provides the formal object for this change. Adding a new entry ℓ to $\mathcal{L}$ extends the language from L_0 to $L_0 \cup \mathcal{L}$, making programs that invoke ℓ shorter than equivalent programs over L_0 alone. This affects the prior where a program ρ that was long (and therefore improbable) in the base language may become short (and therefore probable) in the extended language if it invokes library entries that compress recurring subexpressions. *Library induction* optimizes the library itself,

$$\mathcal{L}^* = \arg \max_{\mathcal{L}} \sum_{i=1}^N \log p(\mathcal{D}_i \mid L_0 \cup \mathcal{L}) + \log p(\mathcal{L}), \quad (3.21)$$

where $p(\mathcal{D}_i \mid L_0 \cup \mathcal{L})$ marginalizes over programs in the extended language and $p(\mathcal{L})$ penalizes overly large or overly specific libraries. This is the formal distinction between learning *within* a representation and learning *of* a representation. Parameter estimation, belief updating, and policy improvement all operate within a fixed $\mathcal{R}$. Library induction changes $\mathcal{R}$ by altering which hypotheses are concise, which candidates are likely under the prior, and which solutions are easy to search for. In the terminology of Section 2.3, it is the meta-level problem made tractable through a specific computational mechanism.

The dynamics of library growth. Library induction is iterative rather than one-shot: the agent solves tasks with the current library, extracts recurring subexpressions from successful solutions, compiles them into new entries, and reuses the enriched library on later tasks. At iteration t , the dynamics can be written as DREAMCODER (Ellis et al., 2021) instantiates this logic as a wake-sleep cycle (Hinton et al., 1995): search occurs in the wake phase, and abstraction extraction in the sleep phase.

$$\begin{aligned}\rho_{i,t}^* &\sim p(\rho \mid \mathcal{D}_i, L_0 \cup \mathcal{L}_t), \\ \mathcal{L}_{t+1} &= f(\mathcal{L}_t, \{\rho_{i,t}^*\}),\end{aligned}\tag{3.22}$$

where $\{\rho_{i,t}^*\}$ are the task solutions found under $\mathcal{L}_t$ and f is the abstraction-extraction procedure.

This iterative structure is the mechanistic basis for path dependence during learning. This is because the library learned from early tasks will shape the search space for later tasks, i.e., programs that invoke early entries are short, hence more probable, and thus more likely to be found and extended. Different task orderings produce different libraries (Zhou et al., 2024b), which define different hypothesis spaces, which support different patterns of generalization¹⁸. This provides a precise computational mechanism for the construction problem defined in the preceding chapter (Section 2.3.3).

What library induction isolates. Library induction isolates representational change from action selection. The agent updates its hypothesis language by adding reusable abstractions, but the task distribution is taken as given. This isolation makes it possible to study how new library entries change description length, posterior mass, and future search without also modeling exploration or control. However, in acting agents, representation learning is coupled to policy, prediction, and data collection. What the agent observes next depends on what it does, and what it does depends on its current representation. Library induction captures the constructive part of this process in its simplest form. The next section introduces the richer setting in which representational change and action are coupled.

18: Elman (1993) demonstrated a related phenomenon in connectionist language learning: networks trained first on simple sentences and then on complex ones learned grammatical structure that networks exposed to the full distribution from the start could not acquire. The mechanism is different (capacity limitation rather than library growth), but the consequence is the same: the order of experience is a structural determinant of the resulting representation.

3.4 World models as learned latents

The preceding section isolated representational construction: how program and library induction add new entries to $\mathcal{L}$. There, tasks are exogenous and only the agent’s internal vocabulary changes. In the real world, however, actions also change the external environment and the problem the agent faces (Kaelbling et al., 1998; Sutton, 1991). Actions determine which observations arrive; observations update representations and predictive models; and these revised models shape future action.

This section treats the *coupled representation-model-policy system* as the latent. Here, the latent is the internal system that maps history to state, predicts future states, and selects action,

$$\begin{aligned}z_t &= \phi(h_t), \\ \hat{z}_{t+1} &\sim m(z_{t+1} \mid z_t, a_t), \\ a_t &\sim \pi(a_t \mid z_t).\end{aligned}\tag{3.23}$$

The inferential target is therefore not a single representation, but the dynamics of a coupled system whose components co-adapt over time.

The learning trajectory itself is informative: which errors persist, which distinctions sharpen, and how action changes the data available for learning. This creates a closed feedback loop in which $\mathcal{R}$ and $\mathcal{F}$ co-evolve—the coupled dynamics diagnosed in Section 2.3, now requiring a formal mechanism. This section develops that mechanism.

3.4.1 The coupling between representation and prediction

In an acting agent, representation and prediction cannot be characterized independently (Ha and Schmidhuber, 2018b; Hafner et al., 2023). The world model $m(z_{t+1} | z_t, a_t)$ predicts consequences of action, but those predictions are only as good as the state $z_t = \phi(h_t)$ over which they are expressed. If the representation collapses histories that imply different future observations or rewards, no downstream model can recover that information from z_t alone (Ferns et al., 2004; Gelada et al., 2019; Still and Precup, 2012). Conversely, if successful prediction repeatedly depends on a distinction that the current representation blurs, learning pressure favors encodings that preserve it more faithfully. Because action changes which observations arrive next, updates to ϕ , m , and π are coupled through the data the agent collects.

Connection to the belief MDP. The formal significance of this coupling becomes clear through the belief MDP introduced in Section 3.2.1. Under exact Bayesian control, the sufficient state is the belief $b_t \in \Delta(S)$, updated by Equation (3.6) and evaluated by the Bellman recursion in Equation (3.8). A learned world model replaces this exact computation with an approximation in the agent’s own representational language¹⁹. The expected value of the learned policy then falls short of the optimal belief-MDP value by a sum of three terms²⁰: a *representation* term, capturing return lost because ϕ collapses histories the task requires distinguishing; a *model* term, capturing return lost because m misrepresents dynamics in latent space; and a *policy* term, capturing suboptimality given (z_t, m) . The representation term is upstream of the other two: if ϕ merges states that the task requires distinguishing, no improvement in m or π can recover the information. This asymmetry is what makes representational curation central in the scaling setting. The cost of a poor representation is not confined to the encoder but propagates through prediction and control.

19: The encoder ϕ maps history to a compact state z_t that serves as an approximate sufficient statistic for the full belief b_t . The transition model m approximates the belief update $\tau(b_t, a_t, o_{t+1})$ without the full Bayesian computation. The policy π acts on this learned state rather than on the exact posterior.

20: Formal decompositions of this kind—with bounds in units of expected return under a fixed initial distribution—appear in simulation lemma analyses (Kearns and Singh, 2002) and bisimulation-based bounds (Ferns et al., 2004). We sketch the structure here; the asymmetry across the three terms is what matters for the present argument.

3.4.2 The ambiguity of prediction error

The coupling described above manifests differently depending on whether the representation is given or learned, and the difference is not merely implementational—it determines what can be inferred from observed prediction errors (Lesort et al., 2018).

In the simplest model-based setting (e.g., Dyna; Sutton, 1991), the agent maintains a tabular model $\hat{T}(s' | s, a)$ estimated from experienced transitions and uses it for simulated planning. The state space is given, so the representation ϕ is the identity and the representation source of error from the previous paragraph drops out: every prediction error is unambiguously model error. Sutton (1991) demonstrated this directly—when a shortcut opens in a maze, the tabular model continues to simulate the old path until real exploration discovers the change, a case of model error that is diagnosed and corrected by further experience.

Once ϕ is learned jointly with m , the situation changes qualitatively. A prediction error $\delta_t = \|z_{t+1} - m(z_t, a_t)\|$ now has two possible sources: the model m may be wrong about the dynamics, or the encoder ϕ may have mapped the observation to a latent state that does not support accurate prediction (Sun et al., 2024). The error decomposes as

$$\delta_t = \underbrace{m(z_t, a_t) - m^*(z_t, a_t)}_{\text{model inadequacy}} + \underbrace{m^*(z_t, a_t) - z_{t+1}}_{\text{representation inadequacy}}, \quad (3.24)$$

where m^* denotes the best possible model given the current representation ϕ . The first term vanishes as the model converges; the second cannot be reduced by improving m alone—it reflects information that ϕ has discarded²¹. The trajectory of learning—which errors decrease, *when* they decrease, and *whether* they reappear after a period of stability—carries information about the coupling between ϕ and m that snapshot performance cannot reveal.

Learned world models in practice. DREAMER (Hafner et al., 2023) instantiates this coupled learning. The architecture learns an encoder $q_\phi(z_t | o_{\leq t}, a_{< t})$, a transition model $p_\psi(z_{t+1} | z_t, a_t)$, a reward predictor $p_\xi(r_t | z_t)$, and a decoder $p_\chi(o_t | z_t)$, trained jointly by maximizing the evidence lower bound on observed trajectories,

$$\begin{aligned} \text{ELBO} = \sum_{t=1}^T & \left[\underbrace{\mathbb{E}_{q_\phi}[\log p_\chi(o_t | z_t)]}_{\text{reconstruction}} + \underbrace{\mathbb{E}_{q_\phi}[\log p_\xi(r_t | z_t)]}_{\text{reward prediction}} \right. \\ & \left. - \underbrace{D_{\text{KL}}[q_\phi(z_t | h_t) \| p_\psi(z_t | z_{t-1}, a_{t-1})]}_{\text{dynamics consistency}} \right]. \end{aligned} \quad (3.25)$$

Here, reconstruction encourages z_t to preserve observable detail; reward prediction encourages it to preserve task-relevant information; and the dynamics-consistency term requires that the representation be *predictable from the model's own dynamics*—the encoder must produce states that the transition model can forecast, and the transition model must learn dynamics that match what the encoder produces.

The tension between reconstruction and reward prediction is the curation trade-off of Equation (2.3) expressed in continuous form²²: reconstruction preserves all observable information, including task-irrelevant detail; reward prediction preserves only what matters for control. The ELBO balances these pressures implicitly through gradient descent, but at the level of the aggregate loss, without isolating individual representational decisions. A representation can predict the next latent state, reconstruct observations, and estimate reward accurately—yet still be organized around the wrong abstractions for future tasks. The ELBO cannot detect this because it evaluates the representation against current data, not against the distribution of future demands (Lan et al., 2022): the representation source of error may be small on the training distribution and catastrophic on the next one.

3.4.3 Model-based learning makes curation visible—and exposes what is missing

Model-based reinforcement learning (MBRL) makes the internal predictive structure an explicit explanatory target (Gelada et al., 2019). It also

21: This decomposition is the world-model analogue of the bias-variance decomposition in supervised learning. Model inadequacy is reducible error (analogous to variance); representation inadequacy is irreducible given the current ϕ (analogous to bias from a misspecified model class). The agent cannot tell, from the prediction error alone, which component dominates.

22: The parallel between DREAMER's world model and DREAMCODER's program library is instructive. Both compress experience into reusable internal structure; both support internal simulation (DREAMER “imagines” trajectories, DREAMCODER “dreams” synthetic tasks from the library). However, a library entry can be individually named, evaluated, and transmitted; a latent dimension cannot.

learns a model of how actions lead to future states and outcomes, and errors in behavior can now be diagnosed at multiple internal levels—a poor policy given a good model, an inaccurate model, or a representation that does not support the distinctions an accurate model would require. World models make curation visible: they provide a setting in which representational adequacy is tested continuously by prediction rather than only by final task reward²³.

The cost of wrong representations. Prediction makes representational adequacy visible, but it also reveals why curation is needed. When the learned representation or model is misaligned with the task, the agent predicts the wrong consequences of action and therefore plans against the wrong problem. An agent whose world model $\hat{M}$ diverges from the true environment M will compute plans that fail, even when those plans are optimal under the believed model,

$$\mathbb{E}_{\hat{M}}[R] \geq \mathbb{E}_{\hat{M}}[R \mid \pi_M^*] \quad \text{does not imply} \quad \mathbb{E}_M[R \mid \pi_{\hat{M}}^*] \geq \mathbb{E}_M[R \mid \pi_M^*], \quad (3.26)$$

where π_M^* is the policy optimized under the wrong model and $\pi_{\hat{M}}^*$ the policy under the true one. The agent may be confident and wrong²⁴. An agent that only *constructs* (by adding to her library without evaluating what it already has) will inevitably accumulate representational debt: a growing collection of entries, some useful, some obsolete, and some actively misleading. The frame problem, recast in these terms, is the ongoing tax on representational vocabulary that has not been maintained.

Intrinsic motivation does not close this gap. Intrinsic motivation signals—novelty, information gain, empowerment—are sometimes proposed as solutions by encouraging more exploration to hope for better generalization. These signals direct *exploration*: they guide the agent toward states that are unfamiliar, informative about model parameters, or controllable (Berlyne, 1950, 1960; Pathak et al., 2017). Two examples are especially relevant here because both are computed through the world model and therefore depend on the current representation. Information gain (Loewenstein, 1994) rewards actions that reduce uncertainty about world-model parameters θ ²⁵,

$$\text{IG}(s, a) = \mathbb{E}_{s'}[D_{\text{KL}}(p(\theta \mid s, a, s') \parallel p(\theta \mid s, a))]. \quad (3.27)$$

Empowerment (Klyubin et al., 2005) rewards states with high channel capacity from actions to future states²⁶,

$$\text{Empower}(s) = \max_{\omega(a|s)} \text{MI}(S'; A \mid S = s) \quad (3.28)$$

Information gain tells the agent which transitions are informative about model parameters; it does not ask whether the model's state space is the right one. Empowerment tells the agent which states offer controllability; it cannot assess controllability in regions that the representation does not yet distinguish. Both signals indicate where exploration is valuable under the current model but neither evaluates whether the representation on which that model is built is itself adequate.

The self-referential nature of representational curation. The gap identified above—that standard learning signals evaluate predictions rather than the representations producing them—is a structural consequence of the self-referential problem in Section 2.3.2. Any curation criterion is computed through the agent's own internal model: prediction error in its latent space, information gain over its parameters, and empowerment

23: In model-free RL, the representation is tested only by the reward signal, which is sparse and delayed. In model-based RL, the representation is tested at every time step by the prediction error of the world model, providing a denser and more immediate signal about representational adequacy.

24: The inequality in Equation (3.26) is a restatement of the *optimizer's curse* (Smith and Winkler, 2006) applied to model-based planning. Because the agent optimizes against its own model, biases in the model are amplified rather than averaged out. In the representational setting, errors in ϕ are not additive losses; they are multiplied by the planning process that relies on them.

25: Information gain targets uncertainty in the current world model, not uncertainty about whether the representation itself is adequate. If the representation conflates two states that should be distinguished, information gain cannot detect the conflation because the uncertainty is already defined over the merged state space.

26: White (1959) identified the cognitive analogue of empowerment under the name *effectance motivation*: the intrinsic satisfaction derived from producing effects on the environment. Empowerment presupposes a model accurate enough to estimate mutual information and a representation rich enough to distinguish the outcomes.

through its transition model. Thus, evaluating whether a representation should change requires using the very representation whose adequacy is at issue. How tractable this loop is depends on whether the representation is implicit or explicit. In models like DREAMER, representation is distributed across network weights, so individual representational changes cannot be separated from aggregate gradient updates. A falling prediction error may indicate genuine improvement or merely drift that fits current data while harming generalization. In library-based models like those in Section 3.3, individual entries can be evaluated against Equation (2.3), but doing so still requires deployment in $\mathcal{E}$, consuming the finite budget the criterion is meant to guide.

The curation chapters address this structure from two sides. Chapter 6 shows how intrinsic exploration signals reshape the representations they are computed over, producing feedback-driven pathologies. Chapter 7 offers a partial resolution by computing empowerment over representational-modification operations rather than external states, so the criterion evaluates the library’s internal organization. The next section turns the same formal apparatus outward: the latent is no longer the agent’s own representation, but another agent’s.

3.5 Other agents as latent systems

The preceding sections treated representational change as explicit construction (Section 3.3) and as ongoing curation during action (Section 3.4)—both within a single agent. Communication is the social version of the same problem: the latent target is now another agent’s internal structure, not merely what it believes about the world but the library $\mathcal{L}^j$ through which it interprets, plans, and learns (Baker et al., 2009; Gmytrasiewicz and Doshi, 2005; Ong et al., 2019).

Social cognition is then a nested inference-and-control problem. Agent i observes agent j ’s actions, utterances, demonstrations, and errors, and uses them to infer $\mathcal{L}^j$ (Degen, 2023; Goodman and Frank, 2016) — but communicative actions are also interventions on it: an explanation or demonstration is chosen partly for how it is expected to reshape the listener’s representational state. The latent target is therefore changing *in response to the very actions used to probe it*. A speaker must estimate what the listener can currently represent, choose a message interpretable with those resources, and predict how it will change what the listener can represent next.

3.5.1 Other agents as latent dynamical systems

From agent i ’s perspective, agent j ’s actions are observable data generated by latent internal variables that i does not directly observe. The generative model has the same form introduced in Section 3.1,

$$\begin{aligned} z^j &\sim p(z^j), \\ a_{1:T}^j &\sim p(a_{1:T}^j | z^j, h_{1:T}^j), \\ p(z^j | a_{1:T}^j) &\propto p(a_{1:T}^j | z^j, h_{1:T}^j) p(z^j), \end{aligned} \quad (3.29)$$

where z^j denotes agent j ’s internal state and h^j its observation history. The modeling is unchanged but the ontology of the latent changes. In

social settings, z^j may include goals, beliefs, policies, world models, or—for the purposes of this thesis—the listener’s library $\mathcal{L}^j$.

Computational theory-of-mind models are the canonical example of this inversion (Rescorla, 2015). Baker et al. (2009) showed that human goal attributions in a navigation task can be predicted by a Bayesian inverse-planning model, which inverts a generative model of rational, goal-directed action from observed paths. The inference succeeds by recovering the latent goal that makes the trajectory probable.

This thesis extends the same logic from goals to representational resources. If an agent generalizes across some tasks but not others, or fails systematically when a missing abstraction is required, those behaviors provide evidence about $\mathcal{L}^j$. Inferring another agent’s library is therefore a structured inverse problem: given how the agent acts, explores, errs, and generalizes, what representational organization would produce that pattern?

3.5.2 Communication as control over another agent’s latent state

The multi-agent inference problem becomes especially consequential once agents can communicate. This can be stated in POMDP by extending the state space to include both the external environment and the listener’s latent state,

$$\begin{aligned} s_t &= (s_t^{\text{ext}}, z_t^j), \\ a_t &\in \mathcal{A}^{\text{env}} \cup \mathcal{A}^{\text{comm}}, \\ z_{t+1}^j &\sim T^j(z_{t+1}^j \mid z_t^j, a_t^{\text{comm}}). \end{aligned} \quad (3.30)$$

Here $\mathcal{A}^{\text{comm}}$ is the space of communicative acts and T^j is the listener’s update process. For environmental actions, the transition kernel describes how the world changes. For communicative actions, the relevant transition kernel describes how another agent’s internal organization changes. The target differs, but the formal structure is the same, i.e., action is selected for its expected effect on a future state²⁷.

Bayesian models of pedagogy make this control perspective explicit. Shafto et al. (2014) model teachers as selecting examples to guide a learner’s posterior toward the intended concept, often choosing diagnostic or contrastive examples rather than merely representative ones. In the language of Equation (3.30), the teacher chooses actions by modeling the learner’s update dynamics. The present thesis uses the same idea at the library level: the communicative act is valuable when it moves $\mathcal{L}^j$ or the listener’s current state z^j toward a form that supports future competence.

27: This inversion of the internal/external boundary is the formal reason that communication connects directly to the rest of the thesis. The same apparatus—latent states, action-conditioned transitions, uncertainty about the current state, utility over future consequences—applies whether the agent is acting on the physical world or on another mind.

3.5.3 From epistemic to decision-theoretic communication

The Rational Speech Acts framework (Frank and Goodman, 2012; Goodman and Frank, 2016) provides a computational framework of communication as latent inference and utility-guided action. A literal listener L_0 interprets utterances against a prior over world states,

$$L_0(w \mid u) \propto \llbracket u \rrbracket(w) p(w). \quad (3.31)$$

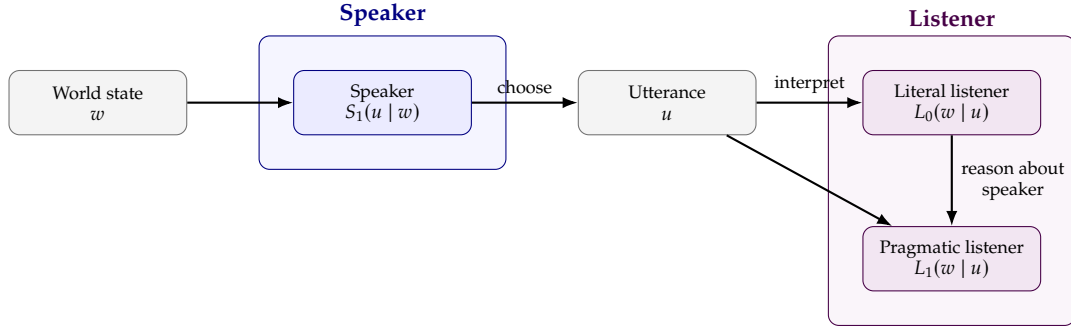


Figure 3.2: Canonical RSA schematic. The speaker chooses an utterance by anticipating listener interpretation, and the listener infers meaning by reasoning about the speaker’s choice.

A pragmatic speaker S_1 selects utterances by reasoning about how the listener will interpret them,

$$S_1(u | w) \propto \exp(\alpha \log L_0(w | u) - C(u)), \quad (3.32)$$

where α is a rationality parameter and $C(u)$ a cost term. A pragmatic listener L_1 inverts the speaker’s model,

$$L_1(w | u) \propto S_1(u | w) p(w). \quad (3.33)$$

28: This is scalar implicature derived from rational inference rather than from a conversational rule. The listener reasons: a rational speaker who knew all students passed would have said “all,” since “all” is more informative and no more costly. The choice of “some” is therefore evidence that the speaker does not know “all” to be true. The pragmatic enrichment emerges from the recursive inference, not from a stipulated maxim.

The recursion produces pragmatic phenomena²⁸—scalar implicature, overinformativeness, referential narrowing—without stipulating them as separate principles. When a speaker says “some students passed” in a context where “all students passed” was available but not chosen, L_1 infers that “all” is false, because S_1 would have used “all” if it were true.

The limitation of epistemic utility. In standard RSA, the speaker’s utility is the listener’s log-posterior at the true state: $\log L_0(w | u)$. The speaker succeeds when the listener *believes* the right thing. For many communicative settings—reference games, scalar inference, politeness—this epistemic criterion is appropriate.

For the settings studied in this thesis, it is insufficient. When a teacher explains a navigation procedure, the goal is not merely that the listener assigns high probability to the correct map. It is that the listener can *act*—navigate successfully, avoid hazards, recover from wrong turns. An explanation that produces correct beliefs about the goal location but does not enable the listener to plan a path is worse than one that enables effective navigation even if belief about the goal is imprecise. The relevant criterion is not epistemic accuracy but downstream competence: the speaker should plan over the listener’s resulting *actions*, with belief accuracy entering only through the actions it produces. This is the cooperative I-POMDP formulation (Gmytrasiewicz and Doshi, 2005), and it is the criterion the communication chapters of this thesis adopt²⁹.

Decision-theoretic communicative utility. The extension replaces the listener’s log-posterior with a measure of downstream action quality,

$$U(u; w, \mathcal{L}^j) = \mathbb{E}_{L(z^j | u, \mathcal{L}^j)}[V(z^j; w)] - C(u), \quad (3.34)$$

where $\mathcal{L}^j$ denotes the speaker’s model of the listener’s current representational resources, $L(z^j | u, \mathcal{L}^j)$ is the listener’s update process under utterance u , and $V(z^j; w)$ measures how well the resulting internal state z^j supports effective action in world state w .

29: This distinction between epistemic and decision-theoretic utility parallels the distinction between model-based and model-free value in Section 3.2.4. Epistemic utility evaluates whether the listener’s internal model is accurate. Decision-theoretic utility evaluates whether the listener’s internal model supports effective action. The latter subsumes the former when the task requires action, because accurate belief is valuable only insofar as it improves decisions.

Three features of Equation (3.34) matter. First, utility depends on the listener’s resulting action capability, not merely belief accuracy. The value function $V(z^j; w)$ asks whether this internal state lets the listener plan, predict, and generalize in the world. A precise but useless representation can therefore score below an approximate but actionable one. Second, utility is conditioned on $\mathcal{L}^j$, the speaker’s model of the listener’s existing knowledge. The same utterance yields different listener states depending on the resources used to interpret it: assuming a missing abstraction makes an explanation incomprehensible, while underestimating the listener makes it verbose. Uncertainty about $\mathcal{L}^j$ is thus a central source of communicative difficulty. Third, the expectation over $L(z^j | u, \mathcal{L}^j)$ makes interpretation stochastic. Listeners may parse, attend to, and integrate the same utterance differently. A good explanation therefore has high expected value across this distribution, remaining robust to variation in listener processing rather than optimal for only one assumed interpretation.

3.5.4 Communication as inter-agent representational alignment

Once communication targets representational state, library overlap becomes a structural determinant of communicative cost (Sucholutsky et al., 2025). A speaker with library $\mathcal{L}^i$ communicating a concept c to a listener with library $\mathcal{L}^j$ pays in proportion to the description length of c under the listener’s vocabulary,

$$C_{\text{comm}}(c; \mathcal{L}^j) \propto |c|_{\mathcal{L}^j}, \quad (3.35)$$

where $|c|_{\mathcal{L}^j}$ is the length of the shortest program in $L_0 \cup \mathcal{L}^j$ that implements c . When the libraries share rich structure, $|c|_{\mathcal{L}^j}$ is short. When they diverge—through different histories, tasks, or curation paths—the speaker must decompose c into primitives the listener has, teach missing entries first, or find an approximate equivalent, and communication becomes a constructive process of building shared representational infrastructure before the target concept can be transmitted³⁰.

Clark and Wilkes-Gibbs (1986) demonstrated this empirically. Pairs who repeatedly described abstract tangram figures developed increasingly compressed referring expressions across rounds — from multi-sentence descriptions to single-word labels — as shared conventions accumulated and $|c|_{\mathcal{L}^j}$ decreased. Teaching makes the same alignment deliberate: the teacher estimates the gap between the learner’s current $\mathcal{L}^j$ and the representational state required for competence, then designs a sequence of interventions whose effects on $\mathcal{L}^j$ shape what subsequent interventions can assume, making teaching a sequential control problem over the listener’s representational trajectory³¹.

This makes communication an inter-agent version of the curation problem from Section 3.4.1. In single-agent curation, the agent evaluates its own representation using that same representation. In communication, the speaker reasons about the listener’s curation from the outside: each utterance both probes and modifies $\mathcal{L}^j$, and the speaker must model how the listener will integrate it with existing structure. This is the final extension of the chapter’s general logic. Latent inference was first applied to individual behavior (Section 3.1), then to acting agents (Section 3.2), then to explicit representational construction (Section 3.3), and then to curated predictive structure (Section 3.4); in the multi-agent setting, the

30: This is the formal analogue of Clark and Brennan (1991)’s common ground: the resources both agents possess and can reference without further explanation are the intersection $\mathcal{L}^i \cap \mathcal{L}^j$, and building common ground is extending that intersection through interaction.

31: The Clark and Wilkes-Gibbs paradigm provides direct behavioral evidence for Equation (3.35). The initial description length is long because the listener’s library lacks the relevant entry. Once the entry is established, subsequent references are short. The transition from long to short descriptions tracks the construction of a shared library entry—communication as representational alignment.

same logic applies once more, with another agent’s representation as both the target of inference and the target of action. What remains is the unifying principle constraining which representational organizations are worth building, maintaining, and transmitting under finite resources.

3.6 Compression as the evaluative principle

Every method developed in this chapter weighs the complexity of an internal representation against the loss the agent suffers by acting on that representation rather than on the source it summarizes. Rate-distortion (Berger, 1971; Shannon, 1959) is the formal vocabulary that makes those differing trade-offs commensurable, and we use it here to place the chapter’s methods on a single (rate, distortion) plane.

The placement matters because it turns the principle chapter’s argument that resource rationality is the normative frame for representational management (Section 2.3.2) into an evaluative criterion. Construction, curation, and communication can each be read on the same axis, and the empirical chapters that follow can each be checked for what they buy at a given representational rate.

However, the unification has a limit. Rate-distortion is computable on the data the agent has already seen, while the value of a representation is settled in the future $\mathcal{E}$ the agent has not yet sampled, and no compression-internal device closes the gap. Section 3.6.4 develops this and the related structural limits.

3.6.1 Rate and distortion as the analytical axes

Let X denote the source the agent must represent (observations, tasks, trajectories, or another agent’s behavior) and Z the agent’s internal representation of it. The *rate* measures the complexity of the representation,

$$R = \text{MI}(X; Z), \quad (3.36)$$

which for discrete libraries reduces to $|\mathcal{L}|$, for variational latents to a KL between encoder and prior, and for transmitted messages to a coding cost $C(u)$. The *distortion* measures the expected task-relevant loss the agent incurs by acting on Z rather than X ,

$$D = \mathbb{E}[d(X, \hat{X}(Z))], \quad (3.37)$$

where the loss function d is determined by what the representation is for. The rate-distortion function $R(D)$ is the minimum rate achievable at distortion level D , and every representational system sits on or above this curve³². Better representations move closer to $R(D)$, retaining less information about the source while incurring no additional task loss.

The distortion is what makes the criterion task-faithful. The methods of the chapter differ less in how they measure rate than in what they take distortion to be. For program induction, the task is data fit, so the natural distortion is the negative log-likelihood of the observed data and the rate is description length $|\rho|_L$ in language L . Dreamer (Hafner et al., 2023) replaces this with a sum of reconstruction and reward-prediction losses, paired with a rate term that is the KL between encoder and transition prior in Equation (3.25). A pragmatic speaker (Frank and Goodman, 2012; Goodman and Frank, 2016) replaces it again with the listener’s expected

32: The information bottleneck (Tishby et al., 2000) expresses the same trade-off as a Lagrangian, $\min_{p(z|x)} \text{MI}(X; Z) - \beta \text{MI}(Z; Y)$, where Y is the task-relevant variable. Different values of β trace out different points on $R(D)$, and the choice of β is the agent’s resource bound.

competence loss $-\mathbb{E}[V(z^j; w)]$ in Equation (3.34), and the rate becomes the message cost $C(u)$. Choosing d is a substantive commitment about what the representation is for rather than a technical detail. A representation optimal under reconstruction distortion preserves observable detail but may be useless for control. One optimal under reward distortion compresses aggressively along task-relevant axes and may fail on tasks that load on currently irrelevant distinctions. The same source admits different optimal compressions under different distortion functions, and the distortion function is determined by $\mathcal{E}$. This is the formal version of the principle chapter's claim that representational value is $\mathcal{E}$ -dependent (Section 2.3).

3.6.2 MDL as the corner where distortion is log-loss

When the task is to encode the data and nothing else, distortion collapses into log-loss, $d(x, \hat{x}) = -\log p(\hat{x})$, and the rate-distortion trade-off becomes a single sum,

$$\mathcal{J}(\text{model, data}) = \underbrace{\text{model complexity}}_{\text{rate}} + \underbrace{\text{negative log-likelihood}}_{\text{log-loss distortion}}. \quad (3.38)$$

This is the minimum description length principle (Grünwald, 2007; Rissanen, 1986). For program induction, where the data the agent must explain is the source it represents, the corner is the right one and Equation (3.38) is the agent's stated objective. For Dreamer and for communication, the corner is a re-description. The Dreamer ELBO admits a coding-theoretic reading via the bits-back argument (Hinton and Van Camp, 1993), and $C(u)$ admits one through standard source coding, but in neither case is description length the agent's objective. It is our way of placing the system on the rate-distortion plane.

Stochastic, not MAP. Even within program induction, the criterion is not the single shortest program. The Bayesian formulation in Equation (3.19) maintains a posterior over programs, and the rate that matters is the posterior-expected description length rather than the length at the MAP,

$$\bar{R}(\mathcal{D}) = \mathbb{E}_{p(\rho|\mathcal{D})}[|\rho|_L]. \quad (3.39)$$

Two-part stochastic coding makes this explicit (Grünwald, 2007; Wallace, Boulton, et al., 1968). The agent commits to a distribution over models and pays an expected code length under that distribution, which is closer to the variational free energy than to a hard argmin. Bayesian model averaging is the rule rather than the exception. Representations that are close to optimal under several short programs are preferable to those that are uniquely optimal under a single one, because the former are more robust to small changes in $\mathcal{E}$. Reading Equation (3.38) through the argmin alone obscures this and overstates how decisively any single program is selected.

Cognitive evidence reads cleanly only when both axes are stated. The cognitive case for compression as a representational principle requires both rate and distortion to do its work. Rosch et al. (1976) showed that basic-level categories (e.g. "dog," "chair," "car") maximize the ratio of within-category similarity to between-category similarity. The numerator is a rate term, since high within-category similarity lets a single label compress many instances. The denominator is a distortion term, since low

33: Zipf’s law (Zipf, 2016) can be derived from optimal coding under a constraint on communicative effort (Cancho and Solé, 2003). Frequent concepts receive short codes and rare concepts long ones, but only because the distortion function (effort under communicative demand) is fixed by the task; under a different distortion function, the optimal codebook would be different.

between-category similarity preserves the discriminations the category is for. Either term alone misreads the result. A pure-rate criterion would collapse all categories into one. A pure-distortion criterion would refuse to label anything. Feldman (2000) found that the subjective difficulty of a Boolean concept tracks its Boolean complexity, the length of the shortest defining formula. Treated as a pure-rate result, this would imply that learners cannot acquire highly compressible-yet-subtle concepts. The honest reading conditions on distortion. At the level of distortion required for accurate categorization, description length is what predicts difficulty³³.

3.6.3 Dynamics of the coding scheme

Rate-distortion is a static criterion, but the lifecycle we study is the dynamics by which the coding scheme itself moves through experience. Each lifecycle stage destabilizes a different part of the rate-distortion trade-off, and the three following paragraphs work through them in turn.

Construction grows the codebook. Section 3.3 introduced library induction in its iterative form, $\mathcal{L}_{t+1} = f(\mathcal{L}_t, \{\rho_{i,t}^*\})$ in Equation (3.22), and the compression view recovers what the iteration is doing. Adding an entry ℓ adds a codeword; programs that previously required many primitives become expressible as a single invocation, and their description length drops. A static joint optimum over codebook and encodings,

$$\mathcal{L}^*, \{\rho_i^*\} = \arg \min_{\mathcal{L}, \{\rho_i\}} |\mathcal{L}| + \sum_{i=1}^N (|\rho_i|_{L_0 \cup \mathcal{L}} - \log p(\mathcal{D}_i | \rho_i)), \quad (3.40)$$

is a useful idealization but flattens the lifecycle. At step t the agent has only solved tasks $1, \dots, t$, the codebook $\mathcal{L}_t$ was built from those solutions, and the encoding of task $t+1$ is constrained by what $\mathcal{L}_t$ already makes short, so alphabet and messages co-evolve and the order of arrival matters (Zhou et al., 2024b). Equation (3.40) is best read as the limit object the sequential process approximates rather than as an optimization the agent performs. The same agent encountering the same tasks in a different order produces a different codebook, makes different programs short, and finds different downstream tasks easy to express. This is the precise computational sense in which construction is path-dependent (Section 2.3.3).

Curation moves the distortion function. Even with $\mathcal{L}$ held fixed, a representation can drift off the rate-distortion frontier because $\mathcal{E}$ has shifted and the distortion function d has moved under it. An entry that compressed a recurring pattern in one task distribution may compress nothing of value in another, and the rate-distortion score of every program that invokes it changes although nothing in the codebook has changed. Compression alone cannot detect this drift, since by construction it scores entries against the current data the agent has access to. If d has shifted but no new task has yet revealed the shift, every entry still looks well-placed. Curation is the lifecycle response to this asymmetry. It must rely on signals other than current-data fit to decide which entries continue to earn their place, which motivates the intrinsic-motivation criteria developed in Part III.

Communication couples two codebooks. In communication, the relevant codebook is no longer the speaker’s. The speaker has $\mathcal{L}^i$, the listener has $\mathcal{L}^j$, and the rate the trade-off depends on is the message length $|c|_{\mathcal{L}^j}$

under the receiver's alphabet (Equation (3.35)), not $|c|_{\mathcal{L}^i}$ under the speaker's. A concept that is short under $\mathcal{L}^i$ may be long, or inexpressible, under $\mathcal{L}^j$, and a speaker who optimizes against their own codebook will produce messages that look efficient to themselves and opaque to the listener. The shared alphabet is the intersection $\mathcal{L}^i \cap \mathcal{L}^j$, and the agents can either restrict the message to that intersection or extend the listener's codebook through teaching, both of which are themselves coding-scheme operations. The communication chapters develop the case where the speaker is uncertain about $\mathcal{L}^j$ and must reason about it as a latent.

3.6.4 What compression does not settle

Rate-distortion provides an evaluative criterion against which all three lifecycle operations can be placed on a common plane. It does not provide the procedures by which any of them is carried out, and four structural limits remain.

Mechanism. Compression tells us what a good representation looks like at a given (rate, distortion) point but not how to find one. The procedures developed earlier in the chapter remain necessary for that, with program induction handling discrete, compositional compressions, gradient-based world-model learning handling continuous and implicit ones, and pragmatic inference handling compressions that have to be transmitted between agents. The criterion tells us what those procedures are optimizing. It does not replace them.

Self-reference. Evaluating a candidate representation requires the agent's current representation. Rate is computed relative to the encoder the agent already has, and distortion is estimated through the model the agent already runs. The bootstrap problem flagged in Section 2.3.2 reappears here unchanged. Improving $(\mathcal{R}, \mathcal{F})$ is a computation that has to be carried out by the current $\mathcal{F}$ acting over the current $\mathcal{R}$.

Cost of evaluation. None of the rate terms is free to evaluate. Program length $|\rho|_L$ requires search through programs to assign, the ELBO requires forward and backward passes through the learned model, and $|c|_{\mathcal{L}^j}$ requires a model of the listener's library before any utterance can even be scored. Evaluation consumes the same finite budget the criterion is meant to allocate, which collapses the comfortable picture in which an outside scoring rule sits above the agent's resource bound. A resource-rational treatment must include the cost of the evaluator alongside the cost of what is evaluated³⁴.

Retrospective vs. prospective distortion. The most consequential limit is that the distortion the criterion would need is over the future task distribution, while the agent has access only to past tasks. The pragmatist value function in Equation (2.3) is an expectation over $\mathcal{E}$, whereas every estimator implementable in this section is computed over $\mathcal{D}_{\text{seen}}$. The gap between them is exactly the non-locality, undecidability, and self-reference triad of Section 2.3.2. A representation that lies on the rate-distortion frontier for the data observed so far need not lie on it for the next task the agent encounters, and no compression-internal device closes this gap.

The empirical chapters take up these limits in different settings. The construct chapters operate inside the lifecycle that path-dependent construction makes inevitable, the curate chapters build proxies for the prospective distortion the criterion alone cannot supply, and the communicate chapter relies on the listener's downstream behavior as the realized distortion that closes the evaluation loop. What unifies them

34: Meta-reasoning research (Gershman et al., 2015; Russell and Wefald, 1991) formalizes this by treating the choice of how much to compute as itself a decision under a value-of-computation criterion. Applied here, the rate-distortion score of a candidate representation has to be discounted by the cost of computing it.

Remark 3.3 (Compression is constrained by use) A representational resource is valuable when it simplifies future cognition without removing the distinctions that later prediction, planning, or communication require. Simplicity in isolation is not the criterion.

is rate-distortion as the common axis along which we read their representational decisions, while each agent retains its own task-relevant objective.

3.7 Summary

This chapter assembled the formal tools for making the library lifecycle tractable. Its common logic is latent inference over generative models of agents: posit hidden structure, specify how it generates behavior, and recover it from data. What varies is the latent format and the inference problem it induces. The agent’s internal world model—reward, representation, value, and policy—was decomposed into experimentally isolable components (Section 3.2). Programs and libraries made construction a problem of hypothesis-space change (Section 3.3). Learned world models exposed the ambiguity between model error and representation error (Section 3.4). Communication extended the same apparatus to inter-agent settings where the target is another mind’s representational state (Section 3.5). Finally, compression supplied the unifying evaluative principle: the description-length prior, ELBO, and communicative utility all instantiate a trade-off between model complexity and data misfit (Section 3.6).

The preceding chapter argued that representational dynamics decompose into construction, curation, and communication, and that the coupled dynamics of $(\mathcal{R}, \mathcal{F})$ under resource constraints make this decomposition necessary. This chapter operationalized that claim. Program induction and library learning provide a mechanism for construction: reusable subroutines compress recurring structure, while discovery order creates path dependence because coding schemes and evaluated data co-evolve. World-model learning provides the setting for curation: representation and prediction are coupled, yet each curation signal is computed through the representation it evaluates, so standard signals do not measure representation directly. Pragmatic inference and decision-theoretic communication provide the framework for sharing structure across agents, where communicative cost depends on the target concept’s description length under the listener’s library. These tools are linked by a common compression criterion that places construction, curation, and communication on the same scale.

Formal machinery specifies *how* to perform inference, not *what happens when it is applied*. The tools introduced here make three empirical questions precise. First, *construction*: when agents build libraries from real-world data, do the resulting structures reflect predicted prerequisite relations, cognitive traits, and path-dependent curriculum effects? Chapter 4 examines path dependence in program induction, while Chapter 5 addresses this with structured probabilistic inference on educational data. Second, *curation*: do intrinsic motivation signals shape different representational repertoires, and can feedback between motivation and representation produce pathologies that no single exploration signal resolves? Chapter 6 studies this in simulated agents, and Chapter 7 proposes an internalized empowerment criterion for library organization. Third, *communication*: when a knowledgeable agent explains a procedure to a partially informed listener, does decision-theoretic utility—measured by downstream action capability rather than belief accuracy—predict explanatory success? Chapter 8 tests this in a partially observable navigation domain where

representational alignment is central. With this vocabulary in hand, the thesis turns to these questions.

Part II.

Construct: how structured knowledge is
acquired

Learning leaves traces: path dependence in human knowledge

4.

“The more constraints one imposes, the more one frees oneself. And the arbitrariness of the constraint serves only to obtain precision of execution.”

— Igor Stravinsky, *Poetics of Music in the Form of Six Lessons*, 1947

The background chapters unified a vocabulary for representational dynamics—the construction, maintenance, and sharing of reusable internal structure. A natural formalism for such structure is the *program*: compositional, hierarchically organized, and amenable to compression through subroutine reuse. This chapter asks whether program induction under resource constraints can serve as a working model of how humans build representational infrastructure, and whether the behavioral signatures of that process are detectable.

We develop a unified framework that models human sequence learning as *resource-rational program induction*: learning under two distinct but interacting constraints, a memory constraint (description length) and a computational constraint (bounded search and backtracking). We instantiate this framework in a hierarchy of five progressively richer models and evaluate it in the domain of melodic sequence learning—a setting that exhibits hierarchical structure and combinatorial richness, yet remains constrained enough for controlled experimentation. The evaluation combines simulations, an online behavioral experiment ($n = 96$), and an exploratory curriculum analysis that connects learning order to representational outcomes.

The chapter operationalizes the *Construct* stage of the library lifecycle. Its core finding is that library construction is an ongoing, path-dependent compression dynamic: what a learner has already built determines what can be efficiently represented and searched for next. Across all behavioral measures, learners exhibit the signatures of a system that compresses experience into reusable representational infrastructure—precisely the process that the background chapters formalized.

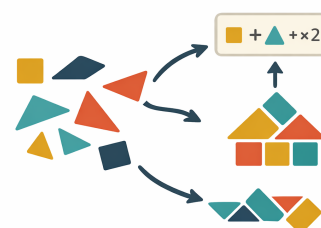
The chapter is organized as follows. [Section 4.1](#) motivates the problem and previews the main results. [Section 4.2](#) reviews the relevant literature on program induction, rate–distortion theory, library learning, and melodic sequence learning. [Section 4.3](#) formalizes the program representation, resource constraints, model hierarchy, and inference procedure. [Section 4.4](#) reports simulation results on compression, learning dynamics, and resource efficiency. [Section 4.5](#) presents the behavioral experiment and its analysis. [Section 4.6](#) develops the exploratory curriculum analysis. [Section 4.7](#) discusses the findings and their limitations.

4.1 Introduction

When a musician learns a new piece, they discover patterns—ascending runs, repeated motifs, rhythmic cycles—and encode the piece in terms of those patterns. A novice who has learned only scales represents a melody as a sequence of scalar movements; an experienced musician who has also internalized arpeggios, chord progressions, and common ornaments can

4.1 Introduction	57
4.2 Background	59
4.3 Methods	62
4.4 Simulations	66
4.5 Behavioral experiment	68
4.6 Curriculum effects	76
4.7 Discussion	78
4.8 Summary	80

Code and experiments available at the
Github repository [chirp](#)



represent the same melody far more compactly. The musician’s internal library of reusable patterns determines which regularities are visible, which generalizations are cheap, and which novel passages can be parsed on first encounter.

How are such libraries acquired? Program induction models (Goodman et al., 2008; Lake et al., 2015; Piantadosi et al., 2012; Rule et al., 2020) formalize the idea as search over compositional hypothesis spaces for programs that generate observed data, with reuse made explicit through library learning (Ellis et al., 2021; Zhao et al., 2023). However, program-based accounts of learning naturally raise the question: how are such representations shaped by the resource limitations under which humans operate? The principle of resource rationality formalizes cognition as optimization under constraints on time, memory, and computation, identifying the best achievable strategy given those limits (Lieder and Griffiths, 2020; Simon, 1955).

A key formal tool for analyzing a subset of these trade-offs is *Rate-Distortion Theory* (RDT) (Figure 4.2), which formalizes the balance between expected reconstruction error (distortion) and informational cost (rate) (Berger, 1971; Gershman, 2021; Shannon, 1959). Although RDT has been applied to efficient perception (Sims, 2016), memory (Bates and Jacobs, 2020; Nagy et al., 2020), and decision-making (Bhui and Gershman, 2018; Lai and Gershman, 2021), it typically treats representations as fixed and not learned over time. Thus, despite some prior work formalizing memory costs using RDT (Gershman, 2021), the computational cost of finding good representations has received comparatively little systematic treatment. As a result, RDT in its standard form provides a limited account of learning efficiency, by failing to capture the computational cost required to infer over structured hypotheses, and how these costs dynamically shift during the learning process (Nagy et al., 2025a).

These limitations are central for program induction. If learners build a library of reusable programs over time, then both the computational and representational costs change with experience: earlier choices shape what can be efficiently inferred and represented later (Dekker et al., 2022; Zhao et al., 2023). Thus, understanding resource-rational program induction requires extending classical RDT to incorporate (i) the computational costs of inference, and (ii) path-dependent learning dynamics that support efficiency.

Remark 4.1 (Path dependence across fields) The idea that current learning depends on prior learning appears across many traditions under different names. In psychology it is often discussed in terms of *learning history*, *set*, or *transfer*; in education as the effects of *curriculum sequence*, *scaffolding*, or *prerequisite structure*; and in other fields as path dependence or hysteresis. The common theme is that earlier experience changes the space of representations or actions available later.

This chapter extends RDT by incorporating the computational cost of inference over structured program hypotheses and the path-dependent dynamics of learning from a growing library of reusable programs. On this account, learners trade off distortion (how much error is incurred) against rate (how efficiently structure can be stored and reused). Simplicity is favored by two interacting constraints: a computational constraint on proposing candidate programs, and a memory constraint favoring compact representations among those candidates. Central to this framework is the idea of a *library* (Section 2.4)—a repertoire of reusable programs discovered and accumulated over time. Libraries help mitigate both constraints. Stored programs can be referenced by a compact pointer rather than spelled out in full, reducing the description length (memory). Simultaneously, stored programs serve as high-quality proposals during inference, reducing the number of candidates that need to be considered (computation). We further distinguish between a *local library*, built within a task to capture context-specific regularities, and a *global library*, which accumulates abstractions that generalize across tasks, together allowing hierarchical reuse of structure across multiple timescales.

We evaluate this framework through three analyses. In *simulations*, hierarchical library learning achieves better rate-distortion trade-offs and stronger one-shot generalization than fixed grammars or surface-level chunking models, while revealing structured interactions between memory and computation. In a *behavioral experiment* ($n = 96$), recall errors, reaction times, and free compositions all show signatures of program-like compression: errors simplify toward shorter programs, reaction times increase at model-predicted program boundaries, and free compositions are best explained by libraries inferred from participants' learning histories. Across measures, our model with participant-specific libraries provides the best account of behavior. Finally, an *exploratory curriculum analysis* shows that presentation order systematically shapes library quality, and that information-theoretic principles from partial information decomposition can guide curricula that induce more effective libraries and better generalization.

4.2 Background

This section reviews four research threads that converge in this chapter: program induction as a model of concept learning, resource rationality and rate-distortion theory as normative accounts of cognitive constraints, library learning as a mechanism for acquiring reusable structure, and sequence learning as an empirical testbed for these ideas.

4.2.1 Program induction and the Language of Thought

The *Language of Thought* hypothesis (Fodor, 1975) holds that mental representations are compositional, productive, and systematic—properties naturally captured by programming languages. These properties suggest that learning can be understood as constructing internal programs: structured symbolic expressions that generate observed data when executed. *Program induction* formalizes this idea as search over structured hypothesis spaces for programs that best explain observations (Gold, 1967; Goodman et al., 2008; Piantadosi et al., 2012; Solomonoff, 1964). The framework has been applied to model one-shot concept acquisition (Lake et al., 2015), rule learning (Rule et al., 2024, 2020), number concepts (Piantadosi et al., 2012), language (Ellis et al., 2022; Piantadosi et al., 2008; Piantadosi and Jacobs, 2016), planning (Correa et al., 2024; Sharma et al., 2021), and abstract reasoning (Dehaene et al., 2022; Zhao et al., 2023).

A central question in program induction is how the space of candidate programs is defined. The dominant approach draws on formal grammars, originating with Chomsky (1956)'s generative grammars for natural language. *Probabilistic context-free grammars* (PCFGs) assign probabilities to production rules, enabling Bayesian inference over parse trees (Chater et al., 2006; Perfors et al., 2011). PCFGs provide a well-defined prior over programs and support compositional structure, but their rule probabilities are fixed: each observation is parsed independently under the same grammar, with no mechanism for adapting knowledge from experiences. *Adaptor grammars* (Johnson et al., 2006) extend PCFGs by adding a nonparametric cache—governed by a Pitman-Yor process (Teh, 2006)—that stores frequently used derivations and makes them available for reuse. This mechanism has been applied to word segmentation (Johnson et al., 2006), morphological learning (O'Donnell, 2015), and concept learning (Liang et al., 2010; Zhao et al., 2023).

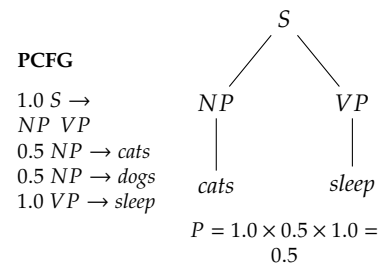


Figure 4.1: A PCFG and a sampled parse tree. Production rules (left) assign probabilities to each expansion. The probability of a parse tree (right) is the product of the rules used.

Table 4.1: Contrasting bounded and resource-rational accounts of cognition under limited memory, time, and computation.

Framework	Definition	Focus	RQ
Bounded rationality	Agents satisfice rather than optimize because cognitive resources are limited (Gigerenzer, 2004; Simon, 1955).	Heuristics used when full optimization is infeasible.	“what would an ideal agent do?”
Resource rationality	Behavior is evaluated relative to the optimal policy under explicit resource constraints (Griffiths et al., 2015; Lieder and Griffiths, 2020).	Normative optimality under constraints.	“what is the best an agent can do given its limits?”

Program-based representations naturally capture compositionality, systematic generalization, and abstract rule structure. But these advantages do not by themselves explain how experience reshapes the hypothesis space over time. In particular, efficient *reuse* of previously discovered components requires more than a compositional representational format: it requires a *library* of reusable abstractions that can be retrieved and recombined in new inferences. Explaining how such libraries are acquired and updated is also a challenge, and the focus of Section 4.2.3.

4.2.2 Resource rationality and rate-distortion theory

Human cognition operates under severe constraints on memory, time, and computation (See Table 4.1 and Section 3.6). A central formal tool for characterizing one aspect of the constraints, memory, is *rate-distortion theory* (RDT), which specifies the optimal trade-off between the fidelity of a representation (distortion) and its informational cost (rate) (Berger, 1971; Shannon, 1959). RDT has provided normative explanations for human perception (e.g., explaining why color categories cluster where they do; Bates et al., 2019; Sims, 2016), memory (e.g., predicting which details are forgotten and which are retained; Bates and Jacobs, 2020; Gershman, 2021; Nagy et al., 2020), and decision-making (e.g., accounting for context effects through information-theoretic costs; Bhui and Gershman, 2018; Lai and Gershman, 2021).

Remark 4.2 (Rate-distortion in a nutshell) An encoder maps observations X to compressed representations Z ; a decoder reconstructs $\hat{X}$ from Z . Rate $R = \text{MI}(X; Z)$ measures how much information the representation retains. Distortion $D = \mathbb{E}[d(X, \hat{X})]$ measures reconstruction error. The RD function $R(D)$ gives the minimum rate needed to achieve distortion at most D —the fundamental limit of lossy compression.

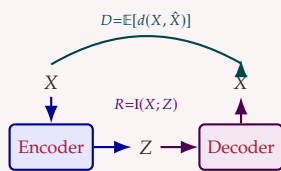


Figure 4.2: The rate-distortion framework. An encoder-decoder pair compresses observations X into representations Z and reconstructs $\hat{X}$. Rate measures the information retained; distortion measures reconstruction error.

However, standard RDT has two limitations as a model of human learning under resource constraints. First, the classical optimization problem assumes a *fixed* source data distribution and a fixed coding scheme by seeking an optimal encoder-decoder pair. If mental representations are programs, then rate corresponds to *description length* (Chater and Vitányi, 2003; Feldman, 2000; Sablé-Meyer et al., 2022; Sow and Eleftheriadis, 2003; Vereshchagin and Vitányi, 2005): the number of bits required to specify a program under a given representational language. When the language changes over learning—for instance, by introducing a new shorthand for a pattern that previously had to be written out in full—the rate of the same program changes as well. Second, standard RDT does not account for the computational cost of discovering such representations. The rate-distortion function is defined as an infimum over all encodings achieving a target distortion level, thereby abstracting away from the search required to find them. But in human cognition, search over candidate programs can be prohibitively expensive. Under bounded time and attention, exploring a combinatorially large hypothesis space is itself a major constraint (Zhou et al., 2024b).

4.2.3 Library learning

Library learning addresses the question of how reusable representational structure accumulates over time, and in doing so provides a mechanism that addresses both limitations of standard RDT identified above. The core idea is that learners maintain a *library*—a growing collection of previously inferred programs that can be retrieved and composed into new programs. A library helps with memory: stored programs can be referenced by a compact pointer rather than spelled out in full, reducing the effective description length of new programs that reuse them. A library also helps with computation: stored programs serve as high-quality proposals during search, reducing the number of candidates that need to be evaluated from scratch.

DreamCoder (Ellis et al., 2021) provides a normative example for library learning using neural networks. It alternates between a *wake* phase (searching for programs that explain observed data) and a *sleep* phase (extracting common programs and adding them to a library)¹. Over training, the learned library becomes more powerful, enabling programs of growing complexity to be expressed and discovered efficiently. Meanwhile, Zhao et al. (2023) applied a library-learning mechanism to model human concept learning descriptively and showed that the resulting libraries are sensitive to learning order. This *curriculum sensitivity* is a general property of incremental library learning under bounded resources. Early examples influence which programs are first reused and assigned compact descriptions; once those representations become cheap and available as building blocks, they bias subsequent search and make some regularities easier to discover than others (Bruner, 1973; Dekker et al., 2022; Ritter et al., 2007). The present chapter formalizes this path dependence within a rate-distortion framework, making explicit the interaction among library induction, memory cost, and computational cost.

Remark 4.3 (Entropy coding) The intuition is the same as in entropy coding: to minimize expected code length, frequent items receive shorter descriptions. For example, in a Huffman code, a high-probability symbol may be assigned a 1-bit codeword, whereas a low-probability symbol may require several bits. Similarly, frequently reused programs can be assigned short pointers, making larger programs that invoke them cheaper to encode on average.

1: The wake-sleep algorithm (Hinton et al., 1995) was originally introduced for latent-variable generative models, alternating between inferring latent variables (wake) and updating the generative model (sleep).

4.2.4 Sequence learning

Sequential data—speech, music, motor actions, symbolic strings—have long been a testbed for theories of rule learning and compression: they are experimentally controllable yet rich enough to express complex structure. Three broad traditions have studied human sequence learning, each emphasizing a different mechanism.

The first tradition, *statistical learning*, studies how learners extract transitional probabilities from sequential input. Saffran et al. (1996) showed that infants can segment continuous speech from such statistics alone², with later work extending this to tonal (Loui et al., 2010; Saffran et al., 1999), visual (Fiser and Aslin, 2002), and cross-modal sequences (Conway et al., 2010). These models capture sensitivity to distributional regularities, but often reduce structure to pairwise or n -gram statistics, leaving open how learners acquire compositional representations over longer spans.

The second tradition, *serial recall and chunking*, studies how sequences are grouped and stored in memory. Lashley et al. (1951) argued that sequential behavior requires hierarchical planning; later work formalized chunking as a memory strategy (Mathy and Feldman, 2012; Miller, 1956), identified chunk boundaries through reaction times in motor sequences (Rosenbaum et al., 2007; Sakai et al., 2003; Verwey and Dronkert, 1996), and modeled serial recall errors via position-item binding³ (Botvinick and Plaut, 2006; Henson, 1998). These approaches explain how sequences are grouped and retrieved under memory limits, but they typically treat

2: In the original experiment, trisyllabic nonsense words (e.g., *pabiku*, *tibudo*) were concatenated into a continuous stream with no pauses. Within-word transitional probabilities were high (e.g., $P(\text{bi} | \text{pa}) = 1.0$) while between-word probabilities were low (e.g., $P(\text{ti} | \text{ku}) = 0.33$). Eight-month-old infants discriminated words from non-words after only two minutes of exposure, demonstrating sensitivity to these distributional cues.

3: Each item is associated with its ordinal position in the sequence; recall errors such as transpositions arise when these bindings decay or interfere (Henson, 1998).

4: Unlike chunks, which are memorized surface sequences retrieved as units, grammars are abstract rules *applied* to generate or classify novel sequences. Chunking compresses recurring forms; grammar learning encodes structure at the category level, independent of surface realization.

chunks as stored verbatim units rather than asking how new structures are constructed from experience.

The third tradition studies *grammar and rule learning*⁴: the acquisition of abstract structure from sequential data. Artificial grammar learning experiments (Pothos, 2007; Reber, 1967) showed that participants can learn to classify novel sequences as grammatical after exposure, and formal models have characterized this capacity using probabilistic grammars (Johnson et al., 2006; O'Donnell, 2015; Perfors et al., 2011) and program induction (Lake et al., 2015; Rule et al., 2020). Grammar-based approaches capture hierarchical and recursive structure, but they typically assume a fixed grammar and do not model how the grammar changes with experience.

Across these traditions, representational vocabularies (statistical learning, standard chunking) or grammars (artificial grammar learning) are typically fixed, and models are often tested on short artificial stimuli. Less is known about how structured representations are constructed and updated from naturalistic sequences under resource constraints. We address this gap in melodic sequence learning. Music is a natural testbed because melodies show cross-cultural regularities (Jacoby et al., 2024; Mehr et al., 2019), hierarchical and recursive structure (Fitch and Martins, 2014; Sievers et al., 2013), and combinatorial richness (Harasim, 2020; Wiggins, 2007), while remaining experimentally controllable.

4.3 Methods

Our model HAG (Hierarchical Adaptor Grammar) formalize sequence learning as inference over generative programs, supported by building a *library* of reusable components. Learning is governed by two interacting constraints: a *memory* constraint that penalizes representational cost of the inferred program (via description length) and a *computational* constraint that bounds the search and backtracking used to identify and revise candidate programs. We first describe the program representation (Section 4.3.1), inference under dual constraints (Section 4.3.2), the model hierarchy (Section 4.3.3), and the dataset (Section 4.3.4).

Formally, to find an efficient representation for an observed sequence $X = [x_1, x_2, \dots, x_T]$, HAG uses two coupled processes, *program inference* and *library updating*. First, HAG searches for a generative program Π such that executing it produces a reconstruction $\hat{X} = g(\Pi)$ of the observed sequence X , where g denote the interpreter that maps a program to its reconstruction. For later use, it is helpful to view a program Π as a concatenation (denoted as $\oplus$) of *subprograms* $\Pi := \pi_1 \oplus \pi_2 \oplus \dots$, which are self-contained, executable subtrees that can be nested or concatenated to produce a subsequence. The reconstructed sequence is thus composed of several subsequences $\hat{X} = [X_1, X_2, \dots]$ where $\hat{X}_i = g(\pi_i)$. After subprograms are selected, HAG updates two libraries: a local library $\mathcal{L}_{\text{loc}}$ that is built within the current sequence, and a global library $\mathcal{L}_{\text{glob}}$ that is updated across sequences to support transfer learning.

4.3.1 Program representation in combinatory logic

We represent programs in typed Combinatory Logic (CL; Liang et al., 2010; Schönfinkel, 1924), a minimal, variable-free formalism for

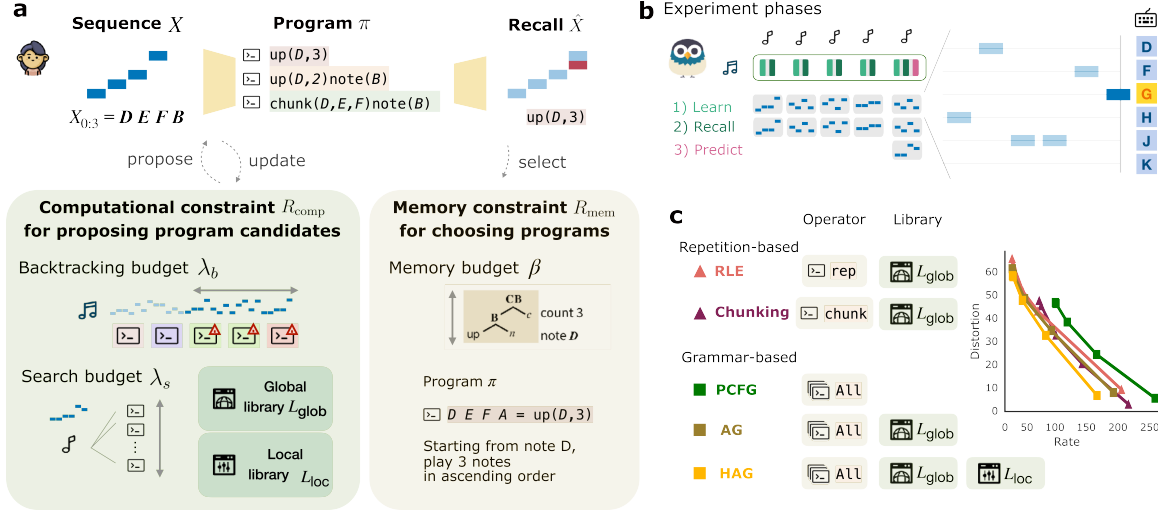


Figure 4.3: Overview of experimental paradigm and computational models. (a) Sequence learning is cast as inference over generative programs that construct observed sequences under two resource constraints. An observed sequence X (left) is explained by a generative program π . A *memory* constraint sets the representational budget for the *final* program (rate), controlled by β and favoring ever shorter programs at ever higher β . A *computational* constraint limits inference by bounding the number of candidate programs that can be proposed and revised, via a search budget N_s and a corrective backtracking budget N_b . Candidate programs draw on a global library L_{glob} and a local, participant-specific library L_{loc} , which are updated as structure is discovered. (b) Trial structure in the behavioural experiment. Participants (or the model) (1) learn a melodic sequence, (2) recall it via reproduction, and (3) continue it to probe generalization under the inferred library. (c) Model comparison and rate-distortion curve under simulation. We contrast repetition-based baselines (RLE, Chunking) and structure-based grammars (PCFG, AG) with hierarchical adaptive grammar learning (HAG). Icons indicate which primitive operators and library components each model uses (e.g., repetition, chunking, global/local libraries). Curves show the trade-off between representational rate and distortion.

composing functions. CL provides a small set of primitive operators—called *combinators*—that can be composed into binary trees, resulting in hierarchical expressions that generate sequences when executed⁵.

A program π in CL is a binary tree of the form $\pi = [r, \Gamma_l, \Gamma_r]$, where r is a *router* (a combinator that directs arguments to subtrees) and Γ_l, Γ_r are the left and right subtrees. Routers are composed from first-order combinators $\{B, C, S\}$:

- B directs the incoming variable first to the right subtree Γ_r , then passes the result to Γ_l .
- C directs the variable first to the left subtree Γ_l , then passes the result to Γ_r .
- S sends the variable to both subtrees simultaneously.

Higher-order routers (e.g., CB, BS) are finite sequences of these first-order combinators and enable complex argument-routing patterns. Any subtree is itself a complete subprogram that can be cached in the library, retrieved, and composed into new programs.

Domain-specific operators. In addition to the standard CL combinators, we include predefined melody-domain operators so that the same program language can express both general compositional structure and melodic regularities. Primitives are divided into base terms and function terms.

Base terms. These define the typed atomic values in the language:

- **Note** $n \in \{1, 2, 3, 4, 5, 6\}$ (type t_n): pitch values.
- **Count** $c \in \mathbb{N}^+$ (type t_c): repetition or step counts.
- **Time index** $m \in \mathbb{N}^+$ (type t_m): positions within a sequence.

Function terms. These define typed operations on base terms:

5: Typed CL is intentionally *not* Turing complete: the type system rules out non-terminating programs. Untyped CL with only S and K is Turing complete (Schönfinkel, 1924), but unrestricted search over such programs is undecidable. The type system trades expressiveness for tractable inference.

Example 4.1 (Router B) $B f g x := f(g x)$ — function composition. e.g. $B(\text{up})(\text{rep}(D))3$ first applies $\text{rep}(D)$ to get $[D, D, D]$, then passes it to up .

Remark 4.4 (CL vs. λ -calculus) CL is a variable-free variant of λ -calculus (Church, 1936): instead of named variables, combinators route arguments structurally. This makes shared substructure syntactically explicit—two programs performing the same computation have the same tree, regardless of naming conventions, which is essential for library deduplication (Liang et al., 2010).

6: A *type* labels what kind of object a term is and what it can combine with. Types prune the search space without evaluation: applying *up* (which expects t_n, t_c) to two counts is a type error, so that branch is rejected immediately (Ellis et al., 2021).

- ▶ $\text{up}(n, c)$: ascending sequence of c semitone steps starting from note n (e.g., $\text{up}(D, 2) = [D, E, F]$).
- ▶ $\text{down}(n, c)$: descending sequence of c semitone steps from note n .
- ▶ $\text{rep}(n, c)$: repeat note (or subsequence) n for c times (e.g., $\text{rep}(D, 3) = [D, D, D]$).
- ▶ $\text{range}(n, s)$: generate a stepwise sequence from start note n with step size s .
- ▶ $\text{reverse}(n)$: reverse the order of a sequence n .
- ▶ $\text{mem}(C)$: verbatim memorization of a specific note sequence C .

Type system. Typed CL assigns distinct types⁶ to the domain and codomain of each primitive and program, ensuring syntactic correctness. Despite sharing numerical values (e.g., piano note 1, repeat 1 time, the first index in an array), different base terms have different types reflecting their distinct semantic roles. The type of a function term is denoted with a right arrow: for example, the type of $\text{get}(n, m)$ is $t_n, t_m \rightarrow t_n$, indicating the types of its input arguments and output. This type system constrains the space of well-formed programs and makes search more

4.3.2 Program inference under dual constraints

Two constraints govern encoding and accumulation in program inference: *memory* limitations on the amount of structure that can be stored, and *computational* limitations on search over candidate programs and revision of earlier commitments. This extends the general resource-rational framework from Section 3.6.

Memory constraint. We first formalize *memory* costs (Gershman, 2021; Nagy et al., 2025a) as the representational complexity of a subprogram $R_{\text{mem}}(\pi)$. We use RDT (Shannon and Weaver, 1949; Sims, 2016; Zhou et al., 2024b) to characterize the goal of learning accurate representations (i.e., minimizing distortion) while also minimizing the complexity (i.e., rate). Thus, the selected subprogram minimizes the weighted sum of distortion and memory cost:

$$\mathcal{J}(\pi) = d(X, \hat{X} \mid \pi) + \beta R_{\text{mem}}(\pi \mid \mathcal{L}_{\text{glob}}, \mathcal{L}_{\text{loc}}). \quad (4.1)$$

Here, $d(\cdot, \cdot)$ is the reconstruction error defined by Hamming distance and β controls the trade-off between accuracy and complexity. Importantly, the memory cost R_{mem} depends on the learner’s current knowledge, i.e., the description length of π depends on what the learner can already express compactly using $\mathcal{L}_{\text{glob}}$ and $\mathcal{L}_{\text{loc}}$ ⁷. When a new sequence invokes a stored subprogram rather than re-deriving a new one, the description length is shorter. Thus, reuse helps reduce memory load through encoding recurring structure once and referencing it thereafter.

Computational constraint. In addition to the cost of storing a representation $R_{\text{mem}}(\pi)$, we also constrain the *computational* costs of discovering it $R_{\text{comp}}(\pi)$. Minimizing the objective function in Equation (4.1) requires searching a combinatorially vast space of possible subprograms. Thus we impose a hard limit on the inference process:

$$\min \mathcal{J}(\pi) \text{ subject to } R_{\text{comp}}(\pi; N_s, N_b) \leq C_{\text{limit}}. \quad (4.2)$$

The computational cost R_{comp} can be decomposed into two budgets (Figure 4.3a). The *search budget* N_s limits how many candidate subprograms the learner can consider, while the *backtracking budget* N_b limits how

7: e.g. if $\text{up}(D, 3)$ is cached in $\mathcal{L}_{\text{loc}}$, it costs one pointer; without it, the full tree must be specified, requiring several bits more. The same computation has different rates under different libraries.

far the learner can revise earlier commitments as new data arrives (see Methods)⁸.

Library updating. The local and global libraries define a structured prior over candidate subprograms: the learner can choose between reusing elements from $\mathcal{L}_{\text{loc}}$, reusing elements from $\mathcal{L}_{\text{glob}}$, or proposing a novel subprogram from the base grammar. Within a sequence, each selected subprogram increases its count in $\mathcal{L}_{\text{loc}}$, promoting further reuse in that same sequence. After the sequence ends, these local counts partially propagate to $\mathcal{L}_{\text{glob}}$, such that reliably useful subprograms become more likely to be reused—and hence cheaper to be represented and selected—in future sequences. Formally, this two-level updating is implemented by a hierarchical Pitman-Yor process⁹ (Teh, 2006) (see Methods), yielding path-dependent compression and transfer.

$$p(\pi | \mathcal{L}_{\text{glob}}, \mathcal{L}_{\text{loc}}) = \underbrace{\alpha_{\text{loc}} p(\pi | \mathcal{L}_{\text{loc}})}_{\text{local reuse}} + \underbrace{\alpha_{\text{glob}} p(\pi | \mathcal{L}_{\text{glob}})}_{\text{global reuse}} + \underbrace{(1 - \alpha_{\text{loc}} - \alpha_{\text{glob}}) p(\pi | H)}_{\text{novel from grammar}} \quad (4.3)$$

where H is the base grammar prior. The weights α_{loc} and α_{glob} control the propensity to reuse cached structure versus proposing a novel program (implemented via hierarchical Pitman-Yor process parameters (Teh, 2006); see Methods). Together, the two constraints interact with library updating: a richer $\mathcal{L}_{\text{glob}}$ reduces R_{mem} by providing compact pointers for recurring subprograms, while simultaneously reducing R_{comp} by supplying high-quality proposals that focus search on already-validated regions of the program space.

4.3.3 Model hierarchy

To capture a spectrum of possible strategies, we consider models that differ in (i) the expressiveness of their program space and (ii) whether they support learning reusable structure (Section A.1.1).

First, tolerant Run-Length Encoding (RLE) captures immediate repetitions as the simplest form of regularity by compressing consecutive identical symbols (Golomb, 1966). Second, Chunking extends note-level repetition to chunk-level by storing contiguous substrings as reusable units (Fonollosa et al., 2015; Miller, 1956; Wu et al., 2023), even across sequences. However, they both remain surface-level mechanisms that memorize and repeat previously observed strings.

In contrast, grammar-based methods in our third and fourth models capture a different kind of structure via hierarchical composition. Early formal approaches such as Probabilistic Context-Free Grammars (PCFGs) (O'Donnell, 2015) to represent nested patterns through fixed production rules. While rule probabilities can in principle be estimated from data, there are often specified in advance and fixed (Figure 4.1). Thus a standard PCFG does not provide a built-in mechanism for discovering and caching new reusable structures during inference (i.e., each sequence is parsed independently under the same grammar). Fourth, Adaptor Grammars (AGs) extend fixed grammars by adding experience-dependent reuse on top of a grammar. Intuitively, AGs allow the model to cache frequently used structures (i.e., programs) into a library and reuse them (Liang et al., 2010; Zhao et al., 2023). However, a key assumption the same cache supports reuse across all contexts, neglecting the distinction between general purpose and locally relevant abstractions.

8: N_s is a *prospective* budget: how many candidates to evaluate before committing to a subprogram. N_b is a *retrospective* budget: how many already-committed boundaries may be revised as new evidence arrives.

9: The Pitman-Yor process generates power-law frequency distributions (Teh, 2006), consistent with Zipfian statistics observed in language and music corpora. Its discount parameter controls how strongly frequent subprograms dominate proposals; its concentration parameter controls the rate at which novel subprograms enter the library.

Our model **Hierarchical Adaptor Grammars (HAGs)** extends AGs across different timescales of reuse. It maintains both a *global* library for generic abstractions and a *local* library for short-term patterns. This hierarchical library lets HAG flexibly combine cross-task transfer with context-specific reuse, which fixed grammars (PCFG), surface-level repetition models (RLE/Chunking), and single-cache AGs cannot capture.

4.3.4 Melodic sequence dataset

We converted a dataset of real-world MIDI files (Garcia-Valencia et al., 2020) into structured piano-roll representations using the following pipeline: (1) *Instrument filtering*: we retained only files whose primary instruments are piano, brass, reed, or synth lead to retain melodic as opposed to ambient (e.g., synth pad) or rhythmic samples (e.g., drums); (2) *Pitch normalization*: we mapped notes to a single octave comprising 12 pitch classes, $\{C, C^\sharp, D, D^\sharp, E, F, F^\sharp, G, G^\sharp, A, A^\sharp, B\}$; (3) *Note constraint*: we only included in our stimulus set sequences containing exactly six unique pitch classes, such that participants could comfortably execute the sequences with both hands on their keyboard during the task; (4) *De-duplication*: we removed duplicate note sequences that resulted from pitch normalization; (5) *Length constraint*: We retained sequences of 80–120 notes, balancing enough material for segmentation against memory load. We then followed the dataset’s original training-evaluation split. After preprocessing, we were left with 298 sequences for training and 96 sequences for testing.

4.4 Simulations

We first report results from model simulations, which clarify how memory and computational limits jointly shape learning and compression. Models achieve better rate-distortion trade-offs when equipped with a hierarchical library, which accumulates and reuses recurring structure across sequences, with HAG outperforming all alternatives. This benefit is consistent across training and held-out generalization sets.

Simulation setup. We randomly sampled 50 melodic sequences from the training set. In all simulations, we treated the model as a plausible observer with access to complete sequences. Each simulation was repeated with 100 random seeds¹⁰, which varied both the presentation order and the stochastic choices involved in program selection. Results were averaged across seeds. Models were compared under matched resource settings where the parameter choices can be found in [Section A.3](#).

Compression performance

To characterize the RD trade-offs, we ran simulations in which each model inferred programs for melodies under matched resource settings, measuring both rate (description length) and distortion (Hamming distance). Models were fit on training melodies (allowing library updates) and then evaluated on held-out melodies to assess one-shot generalization.

Across training melodies, HAG achieved the most favorable RD trade-off: for a given rate, it had lower distortion than any alternative, with AG typically second-best. A paired-samples Wilcoxon test confirmed that HAG attains significantly lower distortion and rate than AG¹¹. AG in turn

10: Seeds vary two sources of stochasticity: (i) the order in which the 50 sequences are presented, which affects what the global library accumulates first; and (ii) the stochastic choices during program selection at each step. Averaging across both isolates structural effects of the model from presentation-order artefacts.

11: HAG vs. AG: $z(99) = 1183, p < .001, r = -.531, CI = [-.702, -.337]$

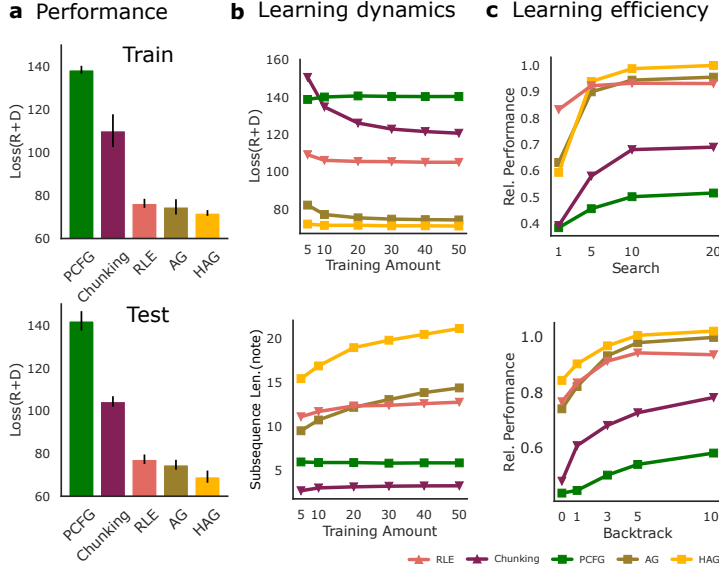


Figure 4.4: Simulation results for the melodic sequence learning task. We study how memory (rate), computation (search and backtracking), and accuracy (distortion) interact over the course of learning among different models. (a) Loss objectives (rate + distortion) for training sequences and one-shot generalization to held-out sequences using learned libraries. (b) Evolution of loss and subsequence length per subprogram as training data increases. (c) RD values as a function of computational resources, showing how search budget (N_s) and backtracking budget (N_b) affect model performance.

outperformed RLE¹², showing the benefit of richer primitives beyond simple repetition.

Under matched conditions on held-out melodies, HAG maintained its advantage over all other models¹³. These results suggest that generalization is mainly driven by the learned libraries¹⁴, since global and local libraries allow HAG to encode unfamiliar sequences with fewer resources. However, under very high memory constraint penalties ($\beta = 10.0$), all models converge to primarily minimizing memory costs at the expense of developing an effective global library, and HAG's advantage over AG diminishes (See Figure A.4).

Learning dynamics

A core property of library-augmented models is that learning improves with experience, because repeated exposure enables the accumulation and reuse of programs. All models with a library showed decreased training loss over time, whereas PCFG showed no improvement because it lacks a mechanism for cross-melody abstraction and must effectively relearn each melody from scratch. Although all library-based models improve with training, they do so via different representational strategies. Chunking improves rapidly early on through near-verbatim memory, resulting in low distortion but relatively high representational cost. As training progresses and the set of reusable substrings saturates, Chunking's advantage diminishes.

Beyond RD curves, a key prediction of library-based models is they can increasingly nest subprograms when constructing new ones. We quantify this via the average reconstruction length of subsequences generated by selected subprograms (Figure 4.4b, right). HAG subprograms consistently accounted for the longest subsequences under fixed computational budgets ($N_s = 10$, $N_b = 1$), followed by AG, while Chunking yielded substantially shorter subsequences. This pattern emerges most prominently when compression is weighted more strongly (i.e., β is large). However, when β is small (e.g., $\beta = 0.1$; Figure A.6), all models minimize distortion and therefore choose conservative subprograms that safely reconstruct shorter subsequences. In this regime HAG does not exceed AG or PCFG. But as β increases, HAG and AG increasingly trade off

12: AG vs. RLE: $z(99) = 1387$, $p < .001$, $r = -.451$, $CI = [-.649, -.241]$

13: HAG vs. AG: $z(99) = 81$, $p < .001$, $r = -.968$, $CI = [-.991, -.925]$

14: On held-out sequences, model parameters are frozen and only the library is available as prior knowledge.

small reconstruction errors for more economical descriptions by calling library items and composing them hierarchically, which yields longer subsequences.

Resource efficiency

Having shown that HAG achieves the best RD trade-off, we next examined its computational cost. Since performance in our framework depends on both representations and inference resources, we varied search budget N_s and backtracking budget N_b (Figure 4.4c; see Figure A.7 for bivariate analyses).

Increasing the search budget N_s yielded only modest gains for HAG, AG, and RLE, especially late in training. With global libraries, these models update priors over programs based on past usage, producing informative proposals even under limited search. In contrast, PCFG showed a paradoxical pattern: increasing N_s sometimes *worsened* performance, because broader search proposed unnecessarily complex subprograms that had not been validated by past experience¹⁵. Increasing backtracking depth N_b benefited all models. Under high backtracking, AG approached HAG's performance, suggesting a memory-computation trade-off: HAG stores an additional local library that strengthens its program prior, whereas AG compensates through more extensive revision of earlier commitments.

Search and backtracking shape behavior in complementary ways. Increasing N_s broadens the set of candidate programs considered at each step, making it more likely that the learner discovers good compositional structure. Increasing N_b allows the learner to revise earlier commitments in light of later evidence, yielding longer but more accurate encodings as partial solutions are progressively repaired and extended (Figure A.7). Together, these simulations predict that resource limits should leave structured signatures in behavior—including systematic error patterns, recall-time dynamics, and experience-dependent generalization—which we test in the behavioral experiment.

Collectively, the simulations show that HAG outperforms others in compression efficiency and generalization, and that performance is bounded by memory and computation. These results motivate our human experiment of the same resource-bounded library-learning mechanisms: these constraints should shape both recall performance and recall-time dynamics, and library reuse should produce experience-dependent generalization. We next test these predictions in an online experiment.

4.5 Behavioral experiment

Guided by predictions from our simulations, we conducted an online experiment on Prolific with human participants ($n = 96$; see Methods). Each participant was presented with five melodic sequences (sampled from the same dataset (Garcia-Valencia et al., 2020) as the simulations), with each sequence broken up into six segments to ensure recall performance remains challenging but feasible (see Methods). The experiment consisted of three phases (Figure 4.3b). In Phase 1, participants were asked to *learn* each segment by reproducing it note by note with visual cues for the correct note. In Phase 2, participants were asked to *recall* the segment from memory. Finally, in Phase 3, participants were asked to

15: This *simplicity bias* under limited search is an interesting consequence of the PCFG's lack of a library. Without accumulated experience to guide proposals, a small N_s tends to accept low-complexity "default" parses, which happen to perform well. Larger N_s uncovers deeper-nested subprograms that increase rate without a corresponding decrease in distortion. Library-based models avoid this trap because accumulated structure constrains the proposal distribution.

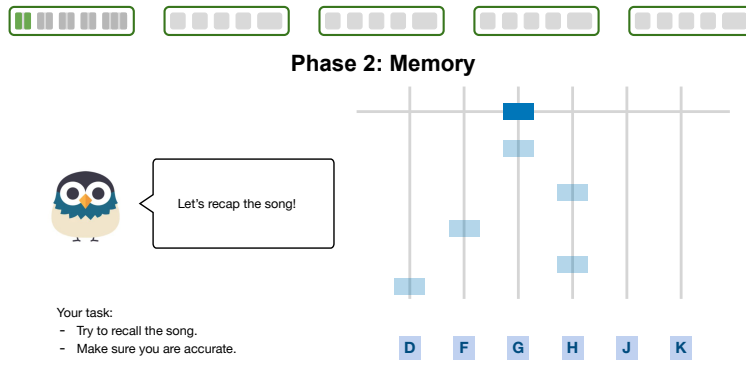


Figure 4.5: Experimental paradigm. Participants completed an online sequence-learning task in three phases. In Phase 1 (*learn*), each melodic segment was presented note-by-note with visual cues requiring a keypress response. In Phase 2 (*recall*), participants reproduced the same segment from memory without cues. Phases 1 and 2 alternated across five segments per sequence. In Phase 3 (*continue*), participants improvised a continuation consistent with the preceding five segments; the sixth segment (never shown during learning) served as ground truth.

continue the sequence by predicting the next segment based on what they previously observed. Phases 1 and 2 were presented in alternating order for the first five segments, while the sixth segment was only used as the ground truth for Phase 3.

We report three results linking behavior to the program-based account. First, we analyze Phase 2 recall errors and RTs using features from a “plausible observer” whose parameters span a broad theory-motivated range rather than being fit post hoc. Second, with the same observer, we test whether learning reflects library growth rather than generic practice. Finally, we fit HAG to trial-by-trial choices and evaluate out-of-sample Phase 3 continuations, testing whether fitted libraries explain how people extrapolate beyond training.

Participants and design. We recruited $N = 100$ participants (56 female; $M_{\text{age}} = 34.28$, $SD = 11.23$) via Prolific. Informed consent was obtained from all participants prior to participation and the study was approved by the Ethics in Psychological Research Commission of the University of Tübingen (Wu_2021/0124/213). We excluded 4 participants because of technical issues with data saving. No participants were excluded based on task performance or a priori criteria. Participants received a base payment of £4.00 plus a performance-contingent bonus ($£2.68 \pm 1.17$). The task lasted approximately 46.87 ± 28.54 min.

4.5.1 Materials and procedure

The experiment was a sequence-learning task (Figure 4.3) in which participants *learned*, *recalled*, and *continued* melodic sequences. Participants completed the study online in a browser environment. After providing informed consent, they received instructions and completed a tutorial. Comprehension was verified with three multiple-choice questions, which participants had to correctly answer before proceeding.

At the start of the experiment, each participant was randomly assigned 5 melodic sequences from the full set of training sequences to ensure a reasonable experiment length, and to avoid both fatigue and sequence-specific biases. Each sequence was subject to a random transposition to prevent over-representation of specific pitch values, while preserving the relative interval structure. An overview of the sequence statistics is provided in Figure A.1. We index the five sequences by $i \in \{1, \dots, 5\}$. Each sequence $X^{(i)} = [x_1^{(i)}, \dots, x_{T^{(i)}}^{(i)}]$ contained a string of discrete MIDI notes, which after preprocessing (seen above) each note $x_t^{(i)} \in \{1, \dots, 6\}$.

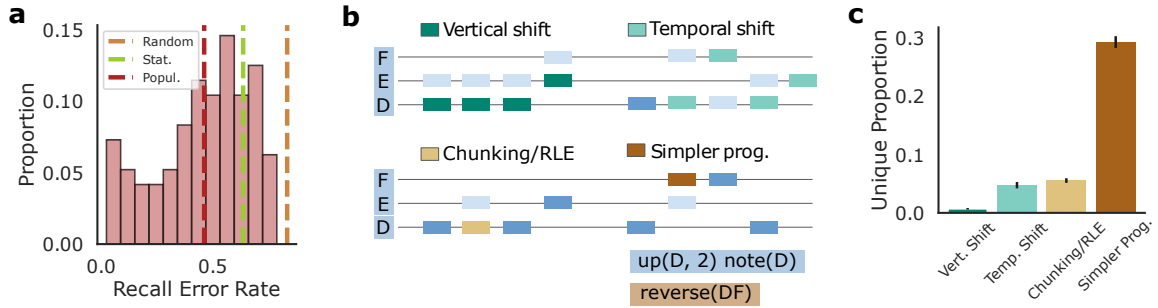


Figure 4.6: (a) The histogram of error rates across participants in Phase 2. Participants on average (Popul.) perform well above chance (Random) and first-order Markovian model (Stat.). (b) Schematic illustration of the four error categories: vertical shifts, temporal shifts, intrusions from earlier notes (RLE) or segments (Chunking), and simplification bias (i.e., corresponds to a simpler program). (c) Frequency of distinct errors across participants and segments (each error counted at most once per category per segments). Program-like errors predominate.

16: Kernel change-point detection finds breakpoints that maximally partition a sequence into statistically homogeneous segments in a kernel-induced feature space. Applied here, it places boundaries where local melodic statistics shift, yielding cuts that respect phrase structure rather than imposing fixed-length splits.

17: mean segment length was 14.89 ± 0.41 notes, first-order transition entropy was 1.21 ± 0.04 bits/note, and repetition complexity was 21.23 ± 0.87 symbols

Additionally, each sequence was further divided into six segments, denoted $X^{(i,1)}, \dots, X^{(i,6)}$, using kernel change-point detection (Celisse et al., 2017) as implemented in the *ruptures* Python package¹⁶. This reduced the memory demands of the task by presenting long sequences (usually over 100 notes) in smaller, more manageable units (12 – 16 notes), while still preserving their higher-order sequential structure. This produced segments of comparable complexity (see Figure A.1 for full distributions¹⁷).

Segments $X^{(i,1)}, \dots, X^{(i,5)}$ formed the training portion of each sequence and were presented in alternating learning and recall phases. During learning, participants viewed each segment note-by-note and pressed the corresponding key before the next note appeared. During recall, they reproduced the same segment from memory without visual cues. Responses for segment j of sequence i are denoted $\tilde{X}^{(i,j)}$; keystroke reaction times are denoted $RT^{(i,j)}$, with analyses conducted on LogRT. The sixth segment, $X^{(i,6)}$, was held out from learning and recall and served as ground truth for Phase 3. After recalling $X^{(i,5)}$, participants instead improvised a continuation consistent with the preceding five segments, allowing their responses to be evaluated against the unseen sixth segment.

Throughout the experiment, participants were incentivized to maximize accuracy in all phases. Participants earned a performance-based bonus computed from note-level accuracy (binary) in all three phases. A brief post-task questionnaire collected demographic information and self-reported strategies. The performance summaries for each stage are shown in Figure A.2.

4.5.2 Plausible observer analysis

Error patterns. We focus on analyzing error patterns during the recall task in Phase 2 to reveal signatures of structured compression. Overall, participants achieved a mean recall error rate of $.46 \pm .20$ (Figure 4.6b; see Figure A.2a for all three phases), which was lower than both a random baseline¹⁸ and a first-order Markov model $p(x_n | x_{n-1})$ fitted on training sequences¹⁹.

Behavior analysis. To first characterize the nature of recall error purely from behavior data, we classified note-level mismatches into four cate-

18: 0.83; Popul. vs. Random: $z(95) = .0, p < .001, r = -1.0, CI = [-1.0, -1.0]$

19: 0.64; Popul. vs. Stats.: $z(95) = .580, p < .001, r = -.75, CI = [-.87, -.61]$

gories. Two categories served as non-diagnostic controls, because they can arise without structured representations:

1. **Vertical shift** (non-diagnostic): A contiguous recalled sequence differed from the target by a constant pitch offset, consistent with motor imprecision or uncertainty about absolute pitch.
2. **Temporal shift** (non-diagnostic): A recalled sequence matched the target after a uniform index lag, consistent with temporal uncertainty.

The other two categories were structure-diagnostic. Together, these latter two categories test whether recall errors reflect systematic resource constraints, either through reuse or simplification:

1. **Repetition error** (structure-diagnostic): Copying earlier material, either run intrusions (RLE-like repeated-note runs) or chunk intrusions (earlier multi-note subsequences replayed in the wrong position).
2. **Simpler program** (structure-diagnostic): The recalled sequence was best explained by a shorter (more compressible) generative program than the target. For each error window, we scored the participant response and the ground truth using a precomputed library of candidate programs, labelling the error as “simpler program” when the response had a higher compressibility score.

Because errors can fall under multiple categories, we only counted unique errors assignable to a single category to avoid overcounting. Across participants, program-based simplifications were the most dominant category of errors ($.293 \pm .010$), followed by repeats ($.055 \pm .004$), temporal shifts ($.047 \pm .005$), and then transpositions ($.007 \pm .001$; Figure 4.6c). Thus, participants not only exhibited lower overall error rates than chance, but also produced error patterns diagnostic of program-like compression.

Choice predictions. We next asked which mechanisms best predict participants’ note-by-note recall choices, including both correct responses and systematic errors. To do so, we fit a logistic mixed-effects regression on participants’ choices using note-level expectation features for vertical and temporal shifts (non-structured errors), together with predictors derived from RLE, Chunking, PCFG, AG, and HAG plausible observers (structured errors). Intuitively, these predictors provide a score reflecting which choices were more likely under each model (see Methods). We jointly entered these predictors in a hierarchical softmax regression with participant-level random effects to predict each recalled note. HAG had the largest positive coefficient (Figure 4.6d)²⁰, indicating that errors were systematically captured by assuming programmatic representations with reusable libraries. By contrast, AG, Chunking, and temporal shifts yielded much smaller positive effects, while RLE, PCFG, and vertical shifts were negative. Excluding HAG significantly worsened fit, as demonstrated by a likelihood-ratio test comparing the full model to the nested model without the HAG predictor ($\chi^2(1) = 2946.21, p < .001$).

20: $\gamma = 5.82, z = 68.62, p < .001, CI = [5.63, 5.98]$

Reaction times. We then analyzed RTs for process-level evidence of program-like compression. In Phase 2, log-transformed RTs generally decreased across relative note position during a segment, but this decline was not monotonic: local jumps in RT duration were observed in addition to the overall downward trend, suggesting intermittent processing boundaries (Figure 4.7a, left). To quantify these boundaries, we computed the note-to-note change in RT within each segment and marked a boundary when the change was unusually large relative to the participant and segment²¹. These boundaries divided segments into runs of relatively fast responses separated by longer pauses. The resulting “chunk” lengths

21: thresholded at the mean ± 1 s.d. of all changes within that segment

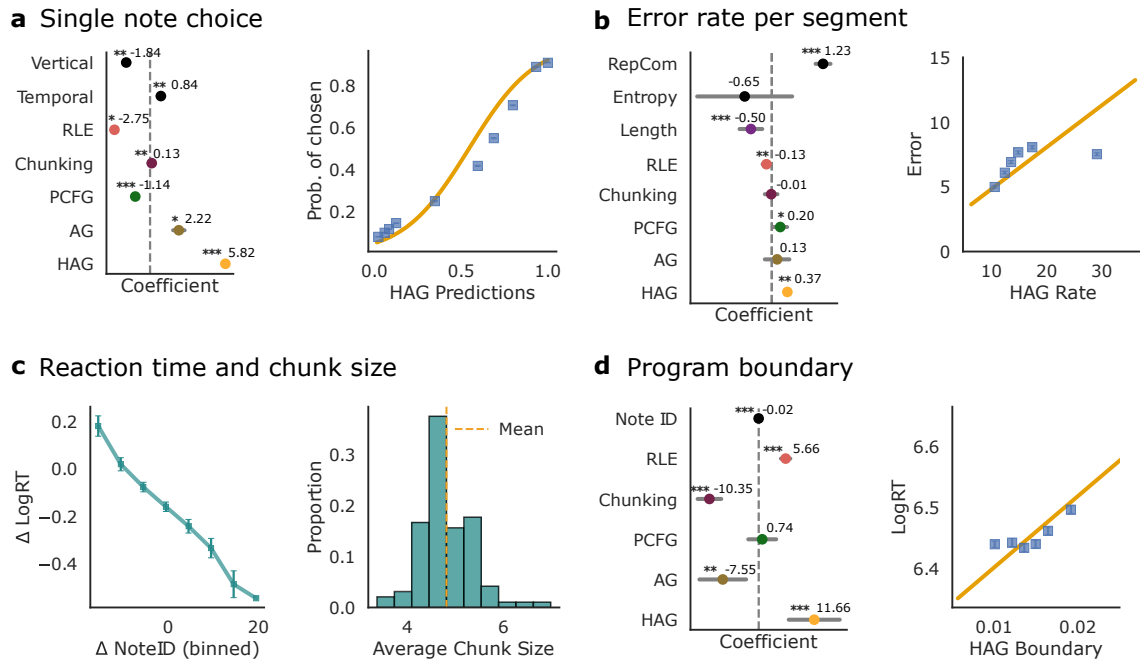


Figure 4.7: Program-based predictors explain human recall and reaction-time behavior. (a) Single note choice: regression coefficients (left) and monotonic relation between HAG predictions and choice probability (right). (b) Error rate per segment: regression coefficients for segment-level error (left) and increasing error with HAG-predicted rate (right). (c) Reaction time and chunk size: mean $\Delta \log \text{RT}$ by relative note position (left) and distribution of inferred average chunk size (right). (d) Program boundary: regression coefficients for boundary effects on log RT (left) and increasing log RT with HAG-predicted boundary probability (right). Significance: * $p < .05$, ** $p < .01$, *** $p < .001$.

averaged 4.81 notes (Figure 4.7a, right), consistent with multi-note grouping rather than uniformly fast note-by-note responding (see Figure A.2b for details).

We next asked whether these RT jumps align with subprogram boundaries. If participants internally represent sequences using concatenation of subprograms, initiating a new subprogram should incur a cognitive cost and produce an RT spike at the transition. To test this, we fit a mixed-effects regression that included model-predicted subprogram boundaries, along with note index as a control variable for generic decreases in RTs. Consistent with this prediction, HAG was the strongest predictor of increased RTs²² and significantly improved model fit when included as a regressor ($\chi^2(1) = 18.18, p < .001$; Figure 4.7b). RLE was also a significantly positive predictor but smaller in magnitude²³. By contrast, AG and PCFG did not explain RT independently of HAG, likely because their boundary predictions overlapped with HAG (AG-HAG: $r = .66$; PCFG-HAG: $r = .58$); we therefore interpret their coefficients as reflecting collinearity rather than evidence against hierarchical structure. Chunking showed a different pattern: its boundaries were associated with faster responses. Follow-up analyses showed that Chunking can only capture smaller patterns than other models (i.e., shorter subsequences per subprogram). Whereas boundaries between larger hierarchical units are associated with RT costs, Chunking places many boundaries within short local subsequences that are executed more quickly overall (see Figure A.9a-c and SI).

Learning dynamics. The preceding analyses showed that *note-level* recall choices and RTs bear signatures of program-like compression. We next asked whether *segment-level* performance across the experiment can be explained by simple exposure-based practice alone, or whether it instead

22: HAG: $\gamma = 11.66, z = 4.26, p < .001, \text{CI} = [6.30, 17.02]$

23: RLE: $\gamma = 5.66, z = 11.18, p < .001, \text{CI} = [4.67, 6.65]$

reflects changes in the internal libraries used to encode each segment.

As a baseline, we first asked how much variance could be explained by practice alone (indexed by segment number and sequence number) without invoking any change in internal representations. Participants showed reliable practice effects: RTs sped up with both segments and sequences²⁴. Consistent with these speed-ups, pauses became less frequent over learning, yielding longer contiguous runs (i.e., larger inferred chunks; Figure A.2a). Error rates also declined over learning (Figure A.2b)²⁵. However, these temporal indices explained relatively little variance in performance ($R^2 < .15$), indicating that simple practice effects capture only a limited portion of the learning dynamics.

We therefore asked whether segment-level performance was better explained by changes in the inferred representations. Applying all models as plausible observers to the same sequences participants experienced, we extracted for each segment a scalar estimate of *segment program complexity*, i.e., the description length of the best inferred program under the model's current library (see Methods). These complexity estimates were then used to predict segment-level recall error above and beyond the baseline practice indices. Segment complexity inferred by HAG²⁶, RLE²⁷ and PCFG²⁸ robustly predicted error rates, such that segments assigned higher program complexity were recalled less accurately (Figure 4.7c). By contrast, Chunking and AG were comparatively weak in the joint model because RLE, Chunking, and AG were highly correlated (Chunking vs. RLE $r = .65$, AG vs. RLE $r = .66$), indicating overlap in the complexity they capture if it comes down to a single scalar metric. Overall, adding program-based complexity predictors significantly improved fit relative to a model containing only temporal learning indices ($\chi^2(1) = 24.11, p < .001$), indicating that learning is better explained by changes in inferred segment structure and library reuse than by time-on-task or surface complexity alone.

4.5.3 Model-based analysis

We fitted each candidate model to individual participants' trial-by-trial choices. Model parameters were estimated via grid search over β (memory budget), N_s and N_b (computational budgets, drawn from Poisson distributions), and α_{glob} , α_{loc} (reuse preferences). Each model was fitted using 50 random seeds, with posterior samples used to construct categorical distributions over note space. Model performance was assessed using the Bayesian Information Criterion (BIC), hierarchical Bayesian model selection (protected exceedance probability, PXP)²⁹, and out-of-sample log loss for Phase 3 predictions. Full details are reported in Section A.6.

Model comparison and recovery. Fitted to trial-by-trial choices of each participant, HAG achieved a significantly higher log-likelihood ($-2.774 \pm .121$) than all alternative models³⁰. To infer the most likely generative model at the population level, we conducted hierarchical Bayesian model selection by computing the protected exceedance probabilities (PXP). HAG yielded the highest PXP and the lowest BIC relative to all baseline models (Figure 4.8a–b). Relative to AG, HAG had a significantly lower BIC³¹. At the individual level, HAG provided the best fit for 53 of 96 participants according to BIC, compared with 21 for AG, 17 for PCFG, 3 for RLE, and 2 for Chunking (Fig. A.10).

We also evaluated model recoverability to assess whether the fitting procedure could reliably identify the true generating model. All mod-

24: RT vs. Segments: $\gamma = -.031, z = -5.741, p < .001, \text{CI} = [-.041, -.020]$;
RT vs. Sequences: $\gamma = -.079, z = -15.377, p < .001, \text{CI} = [-.089, -.069]$

25: Error vs. Segments: $\gamma = -.004, z = -3.555, p < .001, \text{CI} = [-.006, -.004]$;
Error vs. Sequence: $\gamma = -.006, z = -5.742, p < .001, \text{CI} = [-.008, -.004]$ for sequences

26: HAG: $\gamma = .211, z = 8.068, p < .001, \text{CI} = [.160, .262]$

27: RLE: $\gamma = .138, z = 6.550, p < .001, \text{CI} = [.089, .187]$

28: PCFG: $\gamma = .343, z = 5.623, p < .001, \text{CI} = [.084, .602]$

29: PXP (Rigoux et al., 2014) is the posterior probability that a model is the most frequent generative model in the population, corrected for the possibility that all candidate models are misspecified. Unlike raw exceedance probability, PXP is conservative when the best model is only marginally better than alternatives.

30: HAG vs. AG: $z(95) = 4656, p < .001, r = .994, \text{CI} = [.980, .999]$

31: HAG vs. AG: $z(95) = 227, p < .001, r = -.90, \text{CI} = [-.96, .82]$

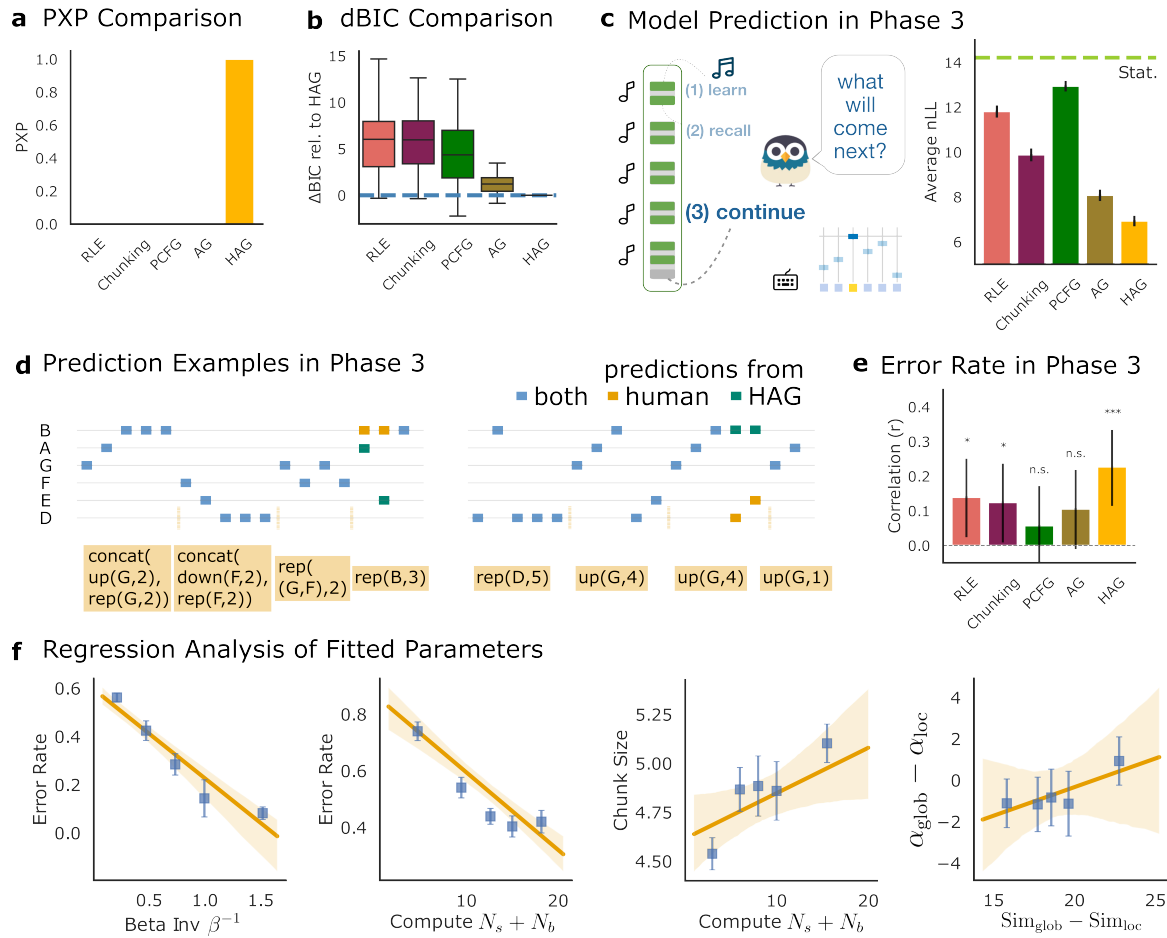


Figure 4.8: Model fitting, comparison, and parameter-behavior relationships. (a) Model comparison using protected exceedance probability (PXP), indicating the likelihood that each model is the most frequent generative model in the population. (b) Model comparison accounting for complexity, shown as relative Δ BIC values with respect to the best model for each participant. (c) Model-based predictions in the Phase 3. Participants predicted the continuation of a sequence based on what they have seen. Model predictions were generated using the participant-specific fitted model. Bars show the negative log-likelihood of participants' responses under the model predictions. (d) Example predictions produced by two participants (p36 and p71) alongside compositions generated by the HAG model using fitted parameters. (e) Correlation between participant Phase 3 error rates and model-predicted error rates, shown separately for each model. Bars indicate Pearson correlation coefficients across participants; error bars denote 95% confidence intervals. Asterisks indicate significance levels. (f) Regression analysis relating fitted model parameters to behavioral measures across participants.

els were highly recoverable (Figure A.13), with the exception of AG ($P(\text{true}|\text{recovered}) = .55$ in the inversion matrix), where AG were frequently mistaken as PCFG, RLE, or Chunking. AG implements a single, shared adaptor (a global library), which allows it to interpolate between qualitatively different behaviors depending on its adaptor strength (α_{glob}). In one extreme regime, weaker reuse (higher α_{glob}) yields behavior close to PCFGs; whereas in a strong-reuse regime (lower α_{glob}), reuse can be amplified even when genuine sequence structure has not yet been identified. Early in training, this can lead the model to propagate whatever patterns are easiest to reuse—often shallow, high-frequency subsequences or repeats—which can resemble surface-level repetition models (RLE/Chunking) especially under finite data. By contrast, HAG has stronger identifiability ($P(\text{true}|\text{recovered}) = .76$) and its parameters are recoverable (Kendall's rank correlation $\tau \geq .56$, all $p < .001$; Figure A.14). HAG's local library can capture within-sequence regularities without forcing them to generalize across sequences. As a result, sequence-specific patterns are less likely to be misrepresented as globally reusable structure, yielding clearer model identifiability in both parameter recovery and model comparison.

Out-of-sample prediction (Phase 3). We then used our fitted models to predict participant responses in Phase 3, where participants freely predicted the continuation of a sequence (observed in segments one to five). For each participant, model fitting resulted in an individualized set of libraries inferred from their behavior up to the onset of Phase 3. We then treated the fitted libraries as a generative model to the sequence continuation, and compared these model generations to participants' free responses. Because PCFG does not possess an explicit library representation, we treated it as a statistical baseline model, akin to a flexible n -gram model, and generated predictions using only its fitted parameters. The results are shown in Figure 4.8c. At the aggregate level, HAG equipped with participant-specific libraries provided the best account of Phase 3 responses, with the lowest negative log-likelihood (nLL) among all models³², and better than either a Markovian or chance baseline³³. Figure 4.8d shows two representative participants whose composition responses were closely matched by one sequence sample from the HAG library (blue represents match), providing qualitative evidence that the model captures participants' internalized structural representations.

Beyond individual examples, we also assessed whether HAG captures systematic variation in composition at the population level. Across participants, the correlation between participant Phase 3 error rates and model-generated error rates for each candidate model (Figure 4.8e). HAG showed the strongest participant-level correspondence ($r = .23$, $p < .001$), exceeding AG ($r = .11$, n.s.), PCFG ($r = .06$, n.s.), and the simpler RLE ($r = .14$, $p < .05$) and Chunking baselines ($r = .12$, $p < .05$). Thus, beyond fitting individual continuations, HAG better captured between-participant variability in how learners generalized sequence structure to unseen continuations. Together, these results suggest that participant-specific hierarchical libraries support out-of-sample prediction of continuation behavior after limited exposure.

Parameter analysis. We next examined the fitted parameters of the winning HAG model. For intuition, we report parameters such that larger values correspond to greater resource demands, facilitating a direct comparison of memory vs. computational trade-offs. Specifically, larger inverse temperature β^{-1} reflects increased memory capacity, whereas larger values of the search budget (N_s), backtracking budget (N_b), and proposal rates (α , α_{glob}) reflect increased computational capacity. Across participants, fitted parameters showed modest but systematic dependencies (Figure A.13c). Memory capacity (β^{-1}) was negatively correlated with computational capacity, including search budget (N_s ; $r = -.24$), backtracking budget (N_b ; $r = -.20$), local proposal rate (α_{loc} ; $r = -.11$), and global proposal rate (α_{glob} ; $r = -.10$), indicating a trade-off between memory and computation. We also observed a strong positive correlation between the search budget (N_s) and the tendency to propose new local programs (α_{loc} ; $r = .19$). This suggests that participants who searched more extensively also relied less on reuse of previously formed local structures and more on proposing novel candidates.

We next asked whether individual differences in parameter estimates could explain behavioral variability. Participants' error rates were significantly predicted by both memory β^{-1} ³⁴ and computational capacity $N_s + N_b$ ³⁵, with greater capacity yielding lower errors. In addition, computational resources positively predicted chunk size, derived from RT (Figure 4.8c), indicating that greater computational capacity supports the formation of larger chunks³⁶. To test whether these effects were captured quantitatively by the fitted models, we simulated each participant using their fitted parameters and compared predicted with observed individual

32: HAG vs. AG: $z(95) = 76381$, $p < .001$, $r = .323$, $\text{CI} = [.224, .420]$

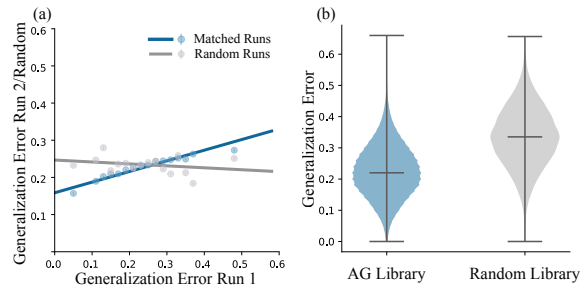
33: $nLL_{\text{HAG}} = 6.93$, $nLL_{\text{Stat.}} = 14.21$, and $nLL_{\text{Random}} = 25.69$

34: β^{-1} : $\gamma = -.370$, $z = -141.32$, $p < .001$, $\text{CI} = [-.375, -.365]$

35: $N_s + N_b$: $\gamma = -.051$, $z = -18.449$, $p < .001$, $\text{CI} = [-.027, -.022]$

36: Chunk size: $\gamma = .038$, $z = 24.478$, $p < .001$, $\text{CI} = [.035, .041]$

Figure 4.9: Curriculum order systematically affects library quality. (a) Scatter plot of generalization error for matched runs (same curriculum, different random seeds) vs. random pairings (different curricula). Matched runs are positively correlated; random pairings are weakly negatively correlated. (b) Generalization performance of learned libraries learned under different curricula vs. randomly generated libraries. While random libraries perform worse on average, their variance is comparable.



differences in error rate and chunk size (Figure A.12). HAG showed the strongest participant-level correspondence on both measures and, in joint mixed-effects regressions controlling for competing models, explained unique variance in both outcomes (statistics in Figure A.12).

Finally, we examined parameters governing global versus local reuse. We used the parameter contrast $\alpha_{\text{glob}} - \alpha_{\text{loc}}$ as an index of relative reliance on between-sequence versus within-sequence reuse, and quantified each participant's training materials using a similarity contrast comparing structure shared across sequences to structure repeated within sequences. Concretely, this contrast summarizes the extent to which a participant's five training sequences contain recurring motifs and transformations that reappear across different sequences versus that are confined to individual sequences. The similarity contrast significantly predicted the reuse-parameter contrast: participants exposed to training sets with more cross-sequence relative to within-sequence structure showed larger values of $\alpha_{\text{glob}} - \alpha_{\text{loc}}$ ³⁷, indicating greater inferred reliance on a global library. This links individual reuse strategies to the structural statistics of the training sequences participants experienced.

37: $\alpha_{\text{glob}} - \alpha_{\text{loc}}$: $\gamma = .277$, $z = 102.254$, $p < .001$, $\text{CI} = [.271, .282]$

4.6 Curriculum effects

The preceding results established that library learning is path-dependent, i.e., the order in which learning materials are encountered shapes the library that is constructed, which in turn determines what can be efficiently represented and generalized. In this section, we report an exploratory simulation study that investigates curriculum effects more directly and proposes a principled method for designing curricula that induce more effective libraries. This analysis has not yet been validated with human participants and should be understood as a proof of concept; we include it because it illustrates a natural consequence of the path-dependent library framework and points toward future experimental work.

4.6.1 Curriculum sensitivity

38: A full posterior $p(\mathcal{L} \mid X^{1:i})$ would be invariant to presentation order because it conditions on all past data jointly. A point estimate cannot be revised without re-running inference, so early examples disproportionately shape which programs are cached first.

Because we approximate the library posterior by a single point estimate³⁸, the model is sensitive to training order. We quantified this by simulating 50 melodies across 1,000 randomly sampled curricula.

Learned libraries varied substantially across curricula, sharing only $3.19\% \pm 0.05\%$ of subprograms on average. To separate curriculum effects from sampling stochasticity, we ran the model twice per curriculum and compared generalization error for matched runs (same curriculum) versus random runs (different curricula). Matched runs were positively

correlated ($r(998) = .286, p < .001$), while random runs were weakly negatively correlated ($r(998) = -.052, p < .001$); this difference was significant ($t(999) = 7.621, p < .001$). Thus, curriculum effects cannot be explained by stochasticity alone. We also compared learned libraries across curricula with randomly generated libraries. Random libraries performed worse on average, but showed comparable variability, indicating that unfavorable curricula can disadvantage adaptive learners. This highlights the practical importance of curriculum design for library-based learning.

4.6.2 Synergistic curriculum design

Given that curricula matter, can we design them *a priori* to induce better libraries? We introduce a synergistic curriculum-building method using principles from Partial Information Decomposition (PID; Williams and Beer, 2010). PID was introduced to resolve a longstanding ambiguity in multivariate information: knowing that two variables together predict a target does not distinguish whether they each provide the *same* information (redundant), *complementary* information (unique), or information that only *emerges jointly* (synergistic) (Williams and Beer, 2010).

PID decomposes the multivariate mutual information $MI(\rho_1, \dots, \rho_M; X)$ between multiple source programs $\rho_m \in \mathcal{L}$ and a target variable X (melodies) into distinct components:

- **Unique** information $U(\rho_m; X)$: present in exactly one program ρ_m .
- **Redundant** information $R(\rho_i, \dots, \rho_j; X)$: encoded separately by multiple programs.
- **Synergistic** information $S(\rho_1, \dots, \rho_M; X)$: jointly encoded by multiple programs together, but not available from any single program alone.³⁹

For a library $\mathcal{L}$ containing two programs, the synergy is:

$$S(\mathcal{L}; X) = MI(\pi_1, \pi_2; X) - R(\pi_1, \pi_2; X) - U(\pi_1; X) - U(\pi_2; X). \quad (4.4)$$

We hypothesize that a synergistic library—one in which programs jointly encode information that none encodes alone—is essential for flexible generalization across novel problems. A synergistic library contains programs that “complement” each other: each program captures a different aspect of the domain, and together they cover the stimulus space more efficiently than redundant or independent programs would⁴⁰.

We strategically select and order melodies to maximize synergy. Given the learned library $\mathcal{L}^{(n)}$ at step n , we select the next presented melody X^* by maximizing the expected synergy of the resulting library:

$$X^* = \arg \max_{X^{(i)} \in \mathcal{X}} S(\mathcal{L}; X) \quad \text{where} \quad \mathcal{L} \sim p(\mathcal{L}^{(n+1)} | \mathcal{L}^{(n)}, X^{(n)}). \quad (4.5)$$

The aim is to construct a training sequence where each new melody adds subprograms to the library that enhance overall representational capacity.⁴¹

4.6.3 Synergistic curricula improve generalization

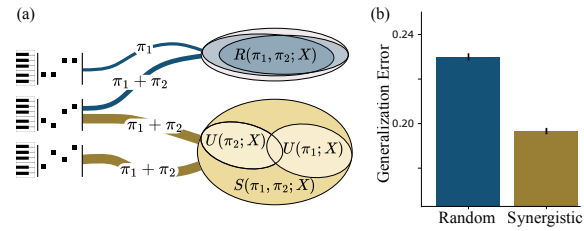
We tested whether generalization performance under a synergistic curriculum improves relative to random orderings. Across 1,000 simulations,

39: Synergy is positive when the programs together predict more about the melodies than either can alone plus their shared component. A library of two identical programs has $S = 0$ and $R > 0$; a library of two perfectly complementary programs has $S > 0$ and $R = 0$.

40: Connecting to Section 3.6, a redundant library wastes representational resources by storing overlapping compressions; a synergistic library maximizes the compression benefit per entry, because each entry extends the library’s coverage into a region of program space not reachable from any other entry alone. Synergy is, in effect, a measure of how well the library satisfies the resource-rational criterion of Equation (2.4).

41: The selection is greedy: it maximizes expected synergy one step ahead without planning over future curricula. A full curriculum search would require evaluating all $(N - n)!$ orderings of remaining melodies, which is computationally infeasible even for small N . Greedy selection provides a tractable approximation that still outperforms random ordering.

Figure 4.10: Synergistic curricula induce more effective libraries. (a) PID decomposition of libraries learned under random (blue) vs. synergistic (yellow) curricula, showing the proportion of redundant R , unique U , and synergistic S information. Synergistic curricula produce libraries with higher joint information and less redundancy. (b) Generalization error under random vs. synergistic curricula across 1,000 simulations. Synergistic curricula yield significantly lower generalization error.



the synergistic curriculum yielded significantly lower generalization error than random curricula ($t(999) = 6.219, p < .001$). Libraries learned under synergistic curricula contained programs with higher joint information and less redundancy, confirming that the PID-based selection method promotes the formation of complementary, mutually informative library entries.

This curriculum analysis is exploratory in several respects. First, the synergy computation assumes access to the current library state, which may not be directly measurable in human learners. Second, the greedy selection algorithm considers only one step ahead; more sophisticated planning over curriculum sequences could yield further improvements⁴². Third, and most importantly, the synergistic curriculum method has not yet been tested with human participants. Despite these limitations, the result provides a proof-of-concept that information-theoretic measures defined over program libraries can guide the design of training sequences, connecting the theoretical framework of library management to practical questions in education and cognitive training.

42: Beam search or Monte Carlo tree search over curriculum sequences are natural extensions. However, their cost scales as $O(b \cdot N)$ per step for beam width b , and the value function (future library synergy) requires simulating forward through many updates—making even moderate look-ahead expensive at realistic dataset sizes.

4.7 Discussion

4.7.1 Findings

Compression signatures. Participants' recall errors were not well explained by memorizing isolated notes or shallow transition statistics. Instead, errors clustered toward more compressible versions of the target sequence. This follows from a compression objective: under resource constraints, learners favor shorter descriptions, even when they distort the original, yielding regularization errors rather than arbitrary substitutions (Bates and Jacobs, 2020; Brady et al., 2009; Culbertson et al., 2012; Reali and Griffiths, 2009). HAG predicted these errors better than all baselines, and removing it significantly worsened model fit. RTs also revealed process-level signatures of latent program structure: participants slowed at boundaries predicted by inferred subprograms. Initiating a new subprogram incurs a cognitive cost analogous to hierarchical action initiation (Lashley et al., 1951; Sakai et al., 2003; Verwey and Dronkert, 1996), and HAG boundary predictions were the strongest RT predictor. Together, these results support internal segmentation at compositional program joints rather than surface-level repetition boundaries.

Library learning over experience. Performance improvements across segments were better explained by decreasing program complexity under the model's evolving library than by time-on-task alone. Once a useful program is cached, later sequences that overlap with it become

cheaper to encode and easier to recover under bounded search (Ellis et al., 2021; Xia and Collins, 2021). Participant-specific HAG libraries also predicted Phase 3 continuation behavior better than all competing models, demonstrating transfer to sequences never encountered during training.

Individual differences as resource allocation. Fitted HAG parameters revealed systematic structure in how participants allocated cognitive resources. Both memory and computational capacity covaried with recall accuracy, and computational resources positively predicted chunk size derived from RTs. The relative reliance on cross- versus within-sequence reuse tracked the structural statistics of each participant's training sequences. These results recast individual differences as variation in resource allocation shaped by learning history, rather than as differences in general ability.

Path dependence. More broadly, library-based compression provides a principled basis for path dependence in learning (Bruner, 1973; Ritter et al., 2007). Because libraries reshape both what is cheap to encode and what is easy to infer, early experiences determine which structures enter the library, and those early commitments bias which regularities become visible under bounded search. Compression, segmentation, and generalization accordingly arise from a single hierarchical system rather than from separate mechanisms.

4.7.2 Limitations

Task scope. The experiment uses short, controlled melodic sequences with restricted rhythmic and timbral variation. This controlled setting supports identification of latent program structure but is not a model of music learning per se, which additionally involves meter, expressive timing, longer-range dependencies, and sensorimotor demands.

Completeness of the cost model. Our framework adds search and backtracking to the standard informational cost, but does not exhaust the space of computational constraints. Maintaining, updating, and refactoring the library itself carry costs that lie outside the current formulation. The framework should therefore be understood as a proof of principle about one class of computational constraints, rather than as a complete account.

Point-estimate approximation. The library posterior is approximated with a single point estimate rather than a full distribution over possible libraries. This approximation is necessary for tractability, but it means the model cannot recover from early commitments without re-running inference from scratch, and it prevents a principled quantification of library uncertainty.

4.7.3 Towards human experiments on curriculum learning

Our exploratory curriculum simulations (Section 4.6.1–Section 4.6.3) show that presentation order systematically shapes library quality and that PID-based synergy provides a principled criterion for selecting beneficial orderings. A natural next step is to assign human participants to different curriculum conditions (random, synergistic, anti-synergistic) and compare their learning outcomes and inferred library structures.

Manipulations of computational budget via time pressure or dual-task load should shift learners toward shallower programs and greater reliance on previously cached subroutines, producing structured changes that go beyond a uniform accuracy drop. Applying the same framework to other compositional domains—language, mathematics, motor sequences—can evaluate whether library learning is a shared mechanism for abstraction and reuse across cognitive systems (Ellis et al., 2021; Zhou et al., 2024a).

4.8 Summary

This chapter showed that human sequence learning reflects resource-rational program induction with hierarchical libraries. Learners compress melodies into compositional, program-like representations, balancing fidelity against description length and bounded search while accumulating reusable structure across contexts. The hierarchical adaptor grammar, with both global cross-melody and local melody-specific libraries, best explained recall errors, reaction times, free compositions, and individual differences in resource allocation. The curriculum analysis showed that library construction is path-dependent and that information-theoretic principles can guide training sequences that build more effective representational infrastructure.

From formalism to behavioral evidence. The background chapters argued that representational dynamics—the construction, curation, and communication of reusable internal structure—should be central objects of study. This chapter provides empirical evidence for that claim. The key finding is not just that program induction is plausible, but that library-growth dynamics are recoverable from behavior: which programs are reused, how libraries reshape inference, and how experience order determines representational infrastructure. Path dependence, predicted as a generic consequence of resource-bounded library learning, is demonstrated in simulation and in human curriculum sensitivity. The memory-computation trade-off predicted by Section 3.6 is likewise recovered as an individual-difference parameter covarying with recall accuracy and chunking behavior.

What this chapter does not address. The present results address *how* libraries are constructed, not whether they transfer to richer settings. The controlled setting—short melodies, restricted pitch sets, and a single learning session—supports identifiability but leaves open whether the same mechanism operates in large-scale naturalistic learning. Chapter 5 takes up this question, moving from hand-crafted symbolic programs to large-scale educational data, where the target representations are prerequisite graphs and individualized cognitive traits rather than program trees. If the construction mechanism generalizes, its traces should appear not only in laboratory recall but also in the structured learning trajectories of hundreds of thousands of students. The chapter also raises a further question: library growth is path-dependent, but who chooses the path? The curriculum analysis showed that different orderings produce different libraries, some systematically better than others. Yet the learner here does not choose its own curriculum. How agents should *actively manage* their representational resources—what to seek, practice, or prune—is the question of curation, and the subject of Part III.

What learners share: transferable structure across learners

5.

“Habit is thus the enormous fly-wheel of society.”

— William James, *The Principles of Psychology*, 1890

The preceding chapter showed that human concept learning can be understood as program induction under resource constraints, and that the order of encounters leaves lasting traces in the representations learners build. That account operated in a controlled laboratory setting: carefully-calibrated stimuli, small participant groups, and predefined discrete symbolic programs over a hand-crafted base language. Two features of that setting—control and scale—cannot be assumed in the wild. Real learners navigate domains structured by prerequisite relations across hundreds of thousands of practice events, at scales where the sample-based Bayesian approximations that work in the lab become computationally intractable.

This chapter shifts the register accordingly, moving from controlled observation to inference from naturalistic data. Rather than designing experiments, building normative and descriptive models of how each single learner builds internal representations, we ask the complementary question: given the behavioral traces left by numerous learners on real-world educational platforms and our prior knowledge on structured internal representations, how can we infer these representations efficiently, and how can we use these representations and generative models to facilitate downstream tasks (e.g., improve prediction and support personalized tutoring)? We present PSI-KT (Predictive, Scalable, and Interpretable Knowledge Tracing), a hierarchical probabilistic generative model of human learners applied to three large-scale educational datasets, which jointly infers latent knowledge states, learner-specific cognitive traits—forgetting rate, long-term memory, volatility, and transfer ability—and a shared prerequisite graph over knowledge components.

Construction here still refers to the learner’s activity, not ours as model designers. In shifting from experiment to modeling, PSI-KT asks whether structured knowledge accumulation leaves interpretable traces in large-scale data at all—by inferring the representations that learners have already built through natural learning. That it does extends the thesis’s core claim beyond the laboratory. The chapter also instantiates the path-dependence theme in a new form: prerequisite structure creates a specific kind of order-dependence, where mastering a concept changes the attractor landscape for subsequent ones, so that the history of what a learner has studied shapes what they can efficiently acquire next.

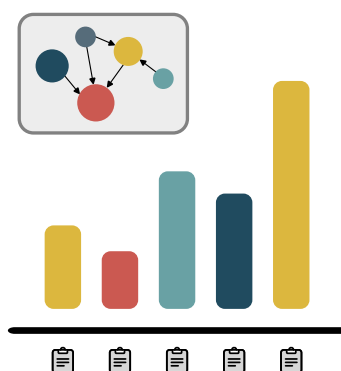
The chapter is organized as follows. [Section 5.1](#) motivates the technical problem and connects it to the field. [Section 5.2](#) reviews related work. [Section 5.3](#) presents the model and inference procedure. [Section 5.4](#) reports empirical evaluations. [Section 5.5](#) situates the findings.

5.1 Introduction

The rise of online education platforms has created new opportunities for personalization in learning, motivating a renewed interest in how

5.1 Introduction	81
5.2 Background	82
5.3 Methods	84
5.4 Results	89
5.5 Discussion	94
5.6 Summary	95

Code and experiments available at the
Github repository [psikt](#)



humans learn structured knowledge domains. Foundational theories in psychology (Ebbinghaus, 1885) have informed *spaced repetition* schedules (Settles and Meeder, 2016), which exploit the finding that an optimal spacing of learning sessions enhances memory retention. Yet beyond the timing of rehearsals, the sequential order of learning materials is also crucial, as evidenced by curriculum effects in learning (Dekker et al., 2022; Dewey, 1910), where exposure to simpler, prerequisite concepts can facilitate the apprehension of higher-level ideas. Cognitive science and pedagogical theories have long emphasized the relational structure of knowledge in human learning (Piaget, 1970; Rumelhart, 2017), with recent research showing that mastering prerequisites enhances concept learning (Brändle et al., 2022; Karuza et al., 2016; Lynn and Bassett, 2020). Yet, we still lack a predictive, scalable, and interpretable model of the structural-temporal dynamics of learning that could be used to develop future intelligent tutoring systems.

Here, we present PSI-KT, a novel approach for inferring interpretable learner-specific cognitive traits and a shared knowledge graph of prerequisite concepts. We demonstrate our approach on three real-world educational datasets covering structured domains, where our model outperforms existing baselines in terms of *predictive* accuracy (both within- and between-learner generalization), *scalability* in a continual learning setting, and *interpretability* of learner traits and prerequisite graphs. Altogether, PSI-KT bridges machine learning and cognitive science, leveraging our understanding of human learning to build the foundations for automated tutoring systems with broad educational applications.

5.2 Background

5.2.1 Problem of knowledge tracing

A central goal in the cognitive science of learning is to infer how knowledge and memory evolve from sparse behavioral traces. Classic work characterizes learning as an accumulation process and forgetting as a time-dependent decay process, with strong and stable individual differences in rates of acquisition, retention, and susceptibility to interference, as well as robust contextual effects such as spacing and recency (Ackerman, 1988; Cepeda et al., 2006; Ebbinghaus, 1885; Underwood and Postman, 1960). Knowledge tracing (KT) formalizes this cognitive inference problem in a setting where we observe sequences of learner-item interactions (Corbett and Anderson, 1994a). Historically, this formalization was motivated by intelligent tutoring systems (ITS)¹ and other adaptive practice platforms, which must decide what to present next based on limited observations (Pressey, 1926). In this chapter, we focus on KT as the problem of predicting future performance while inferring compact latent representations of (i) a learner’s evolving knowledge state, (ii) stable learner-specific traits, and (iii) shared domain structure that governs transfer across knowledge components.

Formally, a learner ℓ receives exercises or flashcards for knowledge concepts (KCs) $x_n^\ell \in \{0, 1, \dots, K\}$ at irregularly spaced times t_n^ℓ , whereupon the performance is recorded, often as correct/incorrect, $y_n^\ell \in \{0, 1\}$. We can formalize KT as a supervised learning problem on time-series data, where the goal of the KT model is to predict future performance (e.g., $\hat{y}_{N+1}$) given all or part of the interaction history $\mathcal{H}_{1:N}^\ell := \{x_n^\ell, t_n^\ell, y_n^\ell\}_{n=1}^N$ available up to time t_N^ℓ . As part of the process,

1: Corbett and Anderson (1994a) describe KT as monitoring a student’s changing knowledge state during practice by updating mastery estimates after each opportunity to apply a rule.

a KT model may infer specific representations of learners or of the learning domain to help prediction. If these representations are interpretable, they can be valuable for downstream learning personalization.

The KT problem is especially interesting given the resource constraints under which learning and instruction operate. Learners have limited time and attention, tutoring systems must make decisions with limited data (especially for new learners or new content), and the computational cost of model updates must be manageable in real-time educational settings. These constraints make the problem a natural testing ground for the resource-rational perspective developed in [Section 2.3.2](#).

5.2.2 Existing models for knowledge tracing

We broadly categorize related KT approaches into psychological and deep learning methods.

Psychological methods. The earliest KT approaches draw directly on psychological theories of memory and learning. The forgetting curve in Ebbinghaus (1885)—the observation that memory strength decays over time in a roughly exponential fashion—has informed spaced repetition schedules (Settles and Meeder, 2016) and *factorbased regression models* that predict performance from hand-crafted features such as repetition counts, time intervals, and KC difficulty (Ackerman, 2014; Gervet et al., 2020; Lindsey et al., 2014; Lord, 2012a; Pavlik et al., 2021). Half-life Regression HLR; Settles and Meeder, 2016, developed at Duolingo, models the probability of recall as a function of the time elapsed since last practice, with the half-life of memory depending on the learner’s history of correct and incorrect attempts. The Predictive Performance Equation PPE; Walsh et al., 2018 extends this approach by modeling the elapsed time of every past interaction with a power function to account for spacing effects. These models are interpretable by construction—each parameter has a clear psychological meaning. However, they rely on hand-crafted summary statistics rather than representations learned from data, and their shallow functional forms limit predictive capacity, especially if they want to model the relational structure between concepts.

Latent variable models represent a step toward richer representations. Bayesian Knowledge Tracing (Corbett and Anderson, 1994b) uses a two-state Hidden Markov Model to represent either mastery or non-mastery of each KC, with transition probabilities governing the chance of learning. Extensions have introduced individualized parameters and contextual estimation (Baker et al., 2008; Käser et al., 2017; Sao Pedro et al., 2013; Yudelson et al., 2013). However, binary states are a coarse approximation of continuous knowledge, and these models typically do not account for the temporal dynamics of forgetting or the structural dependencies between KCs. The Hawkes Knowledge Tracing model HKT; Wang et al., 2021 represents the closest prior work in spirit to the model we developed here. HKT models knowledge evolution as a multivariate Hawkes process, capturing both temporal decay and cross-KC structural influences. However, HKT lacks learner-specific representations entirely—it pools all learners into a single set of dynamics, forgoing personalization.

Deep learning methods. Deep learning methods use flexible models with many parameters to achieve high prediction accuracy. However, this flexibility also makes it difficult to interpret their learned internal representations. The first deep learning methods explicitly modeled sequential interactions with *recurrent neural networks* to overcome the dependence

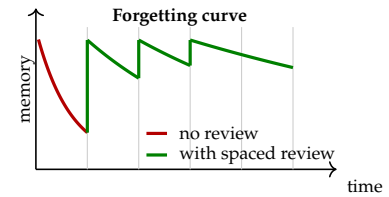


Figure 5.1: Stylized forgetting curves illustrating how spaced review can restore retention and progressively slow subsequent forgetting.

on fixed summary statistics in simpler regression models, with Deep Knowledge Tracing DKT; Piech et al., 2015 pioneering the use of Long Short-Term Memory (LSTM) networks (Hochreiter and Schmidhuber, 1997). A similar architecture, DKTF (Nagatani et al., 2019) incorporated additional input features, whereas Shen et al. (2021) proposed an intricate modular architecture aimed at recovering interpretable learner representations, but neglecting KC relations.

Structure-aware models leverage KC dependencies, accounting for the fact that human knowledge acquisition is structured by dependency relationships i.e., concept maps; Hill, 2005; Koponen and Nousiainen, 2018; Lynn and Bassett, 2020. Tong et al. (2020) empirically estimate KC dependencies from the frequencies of successful transitions. AKT (Ghosh et al., 2020) relies on the attention mechanism (Vaswani et al., 2017b) to implicitly capture structure (Choi et al., 2020; Liu et al., 2023; Pandey and Karypis, 2019; Shin et al., 2021), whereas GKT (Nakagawa et al., 2019) models it explicitly based on graph neural networks (Kipf and Welling, 2016). Recent work towards *interpretable* deep learning KT starts by engineering features such as learner mastery or exercise difficulty (Minn et al., 2022), or infers them with neural networks (QIKT; Chen et al., 2023) (Long et al., 2021). Despite a variety of approaches ranging from post-hoc local explanations to introducing ad-hoc structures or parameters for global interpretability, we still lack a comprehensive evaluation framework for interpretability.

5.2.3 Prerequisites as reusable structured knowledge

A central limitation of existing KT models is their treatment of knowledge components as independent entities. Yet a recurring claim across cognitive science and education is that domain knowledge is *relational*: concepts are organized by dependency structure that enables transfer, abstraction, and efficient inference (Hill, 2005; Koponen and Nousiainen, 2018; Rumelhart and Ortony, 2017). In this view, learning is building supporting structure that changes what can be learned next.

Prerequisite relations operationalize this structure in a particularly tractable way. Empirically, mastering foundational concepts improves learning of downstream ones (Karuza et al., 2016; Lynn and Bassett, 2020), and curriculum order affects compositional generalization (Dekker et al., 2022). From the standpoint of this thesis, prerequisite structure is a paradigmatic example of reusable representational infrastructure. It creates path dependence in the learner’s knowledge trajectory: the same concept, encountered under different prerequisite histories, produces different knowledge states. This is a property of the interaction between learners and concepts through the accumulated structure of prior learning. We will also make this link operational in Section 5.4.4 and Figure 5.7: PSI-KT’s prerequisite-aggregated signal for a target KC predicts a learner’s *initial* performance on that KC, and inferred edge weights track a Bayesian measure of causal support in held-out consecutive interactions.

5.3 Methods

In this section, we describe PSI-KT, our probabilistic hierarchical state-space model of human learning. Briefly, observations of learner performance y (Figure 5.2a, filled/unfilled boxes) provide indirect and noisy

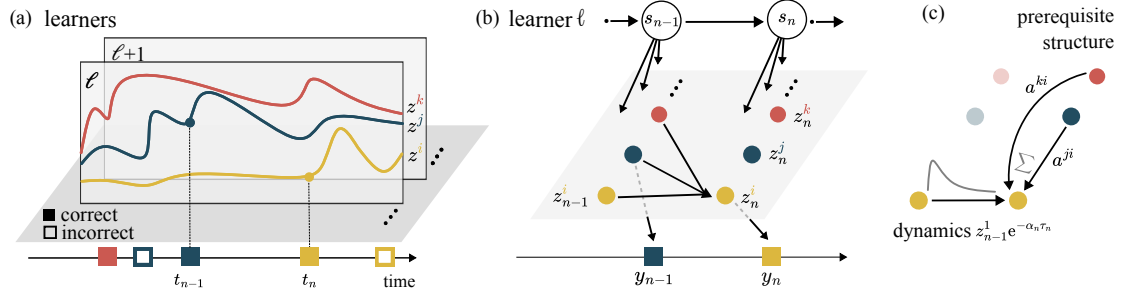


Figure 5.2: PSI-KT is a hierarchical probabilistic state-space model of learning. (a) Latent knowledge states for different KCs (colored curves) are inferred from observations. (b) Full hierarchical model for a single learner: cognitive traits s_n control the coupled dynamics of states z_n^k , which give rise to observations y_n . (c) The dynamics combine memory decay (Equation (5.6)) and structural influences (Equation (5.5)).

evidence about latent knowledge states z (colored curves, with matching dots in Figure 5.2b). These latent states evolve stochastically, in line with the psychophysics of memory (temporal decay in Figure 5.2c), while simultaneously being subject to structural influences from performance on prerequisite KCs (structure in Figure 5.2c). We introduce a second latent level of learner-specific traits s (Figure 5.2b, top), which govern the knowledge dynamics in an interpretable way.

Below, we describe the method in more detail. We start with the generative model (Section 5.3.2). Next, we discuss the joint approximate Bayesian inference of latent variables and estimation of generative parameters (Section 5.3.3). Finally, we show how to derive multi-step performance predictions (see Section 5.3.4 and Figure B.1 for a graphical overview of inference and prediction).

5.3.1 Model overview and design principles

Three design choices shape PSI-KT and distinguish it from prior KT models.

Generative over discriminative. PSI-KT specifies a complete probabilistic process over learner interactions rather than directly mapping histories to predictions. This enables forward simulation of learning trajectories, counterfactual predictions², and uncertainty quantification

Psychologically grounded representations. Each latent variable corresponds to a meaningful quantity. Knowledge states track mastery on a continuous scale. Cognitive traits encode forgetting rate, long-term memory consolidation, knowledge volatility, and transfer ability. The prerequisite graph encodes directed dependency relationships between KCs. This structured parameterization makes the model interpretable without post-hoc explanation.

Shared domain structure, individual learner variation. The prerequisite graph is constructed by all learners while cognitive traits capture how each learner interacts with the domain. This separation lets the model pool evidence across learners to estimate graph structure while still personalizing to each individual.

2: e.g. what if the learner had studied concept A before concept B ?

5.3.2 Probabilistic state-space generative model

We conceptualize observations of learner performance as noisy measurements of an underlying time-dependent knowledge state, specific to each

learner and KC. The evolution of knowledge states reflects the process of learning and forgetting, governed by learner-specific traits. Additionally, knowledge of different KCs informs one another according to learned prerequisite relationships. We translate these modeling assumptions into a generative model consisting of three main components:

1. the learner knowledge state across KCs, $\mathbf{z}_n^\ell = [z_n^{\ell,1} \dots z_n^{\ell,K}]^\top \in \mathbb{R}^K$ (colored curves in [Figure 5.2a](#))
2. learner-specific cognitive traits $s_n^\ell \in \mathbb{R}^4$ (top row in [Figure 5.2b](#))
3. a shared static graph $\mathcal{A}$ of KCs whose edges a^{ik} quantify the probability for a KC i to be a prerequisite for KC k ([Figure 5.2c](#))

State-space model. State-space models (SSMs) (Section 3.1.2) are a framework for partially observable dynamical processes. They represent the inherent noise of measurements y by an emission distribution $p(y_n | z_n)$, separate from the stochasticity of state dynamics, modeled as a first-order Markov process with transition probabilities $p(z_n | z_{n-1})$. The state dynamics are initiated by sampling from an initial prior $p(z_1)$ to iteratively feed the transition kernel, and predictions can be drawn at any time from the emission distribution. To represent the influence of individual cognitive traits over the knowledge dynamics, we additionally condition the z -transitions on the traits s (which also can be observed only indirectly). The three-level SSM hierarchy of PSI-KT consists of:

$$\text{Level 2 (latent cognitive traits):} \quad s_n^\ell \sim p_\theta(s_n^\ell | s_{n-1}^\ell) := \mathcal{N}(s_n^\ell | Hs_{n-1}^\ell, R) \quad (5.1)$$

$$\text{Level 1 (latent knowledge states):} \quad \mathbf{z}_n^\ell \sim p_\theta(\mathbf{z}_n^\ell | \mathbf{z}_{n-1}^\ell, s_n^\ell) := \prod_k \mathcal{N}(\mathbf{z}_n^{\ell,k} | m_n^{\ell,k}, w_n^\ell) \quad (5.2)$$

$$\text{Level 0 (observed learner performance): } \hat{y}_n^\ell \sim p(y_n^\ell | z_n^{\ell,k}) := \text{Bern}(\sigma(z_n^{\ell,k})). \quad (5.3)$$

The choice of Gaussian initial priors (discussed below) and Gaussian transitions ensures tractability, while the Bernoulli emissions model the observed binary outcomes. We now unpack this model and all its parameters in detail, starting with the knowledge dynamics.

Knowledge states z . Recent KT methods Nagatani et al. (e.g., 2019) use an exponential forgetting function based on psychological theories (Ebbinghaus, 1885). Here, we augment this approach by adding stable long-term memory (Averell and Heathcote, 2011), and model the knowledge dynamics $z^{\ell,k}$ of an isolated KC k as a mean-reverting stochastic (Ornstein-Uhlenbeck; OU) process:

$$dz^{\ell,k}/dt = \alpha^\ell(\mu^\ell - z^{\ell,k}) + \sigma^\ell\eta(t). \quad (5.4)$$

Accordingly, the state of knowledge $\mathbf{z}^\ell$ gradually reverts to a long-term mean μ^ℓ with rate α^ℓ , subject to white noise fluctuations $\eta(t)$ scaled by volatility σ^ℓ . To account for the influence of other KCs, we adjust the mean μ_n^ℓ using prerequisite weights a^{ik} (defined in Equation (5.8) below), modulated by the learner’s transfer ability γ_n^ℓ :

$$\tilde{\mu}_n^{\ell,k} := \mu_n^\ell + (\gamma_n^\ell/K) \sum_{i \neq k} a^{ik} z_n^{\ell,i}. \quad (5.5)$$

We obtain the mean $m_n^{\ell,k}$ and variance w_n^ℓ of the transition kernel in Equation (5.2) by marginalizing the OU process over one time step $\tau_n^\ell := t_n^\ell - t_{n-1}^\ell$, which can be done analytically³,

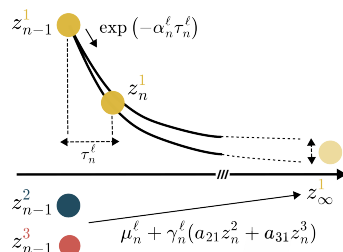


Figure 5.3: Conceptual sketch of the dynamics of knowledge states. Short-term: exponential decay

$$z_n^{\ell,k} = z_{n-1}^{\ell,k} \exp(-\alpha_n^\ell \tau_n^\ell)$$

Long-term: shifted by global structure

$$\tilde{\mu}_n^{\ell,k} := \mu_n^\ell + \gamma_n^\ell \sum_{i \neq k} a_{ik} z_n^{\ell,i}.$$

3: The variance is $w_n^\ell = (\sigma_n^\ell)^2(1 - e^{-2\alpha_n^\ell \tau_n^\ell})/(2\alpha_n^\ell)$ (Särkkä and Solin, 2019).

$$m_n^{\ell,k} = r_n^\ell z_{n-1}^{\ell,k} + (1 - r_n^\ell) \tilde{\mu}_n^{\ell,k}, \quad (5.6)$$

$$\text{with retention ratio } r_n^\ell := e^{-\alpha_n^\ell \tau_n^\ell} \in (0, 1). \quad (5.7)$$

As the time since the last interaction τ_n^ℓ grows, the retention ratio r_n^ℓ decreases exponentially with rate α_n^ℓ , and the knowledge state reverts to the long-term mean $\tilde{\mu}_n^{\ell,k}$, which partly depends on the learner's mastery of prerequisite KCs (Equation (5.5)). This balances short-term and long-term learning, reflecting empirical findings from memory research (Averell and Heathcote, 2011). The structural influences are accounted for in the dynamics of $z_n^{\ell,k}$, thus justifying the conditional independence assumed in Equation (5.2). A Gaussian initial prior $p_\theta(z_1^{\ell,k}) = \mathcal{N}(z_1^{\ell,k} | \bar{z}, w_1)$, where $\bar{z}, w_1 \in \mathbb{R}$ are part of the generative parameters θ , completes our dynamical model of knowledge states.

Learner-specific cognitive traits s . The dynamics of knowledge states (Equation (5.4)-Equation (5.6)) are parameterized by learner-specific cognitive traits $(\alpha_n^\ell, \mu_n^\ell, \sigma_n^\ell, \gamma_n^\ell)$, which we collectively denote s_n^ℓ . Specifically, α^ℓ represents the forgetting rate (Averell and Heathcote, 2011; Ebbinghaus, 1885), μ^ℓ (via $\tilde{\mu}_k^{\ell,n}$) captures long-term memory consolidation (Meeter and Murre, 2004) for practiced KCs and expected performance for novel KCs, σ^ℓ quantifies knowledge volatility, and γ^ℓ measures transfer ability (Bassett and Mattar, 2017) from knowledge of prerequisite KCs. These traits can develop during learning according to Equation (5.1)⁴, starting from a Gaussian prior $p_\theta(s_1^\ell) = \mathcal{N}(s_1^\ell | \bar{s}, R_1)$ where $\bar{s} \in \mathbb{R}^4$ and the diagonal matrices $H, R_1, R \in \mathbb{R}^{4 \times 4}$ are also part of the global parameters θ .

Shared prerequisite graph $\mathcal{A}$. In our model, prerequisite relations influence knowledge dynamics via the coupling introduced in Equation (5.5). We now discuss an appropriate parameterization for the weight matrix of the prerequisite graph, $\mathcal{A} := \{a^{ik}\}_{i,k \in 1:K}$. We assume that prerequisites are time- and learner-independent so that, in the spirit of collaborative filtering (Breese et al., 2013)⁵, we can pool evidence from all learners to estimate them. To prevent a quadratic scaling in the number of KCs, we do not directly model edge weights but derive them from KC embedding vectors u^k in lower dimension $u^k \in \mathbb{R}^D$ with $D \ll K$, collected in embedding matrix $U_{K \times D}$ ⁶. A basic integrity constraint for a connected pair is that dependence of KC i on KC k should trade off against that of k on i , i.e., no mutual prerequisites: $a^{ik} + a^{ki} = 1$. With this in mind, we exploit the factorization of a^{ik} introduced by (Lippe et al., 2021) in terms of a separate probability of edge existence $p(i \circ \circ k)$ and definite directionality $p(i \rightarrow k | i \circ \circ k)$:

$$\begin{aligned} a^{ik} &:= p(i \rightarrow k | i \circ \circ k) p(i \circ \circ k) \\ &= \sigma((u^i)^\top u^k) \sigma((u^i)^\top (M - M^\top) u^k), \end{aligned} \quad (5.8)$$

where the skew-symmetric combination $M - M^\top$ of a learnable matrix M prevents mutual prerequisites. Having presented the generative model, we now turn to inference and prediction.

5.3.3 Amortized inference with a neural network

We now describe how we learn the generative model parameters θ and how we infer the latent states s, z introduced in Section 5.3.2 using a neural network ("inference network"). Since learner-specific latent states s and z are deducible solely from limited individual data, we expect

4: Two rationales. First, traits *e.g.* forgetting rate can shift over time, so a dynamic model captures slow non-stationarities that a fixed parameter would miss. Second, the AR(1) form has three useful properties: $|H_{ii}| < 1$ guarantees stationarity with marginal variance $R_{ii}/(1 - H_{ii}^2)$, bounding trait drift; H_{ii} is the lag-1 autocorrelation, so learning H recovers how persistent each trait is; and the Gaussian form maintains conjugacy with the knowledge state transitions, keeping the ELBO terms for traits analytic.

5: In collaborative filtering, shared structure (*e.g.*, item similarity) is estimated by pooling data across many users, under the assumption that it reflects properties of the domain rather than any individual. Here the same logic applies: prerequisite relationships are a property of the knowledge domain, so all learners' interaction histories constitute evidence about the same shared graph.

6: In our datasets, K ranges from 102 (Assist17) to 835 (Junyi15), giving up to $K^2 \approx 700,000$ potential edge weights. With $D = 32$, the embedding matrix has $K \times D \approx 27,000$ parameters—a $\sim 25\times$ reduction.

Remark 5.1 (No mutual prerequisites) Skew-symmetry gives

$$(u^i)^\top (M - M^\top) u^k = -(u^k)^\top (M - M^\top) u^i.$$

Writing $x := (u^i)^\top (M - M^\top) u^k$:

$$\begin{aligned} p(i \rightarrow k | \cdot) + p(k \rightarrow i | \cdot) \\ = \sigma(x) + \sigma(-x) = 1. \checkmark \end{aligned}$$

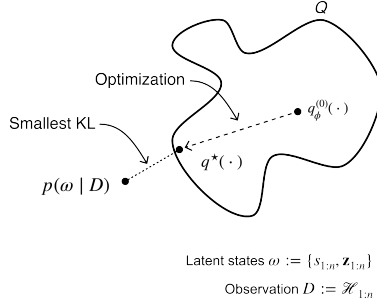


Figure 5.4: Conceptual sketch of VI.

Example 5.1 (Variational inference on state-space models) Variational inference (VI; Blei et al., 2017) approximates an intractable posterior $p_\theta(z|y)$ using a tractable family of distributions $q_\phi(z|y)$ parameterized by ϕ . Thus, inference involves learning both ϕ and θ by maximizing the *evidence lower bound* (ELBO) of the marginal likelihood (Beal and Ghahramani, 2003; Kingma and Welling, 2014):

$$\log p_\theta(y) \geq \text{ELBO}(\theta, \phi; y) \quad (5.9)$$

$$= \mathbb{E}_{q_\phi(z|y)} [-\log q_\phi(z|y)] \quad (5.10)$$

$$+ \log p_\theta(y, z) \quad (5.11)$$

In the context of SSMs, $y = y_{1:N}$ and $z = z_{1:N}$ are sequential observations and latent states. When transition and emission functions are linear Gaussian, Equation (5.1)-Equation (5.3) has analytical solutions. More generally, we can approximate the distribution of latent states with VI (Archer et al., 2015; Wang et al., 2022). For more scalable inference, VI can use an *inference network* (Kingma and Welling, 2014), which outputs a conditional distribution $q_\phi(z|y)$ and is parameterized by ϕ . Through amortized inference, the number of learnable parameters depends only on the network architecture, rather than the amount of data N (Zhang et al., 2018).

non-negligible uncertainty. This motivates our probabilistic treatment of these states using approximate Bayesian inference. By contrast, the model parameters θ (KC parameters U, M in Equation (5.8), transition parameters $\bar{s}, H, R_1, R$ in Equation (5.1), and $\bar{z}, w_1$ in Equation (5.2)) can be estimated from all learners, and we thus treat them as point-estimated parameters as described below (detailed derivation in Section B.1.) Here, without loss of generality, we show the inference for a single learner.

Inference on a fixed learning history

Here, we assume the full interaction history $\mathcal{H}_{1:N}^\ell$ is available for inferring the posterior over latents $p_\theta(z_{1:N}, s_{1:N}^\ell | y_{1:N}^\ell)$. We approach the problem using variational inference (VI). In VI, we select a distribution family q_ϕ with free parameters ϕ to approximate the posterior p_θ by minimizing their KULLBACK-LEIBLER divergence. This can only be done indirectly, by maximizing a lower bound to the marginal probability of the data, the *evidence lower bound* (ELBO). Here, we adopt the mean-field approximation $q_\phi(z_{1:N}, s_{1:N}^\ell | y_{1:N}^\ell) = q_\phi(z_{1:N}) q_\phi(s_{1:N}^\ell)$ and jointly optimize the generative θ and variational ϕ parameters using variational *expectation maximization* (EM; Attias, 1999; Beal and Ghahramani, 2003; Dempster et al., 1977). Motivated by real-world scalability, we introduce an inference network (see Figure B.1 for the architecture) to amortize the learning of variational parameters ϕ across learners, and we employ the *reparametrization trick* (Kingma and Welling, 2014) to optimize the single-learner ELBO:

$$\begin{aligned} \text{ELBO}^\ell(\theta, \phi) &= \mathbb{E}_{q_\phi(s_{1:N}^\ell)} [-\log q_\phi(s_{1:N}^\ell) + \log p_\theta(s_1^\ell) + \sum_{n=2}^N \log p_\theta(s_n^\ell | s_{n-1}^\ell)] \\ &+ \mathbb{E}_{q_\phi(z_{1:N}^\ell)} [-\log q_\phi(z_{1:N}^\ell) + \log p_\theta(z_1^\ell) + \sum_{n=1}^N \log p_\theta(y_n^\ell | z_n^\ell, x_n)] \\ &+ \mathbb{E}_{q_\phi(z_{1:N}^\ell) q_\phi(s_{1:N}^\ell)} [\sum_{n=2}^N \log p_\theta(z_n^\ell | z_{n-1}^\ell, s_n^\ell)]. \end{aligned} \quad (5.12)$$

The SSM emissions and transitions were introduced in Equation (5.1)-Equation (5.3), along with the respective initial priors. To allow for a diversity of combinations of learner traits to account for the data, we model the variational posterior across learners, $q_\phi(s_{1:N})$, as a mixture of Gaussians (see Section B.3).

Inference in continual learning

In real-world educational settings, a KT model must flexibly adapt its current variational parameters ϕ_n with newly available interactions $(x_{n+1}^\ell, t_{n+1}^\ell, y_{n+1}^\ell)$. Retraining on a fixed, augmented history $\mathcal{H}_{n+1}^\ell$ to obtain an updated ϕ_{n+1} is possible (Equation (5.12)), but expensive. Instead, in PSI-KT, we use the parameters ϕ_n of the current posterior $q_{\phi_n}(z_n^\ell, s_n^\ell)$ to form a next-time prior,

$$\tilde{p}(z_{n+1}^\ell, s_{n+1}^\ell) := \mathbb{E}_{q_{\phi_n}(z_n^\ell, s_n^\ell | y_{1:n}^\ell)} [p_\theta(s_{n+1}^\ell | s_n^\ell) p_\theta(z_{n+1}^\ell | s_{n+1}^\ell, z_n^\ell)]. \quad (5.13)$$

Due to the Bayesian nature of our model, we can now update this prior with the new evidence y_{n+1}^ℓ at time t_{n+1}^ℓ using *variational continual learning*

(VCL; Loo et al., 2020; Nguyen et al., 2017), i.e., by maximizing the ELBO:

$$\begin{aligned}
& \text{ELBO}_{\text{VCL}}^{\ell}(\theta, \phi_{n+1}) \\
&= \mathbb{E}_{q_{\phi_{n+1}}(s_{n+1}^{\ell})} [-\log q_{\phi_{n+1}}(s_{n+1}^{\ell})] \\
&+ \mathbb{E}_{q_{\phi_{n+1}}(z_{n+1}^{\ell})} [-\log q_{\phi_{n+1}}(z_{n+1}^{\ell}) + \log p_{\theta}(y_{n+1}^{\ell} | z_{n+1}^{\ell, x_{n+1}})] \\
&+ \mathbb{E}_{q_{\phi_{n+1}}(z_{n+1}^{\ell}, s_{n+1}^{\ell})} [\log \tilde{p}(z_{n+1}^{\ell}, s_{n+1}^{\ell})]. \tag{5.14}
\end{aligned}$$

Maximizing this $\text{ELBO}_{\text{VCL}}^{\ell}$ allows us to update the parameters ϕ_{n+1} based on a new interaction $(x_{n+1}^{\ell}, t_{n+1}^{\ell}, y_{n+1}^{\ell})$ directly from the previous parameters ϕ_n , i.e., without retraining.

5.3.4 Predictions

To predict a learner's performance on KC x_{n+1}^{ℓ} at t_{n+1}^{ℓ} , we take the current variational distributions over s_n^{ℓ} and z_n^{ℓ} and transport them forward by analytically convolving them with the respective transition kernels (Equation (5.1) and Equation (5.2)). We then draw $z_{n+1}^{\ell, x_{n+1}}$ from the resulting distribution, and predict the outcome $\hat{y}_{n+1}^{\ell}$ by Equation (5.3). When predicting multiple steps ahead, we repeat this procedure without conditioning on any of the previously predicted $\hat{y}_{n+m}^{\ell}$.

5.4 Results

We argue above that KT for personalized education must predict accurately, scale well with new data, and provide interpretable representations. We now empirically assess these desiderata, comparing PSI-KT with up to 8 baseline models across three datasets from online education platforms. Concretely, we evaluate (i) prediction accuracy, quantifying both within-learner prediction and between-learner generalization (Section 5.4.2), (ii) scalability in a continual learning setting (Section 5.4.3), and (iii) interpretability of learner representations and prerequisite relations (Section 5.4.4).

5.4.1 Datasets

Assistments and Junyi Academy are non-profit online learning platforms for pre-college mathematics. We use Assistments' 2012 and 2017 datasets (Assist12 and Assist17) and Junyi's 2015 dataset (Junyi15; Chang et al., 2015), which in addition to interaction data, provides human-annotated KC relations (see Table 5.1 and Section B.2.2 for details).

We select HLR from Duolingo and PPE as two influential psychologically-informed regression models. From the models that use learnable representations, we include two established deep learning benchmarks, DKT and DKTF, which capture complex dynamics via LSTM networks, as well as the interpretability-oriented QIKT.

	Assist12	Assist17	Junyi15
# Learners L	46,674	1,709	247,606
# KCs K	263	102	722
# Interactions	3.5M	0.9M	26M
KC pre	✗	✗	✓
KC sim	✗	✗	✓

Table 5.1: Dataset statistics for the three online learning platforms used in evaluation. Assist12 and Assist17 are from ASSISTments; Junyi15 from Junyi Academy. Junyi15 uniquely provides human-annotated KC prerequisite and similarity graphs, used to evaluate graph interpretability (Section 5.4.4).

Table 5.2: Prediction accuracy. FT indicates additional fine-tuning and $\uparrow$ indicates larger values are better. The **best model performance** is in bold and the 2nd best is underlined.

Dataset	Experiment	HLR	PPE	DKT	DKTF	HKT	AKT	GKT	QIKT	PSI-KT
Assist12	Within $\uparrow$	.54 _{.03}	.65 _{.01}	.65 _{.03}	.60 _{.01}	.55 _{.01}	<u>.67</u> _{.02}	.63 _{.03}	.63 _{.03}	.68 _{.02}
	Between $\uparrow$	.50 _{.03}	.50 _{.02}	.55 _{.02}	.51 _{.01}	.54 _{.00}	.58 _{.02}	.61 _{.02}	<u>.60</u> _{.02}	.61 _{.03}
	w/ FT $\uparrow$	.52 _{.02}	.53 _{.01}	.58 _{.00}	.55 _{.01}	.55 _{.00}	<u>.61</u> _{.00}	.62 _{.02}	.60 _{.03}	.62 _{.02}
Assist17	Within	.45 _{.01}	.53 _{.02}	.57 _{.02}	.53 _{.03}	.52 _{.03}	.56 _{.02}	.56 _{.04}	.58 _{.02}	.63 _{.02}
	Between	.33 _{.03}	<u>.51</u> _{.02}	.51 _{.00}	.48 _{.00}	<u>.51</u> _{.02}	.47 _{.01}	.53 _{.02}	.50 _{.02}	.53 _{.02}
	w/ FT	.41 _{.04}	.51 _{.00}	.51 _{.03}	.53 _{.01}	.51 _{.03}	.51 _{.02}	<u>.54</u> _{.03}	.51 _{.04}	.56 _{.02}
Junyi15	Within	.55 _{.02}	.66 _{.03}	.79 _{.03}	.78 _{.01}	.63 _{.02}	.81 _{.02}	.78 _{.02}	.81 _{.02}	.83 _{.02}
	Between	.48 _{.02}	.55 _{.02}	.76 _{.00}	.76 _{.02}	.61 _{.01}	.73 _{.01}	<u>.77</u> _{.03}	.76 _{.03}	.79 _{.03}
	w/ FT	.52 _{.00}	.65 _{.03}	.81 _{.01}	.84 _{.01}	.64 _{.03}	<u>.83</u> _{.00}	.79 _{.03}	.80 _{.03}	.80 _{.02}

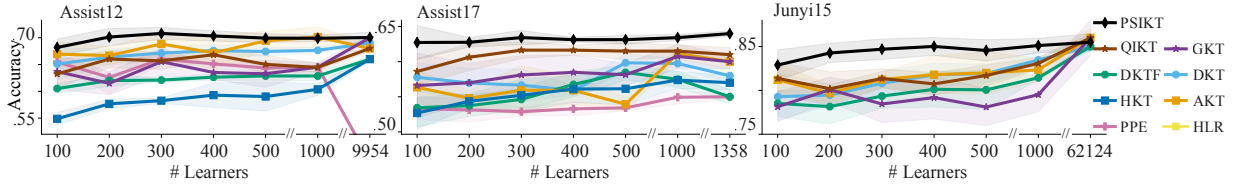


Figure 5.5: Within-learner prediction performance (mean $\pm$ SEM) as a function of cohort sizes from 100 to the maximum available in each dataset (we omit HLR for legibility; see Table 5.2.)

5.4.2 Prediction and generalization performance

In our evaluations, we mainly focus on prediction and generalization when training on 10 interactions from up to 1000 learners. Good KT performance with little data is key in practical ITS to minimize the number of learners on an experimental treatment (principle of equipoise, similar to medical research; Burkholder, 2021), to mitigate the cold-start problem, and to extend the usefulness of the model to classroom-size groups. To provide ITS with a basis for adaptive guidance and long-term learner assessment, we always predict the 10 next interactions. Figure 5.5 shows that PSI-KT's *within-learner prediction performance* is robustly above baselines for all but the largest cohorts ($> 60k$ learners, Junyi15), where all deep learning models perform similarly. The advantage of PSI-KT comes from its combined modeling of KC prerequisite relations and individual learner traits that evolve in time (see Figure B.6 for ablations). The *between-learner generalization* accuracy of the models above, when tested on 100 out-of-sample learners, is shown in Table 5.2, where fine-tuning indicates that parameters were updated using (10-point) learning histories from the unseen learners. PSI-KT shows overall superior generalization except on Junyi15 (when fine-tuning).

5.4.3 Scalability in continual learning

In addition to training on fixed historical data, we also conduct experiments to demonstrate PSI-KT's scalability when iteratively retraining on additional interaction data from each learner. This parallels real-world educational scenarios, where learners are continuously learning. Each model is initially trained on 10 interactions from 100 learners. We then incrementally provide one data point from each learner, and evaluate the training costs and prediction accuracy. Figure 5.6 shows PSI-KT requires the least retraining time, retains the best prediction accuracy, and thus achieves the most favorable cost-accuracy trade-off (details in Section B.4.3).

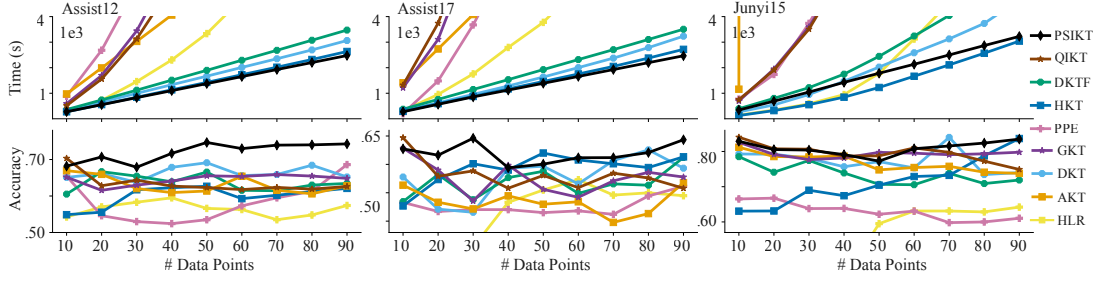


Figure 5.6: Continual learning. (Top) Cumulative training time. (Bottom) Prediction accuracy on the next 10 time steps. We omit results when time is above, or accuracy is below, the range of the axes.

5.4.4 Interpretability of representations

We now evaluate the interpretability of both learner-specific cognitive traits s^ℓ and the prerequisite graphs $\mathcal{A}$. We first show that our model captures learner-specific and disentangled traits that correlate with behavior patterns. Next, we show that our inferred graphs best align with ground truth graphs, and the edge weights predict causal support on downstream KCs.

Learner-specific cognitive traits

For each learner, PSI-KT infers four latent traits, each with a clear dynamical role specified by the OU process (Equation (5.5)-Equation (5.6)). In contrast, high-performance baselines (AKT, DKT, and DKTF) describe learners via 16-dimensional embeddings solely constrained by network architecture and loss minimization. Another model QIKT constructs 3-dimensional embeddings with each element connected to scores of knowledge acquisition, knowledge mastery, and problem-solving. We collectively refer to these learner-specific variables as *learner representations*. Here, we empirically show that PSI-KT representations provide superior interpretability. We ask that learner representations be (i) *specific* to individual learners; (ii) *consistent* when trained on partial learning histories; (iii) *disentangled* (i.e., component-wise meaningful, as in Bengio et al., 2013); and (iv) *operationally interpretable*, so that they can be used to personalize future curricula. We evaluate desiderata 1-3 with information-theoretic metrics (Table 5.3; see Section B.5 for details), and desideratum 4 with regressions against behavioral outcomes (Table 5.4).

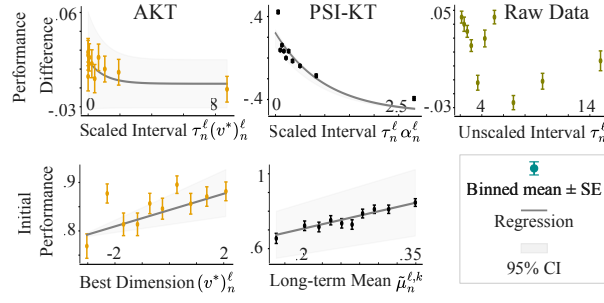
Metric	Dataset	Baseline	PSI-KT
Specificity $MI(s; \ell) \uparrow$	Assist12	<u>8.8</u>	<u>8.4</u>
	Assist17	<u>10.1</u>	<u>10.0</u>
	Junyi15	<u>13.5</u>	<u>14.4</u>
Consistency ⁻¹ $\mathbb{E}_{\ell_{\text{sub}}} MI(s^\ell; \ell_{\text{sub}}) \downarrow$	Assist12	<u>12.2</u>	<u>7.4</u>
	Assist17	<u>6.4</u>	<u>6.4</u>
	Junyi15	<u>7.7</u>	<u>5.0</u>
Disentanglement $D_{KL}(s \ell) \uparrow$	Assist12	<u>2.3</u>	<u>7.4</u>
	Assist17	<u>0.6</u>	<u>8.4</u>
	Junyi15	<u>5.0</u>	<u>11.5</u>

Table 5.3: Information-theoretic evaluation of learner representations against the best baseline per dataset. Specificity ($\uparrow$) measures how uniquely representations identify individual learners; consistency ($\downarrow$) measures stability across partial interaction histories; disentanglement ($\uparrow$) measures whether each dimension is individually informative about learner identity.

Specificity, consistency, and disentanglement. Learner representations s should be maximally *specific* about learner identity ℓ , which can be quantified by the mutual information $MI(s; \ell) = H(s) - H(s | \ell)$ being high, where H denotes (conditional) entropy. Additionally, when we infer representations $s^{\ell_{\text{sub}}}$ from different subsets of the interactions of a fixed learner, they should be *consistent*, i.e., each $s^{\ell_{\text{sub}}}$ should be minimally informative about the chosen subset (averaged across subsets), such that

Figure 5.7: Operational interpretability of representations, Junyi15 dataset. See text for axes labels and Section B.5.4 for additional results.

Table 5.4: Coefficients and p -values of regressions relating $\exp(-\alpha_n^\ell \tau_n^\ell)$ and $\bar{\mu}_n^{\ell,k}$ to unseen behavioral data across datasets.



Behavioural signature	Dataset	Best Baseline	PSI-KT
Performance difference	Assist12	0.01, .67	0.30 , <.001
	Assist17	-0.03, .30	0.56 , <.001
	Junyi15	0.03, .06	0.72 , <.001
Initial performance	Assist12	0.04, .01	0.54 , <.001
	Assist17	0.05, .01	3.70 , <.001
	Junyi15	0.04, .02	0.92 , <.001

$\mathbb{E}_{\ell_{\text{sub}}} \text{MI}(s^\ell; \ell_{\text{sub}}) = \mathbb{E}_{\ell_{\text{sub}}} [H(s | \ell) - H(s | \ell_{\text{sub}})]$ should be low. Note that sequential subsets are unsuitable for this evaluation, since representations evolve in time to track learners' progression. Instead, we define subsets as groups of KCs whose average presentation time is approximately uniform over the duration of the experiment (see Section B.5.1 for details). Lastly, learner representations should be *disentangled*, such that each dimension is individually informative about learner identity. We measure disentanglement with $D_{\text{KL}}(s || \ell) := H(s) - H(s | \ell)_{\text{diag}}$, a form of specificity that ignores correlations across s^ℓ dimensions by estimating the conditional entropy only with diagonal covariances.

In empirical evaluations (Table 5.3), PSI-KT's representations offer competitive specificity despite being lower-dimensional, and outperform all baselines in consistency and disentanglement. While disentanglement aids interpretability (Freiesleben et al., 2022), it does not itself entail domain-specific meaning for representational dimensions. We now demonstrate that PSI-KT representations correspond to clear behavioral patterns, which is crucial for future applications in educational settings.

Operational interpretability. Having shown that PSI-KT captures specific, consistent, and disentangled learner features, we now investigate whether these features relate to meaningful aspects of future behavior, which would be useful for scheduling operations for ITS. We indeed find that the learner representations of PSI-KT forecast interpretable behavioral outcomes, such as performance decay or initial performance on novel KCs. Concretely, consider the observed *one-step performance difference* $\Delta y_n^\ell := y_n^\ell - y_{n-1}^\ell$. We expect it to be lower for longer intervals $\tau_n^\ell = t_n^\ell - t_{n-1}^\ell$ due to forgetting. However, we recognize no clear trend when plotting Δy_n^ℓ over τ_n^ℓ for the Junyi15 dataset (Figure 5.7, top right). We can explain this observation because different learners forget on different time scales. Plotting the same test data instead over scaled intervals $\tau_n^\ell \alpha_n^\ell$ (Figure 5.7, top center) shows a clear trend against an exponential fit (solid line) with less variability, demonstrating that α_n^ℓ (derived from past data only) adjusts for individual learner characteristics and can be interpreted as a personalized rate of forgetting. Here, the choice of the factor α_n^ℓ is motivated by our inductive bias (Equation (5.4)). The trend is much less clear for all baselines: Figure 5.7 (top left) uses the best fitting component across all learner representations from all baselines (full results in Figure B.2 in Section B.5.4). Analogously, when

Table 5.5: (Left) Alignment of inferred graphs with annotated graphs for the Junyi15 dataset. (Right) Regression coefficients and p -values relating causal support to inferred edge probabilities. All baseline models either lack significance or negatively predict causal support (Figure B.5).

Metric	MRR $\uparrow$	JS expert $\uparrow$	JS crowd $\uparrow$	nLL $\downarrow$	coefficient $\uparrow$, p -value $\downarrow$		
Dataset	Junyi15				Assist12	Assist17	Junyi15
Best Baseline	.0082	.0015	.0047	3.03	1.05, .253	0.22, .792	0.42, .593
PSI-KT	.0086	.0019	.0095	4.11	1.15 , .003	0.28 , <.001	0.97 , <.001

we consider *initial performance on a novel KC*, we find for PSI-KT that $\tilde{\mu}_n^{\ell,k}$ (which aggregates mastery of prerequisites for KC k at time t_n , see Equation (5.5)) explains it better than the best baseline Figure 5.7 (bottom panels). Table 5.4 shows that these superior interpretability results are significant and hold across all datasets. In Section B.5.4, we discuss two more behavioral signatures (performance variability and prerequisite influence) and show they correspond to the remaining components γ_n^ℓ and σ_n^ℓ .

Prerequisite graph

PSI-KT infers a prerequisite graph based on all learners' data, which helps it to generalize to unseen learners. Beyond helping prediction, reliable prerequisite relations are an essential input for curriculum design, motivating our interest in their interpretability. Figure 5.8a shows an exemplary inferred subgraph with the prerequisites of a single KC. To quantitatively evaluate the graph, we (i) measure the alignment of the inferred vs. ground-truth graphs and (ii) correlate inferred prerequisite probability with a Bayesian measure of causal support obtained from unseen behavioral data.

Alignment with ground-truth graphs. We analyze the Junyi15 dataset, which uniquely provides human-annotated evaluations of prerequisite and similarity relations between KCs. We discuss here the alignment of prerequisites and leave similarity for Section B.6. The Junyi15 dataset provides both an expert-identified prerequisite for each KC, and crowd-sourced ratings (6.6 ratings on average on a 1 – 9 scale). To compare with expert annotations, we compute the rank of each expert-identified prerequisite relation $i \rightarrow k$ in the relevant sorted list of inferred probabilities $\{a^{ik}\}_{j=1}^K$ and take the harmonic average (mean reciprocal rank, MRR; Yang et al., 2014). Next, we compute the negative log-likelihood (nLL) of inferred edges a^{ik} using a Gaussian estimate of the (rescaled) crowd-sourced ratings for the $i \rightarrow k$ KC pair. We finally calculate the Jaccard similarity (JS) between the set of inferred edges ($a^{ik} > 0.5$) and those identified by experts as well as crowd-sourced edges with average ratings above 5. The results in Table 5.5 (left columns) consistently highlight PSI-KT's superior performance across all criteria (see Section B.6.1 for details).

Causal support across consecutive interactions. For education applications, we are interested in how KC dependencies impact learning effectiveness. If KC i is a prerequisite of KC k , mastering KC i contributes to mastering KC k , indicating a causal connection. In this analysis, we show that inferred edge probabilities a^{ij} (Equation (5.8)) correspond to causal support $\text{support}_{i \rightarrow k}$ (Equation (5.15)), derived from behavioral data through Bayesian causal induction (Griffiths and Tenenbaum, 2009). Specifically, we model the relationship between a candidate cause C and effect E , i.e., a pair of KCs in our case, while accounting for a constant

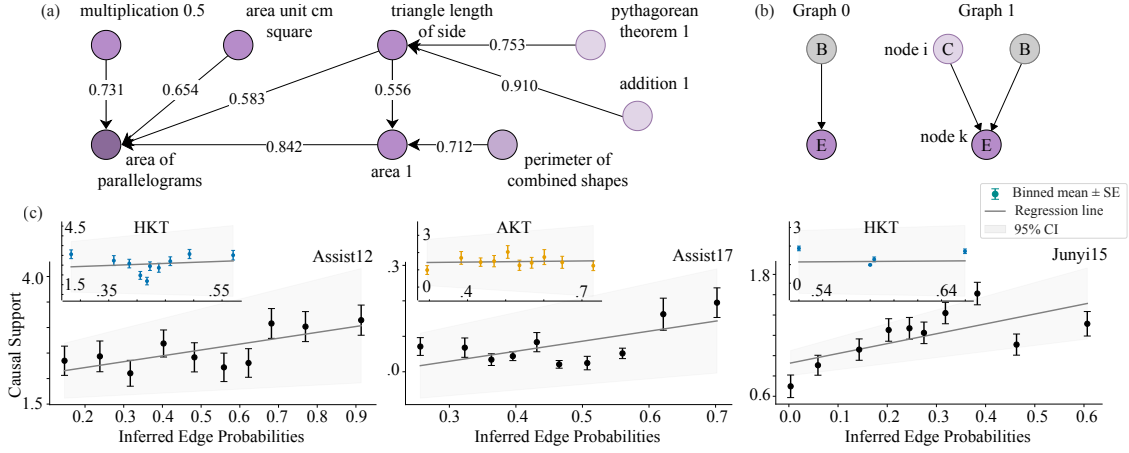


Figure 5.8: Graph interpretability. (a) Subgraph inferred by PSI-KT on the Junyi15 dataset, showing prerequisites of target KC ‘area of parallelograms’. (b) Hypothesized causal graphs, where Graph 1 assumes a causal relationship exists from KC i to KC k , while Graph 0 is the null hypothesis. (c) Regression of edge probabilities against causal supports. Insets show the best baseline model.

background cause B , representing the learner’s overall ability and the influences of other KCs. We consider two hypothetical causal graphs, where Graph 0 $G_{i \rightarrow k}$ represents the null hypothesis of no causal relationship, and Graph 1 $G_{i \rightarrow k}$ assumes the causal relationship exists, i.e. correct performance on KC i causally supports correct performance on KC k (Figure 5.8b). We estimate causal support for each pair of KCs $i \rightarrow k$ based on all consecutive interactions in the behavioral data $\mathcal{H}$ from KC i at time t_n to KC k at time t_{n+1} , as a function of the difference in log-likelihoods of the two causal graphs (see Section B.6.3 for details):

$$\text{support}_{i \rightarrow k} := \log P(\mathcal{H} | G_{i \rightarrow k}) - \log P(\mathcal{H} | G_{i \nrightarrow k}). \quad (5.15)$$

We then use regression to predict $\text{support}_{i \rightarrow k}$ as a function of edge probabilities a^{ij} inferred from different models. The results are visualized in Figure 5.8c and summarized in Table 5.5 (right). The larger coefficients indicate that our inferred graphs possess superior operational interpretability (Section 5.4.4).

5.5 Discussion

5.5.1 Findings

PSI-KT’s empirical advantage is clearest where data is scarce. It achieves competitive performance with 100-500 learners, a regime where deep learning baselines require at least 60k learners to reach comparable accuracy. This is a direct advantage of structural prior information. By specifying that knowledge dynamics follow an Ornstein-Uhlenbeck(OU) process coupled through a prerequisite graph, the model needs far fewer examples to identify the relevant dimensions of variation. The cost is symmetrical—PSI-KT can only capture patterns consistent with its generative assumptions. The flexibility advantage of DKT or AKT is real in large-data regimes, where arbitrary patterns can be learned from data alone. An open question for future KT research is how to retain PSI-KT’s continual-learning and interpretability properties while extending its predictive capacity beyond this data-scarce regime.

The interpretability results are not incidental—they follow from the

generative commitment. PSI-KT's representations are specific, consistent, and disentangled because the inference procedure has no degrees of freedom to spread information across dimensions arbitrarily: each trait has a designated role in the generative process, and the posterior over traits must explain the data through that role. Baseline embeddings are constrained only by prediction loss, so information distributes across dimensions in whatever way minimizes loss—producing representations that are individually uninformative about any single behavioral dimension. The trade-off is that PSI-KT's representations are expressive only within the language of its generative model. This is a feature for educational applications where representations must be communicable to teachers and learners—but it does constrain the class of learners and domains the model can accurately describe.

5.5.2 Limitations

Regarding the predefined structures, PSI-KT uses an analytically marginalizable OU process for knowledge states, resulting in an exponential forgetting law, similar to most recent KT literature. Future work should support ongoing debates in cognition by offering alternative modeling choices for memory decay (e.g., power-law; Wixted and Ebbesen, 1997), thus facilitating empirical studies at scale. Meanwhile, PSI-KT normalizes reciprocal dependencies in the prerequisite graph but does not enforce global structural constraints such as acyclicity. Imposing such constraints—for example, via differentiable DAG regularization (Zheng et al., 2018)—could improve both inference efficiency and the interpretability of the resulting graph, though at increased computational cost during training. Although we designed PSI-KT with general structured domains in mind, our empirical evaluations were limited to mathematics learning by dataset availability. We highlight the need for more diverse datasets for structured KT research to strengthen representativeness in ecologically valid contexts.

In general, the most natural extension is coupling PSI-KT's diagnostic capabilities with a prescriptive component for curriculum optimization—pairing the inferred knowledge states and prerequisite graph with a sequential decision process that determines what to teach next. The model's continual learning properties make it compatible with online reinforcement learning approaches (Raffin et al., 2021), and the interpretable trait parameters could serve directly as the state representation for a personalized tutoring policy, closing the loop between inference and intervention.

5.6 Summary

PSI-KT shows that the structured representations posited by cognitive theories of human learning—prerequisite relations between KCs, individualized cognitive traits, mastery trajectories—are not artefacts of controlled experiments but recoverable signals in large-scale behavioral data. Where the preceding chapter observed representation construction directly in the laboratory, this chapter demonstrates its effects in the wild: the traces that learners leave in naturalistic interaction data carry enough structure to support interpretable, predictive inference at scale. Both forms of evidence converge on the same claim—that human learners

build structured representations, and that the history of what they have constructed shapes what they can efficiently acquire next.

This convergence also surfaces questions that inference alone cannot answer. PSI-KT can tell us *what structure exists* in a domain and *how individuals differ* in their learning dynamics, but it does not tell us *what a learner should study next* or *what drives an agent to seek out new knowledge in the first place*. These are questions of *curation*—the active management of representational resources—and they are the subject of [Part III](#). [Chapter 6](#) examines how intrinsic motivation drives agents to explore and acquire representations, asking whether the urge to learn is better understood as curiosity about the unknown or as competence over what can be controlled. [Chapter 7](#) then asks how an agent should select and maintain its internal knowledge representations, introducing empowerment-based criteria for deciding what is worth keeping. The interpretable traits inferred by PSI-KT—forgetting rate, transfer ability, and the rest—offer a natural bridge: they are precisely the kind of individualized parameters whose active curation could determine whether learning is effective or wasteful.

Finally, the prerequisite graph inferred by PSI-KT foreshadows the communication themes of [Part IV](#). If a tutor knows the relational structure of a domain, they can transmit not just isolated facts but the organizational scaffolding that makes future acquisition more efficient—a form of library communication that it will develop as resource-rational pedagogical explanation.

Part III.

Curate: how structured knowledge is
selected

Curiosity is not free: how exploration shapes representation

6.

“Like the entomologist in pursuit of brightly colored butterflies, my attention hunted, in the flower garden of the gray matter, cells with delicate and elegant forms, the mysterious butterflies of the soul, whose beating of wings may one day reveal to us the secrets of the mind.”

— Santiago Ramón y Cajal, *Recollections of My Life*, 1917

The preceding chapters established that structured representations—prerequisite graphs, path-dependent programs—are recoverable from behavioral data, and that the history of what a learner has encountered shapes what they can efficiently learn next. But those accounts were retrospective: given the traces a learner has already left, we infer what they have built. What they do not address is the generative side—*what drives a learner to seek out new experiences and build representations in the first place?* Without such a mechanism, library construction is passive and contingent solely on externally provided experience, while “curation” requires a learner that actively selects what is worth representing.

Humans seek out novelty (Berlyne, 1960), are drawn to information gaps (Loewenstein, 1994), and are intrinsically motivated by competence and causal control (Deci and Ryan, 1985; White, 1959). These drives form a rough spectrum from pure *curiosity*—seeking the unknown—to pure *competence*—seeking the control. Yet their precise consequences remain difficult to isolate in human studies, where motivation, learning history, and environment cannot be held constant simultaneously.

We operationalize this spectrum computationally, formalizing three signals—*novelty*, *information gain*, and *empowerment*—and studying them in two agent architectures. A Tabular agent with fixed representations isolates each signal’s behavioral signature under controlled conditions; a Dreamer agent that learns latent representations from experience closes the loop between its motivation and its model.

The experiments establish that drive selection has direct and qualitatively different representational consequences, and that combining information gain with empowerment resolves the failures of each drive in isolation. No single intrinsic drive is a sufficient policy for selecting what is worth representing, i.e., for *Curation*. The following chapter builds on this finding, internalizing the empowerment criterion—asking not which external states give an agent control, but which internal representational structures maximize the agent’s capacity to predict, distinguish, and act.

The chapter is organized as follows. [Section 6.1](#) motivates the technical problem and connects it to the field. [Section 6.2](#) situates the three intrinsic signals in the relevant literature. [Section 6.3](#) describes the environments, agent architectures, and motivation formalizations. [Section 6.4](#) examines the Tabular agent, using fixed representations to isolate the behavioral signature of each signal. [Section 6.5](#) examines the Dreamer agent, revealing how motivation and representation co-evolve when the feedback loop between them is closed. [Section 6.6](#) interprets the findings and their implications for the curation framework.

6.1 Introduction	100
6.2 Background	101
6.3 Methods	103
6.4 Experiment 1: Tabular Agent	110
6.5 Experiment 2: Dreamer Agent	114
6.6 Discussion	122
6.7 Summary	123

Code and experiments available at the
Github repository [dreamer-curiosity](#)



6.1 Introduction

Picture a child playing with two brand-new toys. One lights up predictably whenever a button is pressed; the other flickers randomly, indifferent to the child's actions. Which toy captivates the child longer? The answer depends not just on novelty, but on a deeper interplay between two fundamental drives: *curiosity*, which compels agents to seek new knowledge, and *competence*, which drives agents to leverage what they know in order to influence their environment (Meltzoff et al., 2012; Nussenbaum and Hartley, 2019; Schmidhuber, 1991). Intelligent agents—from human infants to autonomous robots—must be drawn simultaneously to the unknown, in the quest to reduce uncertainty, and to the predictable, in the desire to exert causal control.

These two broad families capture the drives: *curiosity*-based signals (novelty, information gain) reward uncertainty reduction and world-model refinement (Gottlieb et al., 2013; Modirshanechi et al., 2022; Nelson, 2005; Ten et al., 2024), while *competence*-based signals (empowerment, skill acquisition) reward expanding the capacity to predict and control (Aubret et al., 2023; Gopnik, 2024; Klyubin et al., 2005; Poli et al., 2020; Salge et al., 2014). What makes all three difficult to characterize is that each is mediated by the agent's *world model* (Ha and Schmidhuber, 2018b)—the representation that translates raw experience into signal values and signal values into exploration behavior.

Most existing reinforcement learning (RL) agents hold this intermediary fixed—they treat the agent's state representation as given and ask which signal best directs exploration within that space (Burda et al., 2018; Dulac-Arnold et al., 2021; Gruaz et al., 2024; Pathak et al., 2017). This is a useful baseline, but it breaks the feedback: when the world model is learned alongside the policy, the signal shapes the data, the data shapes the model, and the model reshapes the signal. Whether this feedback loop amplifies or undermines the intended exploration behavior—and whether the answer differs across the curiosity-competence spectrum—is the question this chapter puts to empirical test.

This chapter investigates that feedback loop through a controlled two-experiment design. We formalize three intrinsic motivation signals, spanning the *novelty*, *information gain*, and *empowerment*, and study them under two complementary conditions. In **Experiment 1**, a tabular agent with fixed, hand-crafted state representations allows the signals to be examined in isolation: any behavioral difference across conditions must arise from the signal itself, not from how the agent perceives the world. This controlled baseline reveals what each drive *asks for*: the precise mechanism of each signal's failure mode, the uniform-prior trap that renders empowerment impotent in novel environments, and the complementarity that emerges when information gain and empowerment are combined. In **Experiment 2**, a world-model agent (Hafner et al., 2023) learns latent representations from pixel observations, closing the feedback loop between motivation and representation. This setting reveals a qualitatively different class of phenomena: pseudo-novelty arising from representational drift, aleatoric fixation in information gain that resists decay, and empowerment estimates that depend on world model quality—a chicken-and-egg problem that makes competence-seeking ineffective before the model it requires has been built.

6.2 Background

6.2.1 Intrinsic motivations in psychology and RL

What curiosity is and when it peaks. Curiosity¹ has long been recognized as a fundamental epistemic drive, but its modern scientific treatment begins with Berlyne (1950) and Berlyne (1960). Berlyne’s central claim was that curiosity is mediated by an organism’s *arousal* level and a set of *collative properties* of stimuli, i.e., novelty, complexity, incongruity, and conflict. These properties generate arousal because they create conceptual conflict—the simultaneous activation of incompatible internal responses. Crucially, Berlyne proposed an *optimal arousal hypothesis*: curiosity is strongest at intermediate levels of arousal, producing the now-familiar inverted-U relationship between stimulus complexity and interest. A stimulus that is entirely familiar evokes no curiosity; one that is entirely unpredictable evokes aversion rather than engagement; the sweet spot lies between. Berlyne also introduced a four-way taxonomy distinguishing two orthogonal dimensions:

By modality

- **Epistemic curiosity**: the desire to acquire knowledge, typically expressed through reading, questioning, or reasoning.
- **Perceptual curiosity**: the drive to seek sensory stimulation, expressed through looking, listening, or exploring the immediate environment.

By scope

- **Specific curiosity**: directed at a particular gap or source of uncertainty (*what does this unfamiliar word mean?*).
- **Diversive curiosity**: a more general seeking of stimulation to escape boredom, with no specific informational target.

Information-based refinement: from arousal to bits. Loewenstein’s *information-gap theory* (Loewenstein, 1994) holds that curiosity is triggered by the awareness of a gap between what one knows and what one wishes to know—a motivational tension resolved only by acquiring the missing information. The *learning-progress hypothesis* (Schmidhuber, 1991) shifts from static awareness to dynamic anticipation: curiosity is strongest where the predictive model is uncertain *and* improvable, so irreducibly stochastic stimuli evoke no sustained curiosity—this is the mechanism behind the noisy-TV failure mode (Oudeyer and Kaplan, 2007). Dubey and Griffiths (Dubey and Griffiths, 2020) unified these accounts, showing that novelty, complexity, and information gain are all instances of maximizing expected bits learned per observation². For a taxonomy of information-gain variants and their computational implementations, see Section C.2. Information is the curiosity-based signal studied in this chapter precisely because of this theoretical advantage: its sensitivity to reducible rather than irreducible uncertainty gives its exploration behavior a predictable structure under fixed representations. But the same property defines a specific failure mode in Section 6.5: when the world model misclassifies learnable structure as irreducible noise, information gain fixates on stochastic regions rather than retreating from them—a failure that cannot occur when representations are fixed.

Styles of seeking as signatures. A parallel perspective asks not *why* agents are curious but *how* curiosity manifests in behavior. Zurn and Bassett (2022) identified three archetypal styles (Lydon-Staley et al., 2021; Patankar et al., 2022). When exploring a knowledge space, represented by a network of concepts, *hunters* pursue targeted, closely connected concepts, building tight networks with high clustering coefficients and short characteristic path lengths: they know what they are looking for and follow the trail systematically; *busybodies* sample diverse, disconnected topics, producing loose networks with long path lengths and

1: Augustine named *curiositas*—the restless craving for spectacle and novelty—a form of lust, ranking it alongside pride and sensual desire as a cardinal distraction from wisdom.

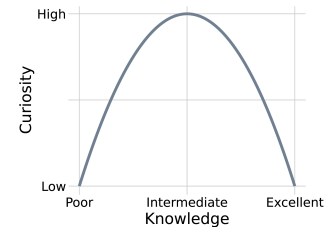


Figure 6.1: Inverted-U shaped relationship between knowledge and curiosity (Ten et al., 2025).

2: This unification formally derives Berlyne’s inverted-U: random stimuli and familiar stimuli both yield near-zero bits, while stimuli of intermediate complexity maximize expected information per observation.



Figure 6.2: Three curiosity styles as knowledge networks. Hunters build tight, locally connected networks; busybodies construct loose, wide-ranging ones; dancers bridge otherwise disconnected modules (Zhou et al., 2020).

low clustering: their seeking is broad but shallow, jumping between distant subjects without dwelling; *dancers* take creative leaps across typically siloed domains, producing hybrid knowledge structures that break disciplinary boundaries: their seeking is neither targeted nor random but generative, forming new conceptual connections. These styles map onto Berlyne’s specific-diversive dimension: hunters approximate specific epistemic curiosity, busybodies approximate diversive perceptual curiosity, and dancers occupy a space the specific-diversive distinction does not fully capture. This behavioral distinction helps interpret which intrinsic reward signals produce which exploration regimes—a question we return to in [Section 6.4](#).

Intrinsic motivation for exploration. The behavioral signatures above arise from drives that are, in humans, entirely endogenous: no external reward is needed to sustain a hunter’s systematic search or a busybody’s wide-ranging exploration. This is precisely one of the bottlenecks in reinforcement learning (RL). Standard RL assumes dense, task-specific reward signals ([Section 3.2](#)), but many interesting domains—open-ended environments, developmental robotics, unsupervised skill acquisition—only have sparse or even lack external rewards. An RL agent that cannot explore without reward feedback consequently cannot build the world model needed to exploit a task signal when one eventually appears (Dulac-Arnold et al., 2021; Ocana et al., 2023). The cognitive developmental literature suggests a solution: children learn to walk, speak, and manipulate objects in reward-sparse environments, sustained by internal drives—the discomfort of an unresolved information gap, the pleasure of mastery, the pull of the controllable (Ryan and Deci, 2000; White, 1959). Formalizing these drives as reward signals that RL agents can optimize is the program of intrinsic motivation research (Gruaz et al., 2024; Lidayan et al., 2025; Modirshanechi et al., 2022; Oudeyer and Kaplan, 2007; Schmidhuber, 1991).

Two major families of signals have emerged, mirroring the curiosity-competence distinction. *Curiosity-based* signals reward information acquisition and uncertainty reduction—novelty, information gain, prediction error—formalizing the information-gap and learning-progress traditions above. *Competence-based* signals reward the capacity to influence and control outcomes—empowerment, skill acquisition, causal control—formalizing the effectance drive in White (1959) and the competence need in Deci and Ryan (1985). The formal definitions of the three signals studied in this chapter are given in [Section 6.3](#).

6.2.2 The perception-action-representation loop

Having outlined the major interpretations of intrinsic motivation, we now consider the feedback process through which it operates in a learning agent. A central insight from cybernetics, ecological psychology, and cognitive science is that perception and action are not cleanly separable sequential stages, but form a continuous sensorimotor cycle (Gibson, 2014; O’regan and Noë, 2001; Varela et al., 1991). What an agent does partly determines what it can observe next, and what it observes in turn constrains which actions are possible, useful, or available. In agents with learned internal models, this coupling extends one level further: the representations through which observations are interpreted are themselves shaped by the data generated by past actions. Thus, an agent’s behavior influences both its future observations, and the representational

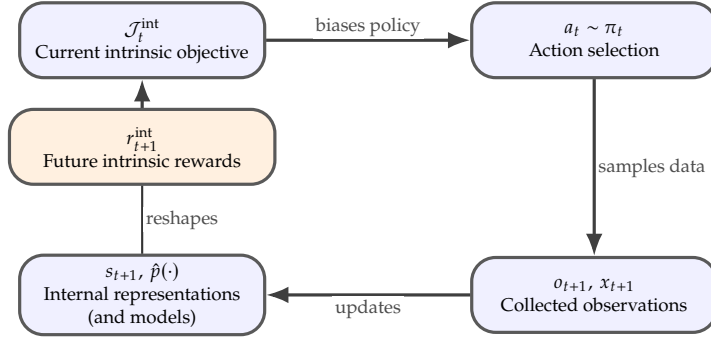


Figure 6.3: The perception-action-representation loop underlying intrinsic motivation. Actions $a_t \sim \pi_t$, biased by the current objective $\mathcal{J}_t^{\text{int}}$, produce observations (o_{t+1}, x_{t+1}) that update internal representations s_{t+1} and predictive model $\hat{p}(\cdot)$; these in turn determine intrinsic rewards r_{t+1}^{int} , which feed back into the next objective and close the loop.

basis on which those observations are understood (Ha and Schmidhuber, 2018b).

In the context of intrinsic motivation, this circular dependence acquires an additional layer. An agent’s current intrinsic objective influences which actions it selects; those actions determine which observations are collected; and those observations shape the internal representations and predictive models from which future intrinsic rewards are computed. We refer to this here as the *perception-action-representation loop*, to emphasize the representational layer that mediates between action and subsequent motivation. The central claim is that intrinsic motivation is not only a driver of exploration, but can also indirectly reshape the very state space over which future exploration incentives are defined.

This perspective helps clarify an important distinction that is often left implicit in the intrinsic-motivation literature: whether the agent’s state representation is fixed or learned. Much prior work assumes that the relevant state space is given—motivation influences actions and actions influence observations, but the representational substrate itself does not change with experience. The intrinsic reward landscape may therefore evolve through counts, beliefs, or model uncertainty, while the underlying state description remains fixed. When representations are learned, by contrast, this dependence becomes stronger. The exploration policy shapes the distribution of visited data, the visited data shapes the learned representation and world model, and the learned representation in turn affects how future states are predicted, distinguished, and rewarded (Ha and Schmidhuber, 2018b; Hafner et al., 2023; Houthoofd et al., 2016).

In practice, the distinction is not absolute. Many agents combine partially fixed perceptual features with learned latent dynamics or reward-relevant embeddings. Still, the contrast is conceptually useful: with fixed representations, intrinsic motivation operates on a stable substrate; with learned representations, it operates on a substrate that is itself shaped by experience.

6.3 Methods

We investigate the interaction between intrinsic motivation and state representation using model-based reinforcement learning (RL) agents in grid-world environments. The experimental design isolates the effect of interest through a two-stage comparison: (i) a Tabular agent with fixed, hand-crafted state representations allows the intrinsic motivation signals themselves to be studied in isolation; (ii) a Dreamer agent that learns latent representations from observations reveals how motivation and representation co-evolve when the feedback loop between them is closed.

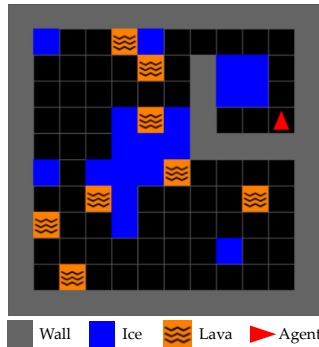


Figure 6.4: Mixed-state playground. The mixed-state playground contains lava (fatal), wall (impassable), and ice (stochastic transitions) cells. The agent starts in the upper right corner and must navigate a bottleneck to access a larger exploration area.

6.3.1 Environment

Our environments are built on the MiniGrid framework (Chevalier-Boisvert et al., 2023), a two-dimensional grid-world platform that supports partially observable interactions with discrete actions. To study the interaction between curiosity and competence, we require an environment in which risk, uncertainty, and controllability can be independently manipulated. We therefore design a *mixed-state playground* (Figure 6.4) containing three cell types:

- ▶ **Wall:** walls block movement and constrain the agent’s reachable paths. Walls impose deterministic structural constraints.
- ▶ **Ice:** transitions on ice are stochastic—the agent slips to a randomly selected neighbouring cell. Ice represents irreducible aleatoric uncertainty, where repeated experience cannot improve prediction.
- ▶ **Lava:** entering a lava cell terminates the episode with an irreversible penalty. Lava represents fatal, unrecoverable risk.

The spatial layout creates informative contrasts. The agent starts in an upper corner room and must navigate through a bottleneck of walls and lava to reach a larger exploration area. This larger area can be accessed either via a shortcut through a stochastic ice patch or via a longer but fully deterministic route. The design produces distinguishable exploration signatures: curiosity-driven agents should be drawn to the ice (high uncertainty), empowerment-driven agents should avoid it (low controllability), and a balanced agent should explore deterministic areas before cautiously approaching stochastic regions.

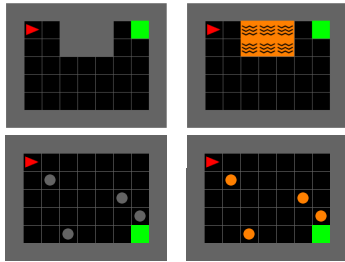


Figure 6.5: Unary-state environments. Four variants cross two axes: *harmful* (fixed lava or orange bouncing balls) vs. *harmless* (grey bouncing balls or open floor), and *stochastic* (randomly moving balls) vs. *deterministic* (no moving obstacles). Each combination isolates a distinct pairing of danger and unpredictability for the generalization experiments.

To isolate the contribution of individual environmental features, we additionally construct a set of *unary-state grid worlds* (Figure 6.5) that decompose the mixed environment along two orthogonal axes: *harmful* versus *harmless* (presence or absence of lava and hazardous moving obstacles) and *deterministic* versus *stochastic* (presence or absence of stochastic transitions). This 2×2 design yields four environment variants, each testing a different combination of risk and unpredictability, used in the generalization experiments of Section 6.5.

6.3.2 Agent-environment interaction

Interactions are formalized as a Partially Observable Markov Decision Process (POMDP) (Section 3.2). At each time step, the agent receives a partial observation $o \in \mathcal{O}$ of its immediate surroundings which is a local egocentric view of the grid. Based on a state representation s derived from the current observation, the agent selects an action $a \sim \pi(a|s)$ from a discrete action space $\mathcal{A} = \{\text{forward}, \text{turn_left}, \text{turn_right}\}$. The environment transitions to a new state s' according to the transition dynamics $p(s'|s, a)$, and the agent receives a reward. During the exploration (pretraining) phase, rewards consist entirely of intrinsic motivation signals; extrinsic rewards are introduced only during a subsequent generalization phase (see Section 6.5.3).

6.3.3 Agent architectures

A central challenge in studying intrinsic motivation is disentangling the role of the *signal itself*—what the agent is rewarded for—from the role of the *representation over which it is computed*. To address this, we use two architectures that span a spectrum from fixed to learned representations,

enabling us to study curiosity and competence in isolation (Tabular) and in co-evolution (Dreamer).

Tabular agent

The Tabular agent receives a hand-crafted state representation that reduces the full environment state to a tuple $s = (x, y, \theta) \in \mathbb{N}^3$, where x and y are grid coordinates and θ is the agent's facing direction. This representation is fixed throughout training; the agent's learning task is restricted to estimating the transition dynamics $p(s' | s, a)$ and deriving a policy over those known states.

The transition model is estimated via a Bayesian count-based approach. Let $N(s, a, s')$ denote the number of observed transitions from state s to s' under action a . The posterior predictive distribution is:

$$p(s' | s, a) = \frac{\alpha N(s, a, s') + 1}{\sum_{\tilde{s}' \in \mathcal{S}} \alpha N(s, a, \tilde{s}') + 1}, \quad (6.1)$$

where $\alpha = 100|\mathcal{S}|$ is a scaling factor that controls the rate at which the agent updates from the uninformative (uniform) prior toward the empirical counts. A key design choice is that the agent begins with a *uniform* prior over next states: before any transition is observed, every (s, a) pair appears equally likely to lead to any state. This prior has a non-obvious but important consequence for empowerment computation³, discussed in [Section 6.4.4](#).

The agent plans with a model-based *prioritized sweeping* procedure over the learned transition dynamics (Moore and Atkeson, 1993). Rather than updating all state-action values uniformly, the planner repeatedly applies Bellman backups to the states whose values are expected to change the most, and then propagates those changes backward to likely predecessor states. Concretely, it maintains a state-value function $Q(s, a)$ and updates it according to

$$Q(s, a) = \sum_{s' \in \mathcal{S}} p(s' | s, a) [(1 - \gamma) r(s, a, s') + \gamma U(s') (1 - \mathbb{1}_{\text{term}}(s'))] \quad (6.2)$$

where $\gamma = 0.5^{\sqrt{2}/|\mathcal{S}|}$ is a state-count-dependent discount factor⁴ that makes the effective planning horizon grow with environment size, and where terminal transitions contribute no future value. Prioritized sweeping then orders these backups by the magnitude of their pending change, so that large value updates are propagated efficiently through the learned model⁵. Actions are selected greedily from the current model-based action values:

$$\pi(a | s) = \arg \max_{a \in \mathcal{A}} Q(s, a). \quad (6.3)$$

The Tabular agent serves as a controlled baseline. Because its state representation is fixed and interpretable, any differences in exploration behavior across intrinsic motivation conditions can be attributed entirely to the motivation signal, not to differences in learned representations. An important additional consideration is the model-based Q-update ([Equation \(6.2\)](#)). Because intrinsic signals are inherently nonstationary, which decay as the agent explores, a model-free approach that stores and bootstraps old reward values would systematically overestimate the

3: With a uniform prior, every state appears equally reachable from every other, so all states look identically empowering before any data is collected—the agent has no basis for preferring open regions over walls or lava.

4: This sets γ so that the planning horizon roughly matches the expected number of steps to cover the state space—larger environments get longer horizons automatically.

5: This differs from the more standard updating equation

$$Q(s, a) = \sum_{s' \in \mathcal{S}} p(s' | s, a) [r(s, a, s') + \gamma \max_{a'} Q(s', a')].$$

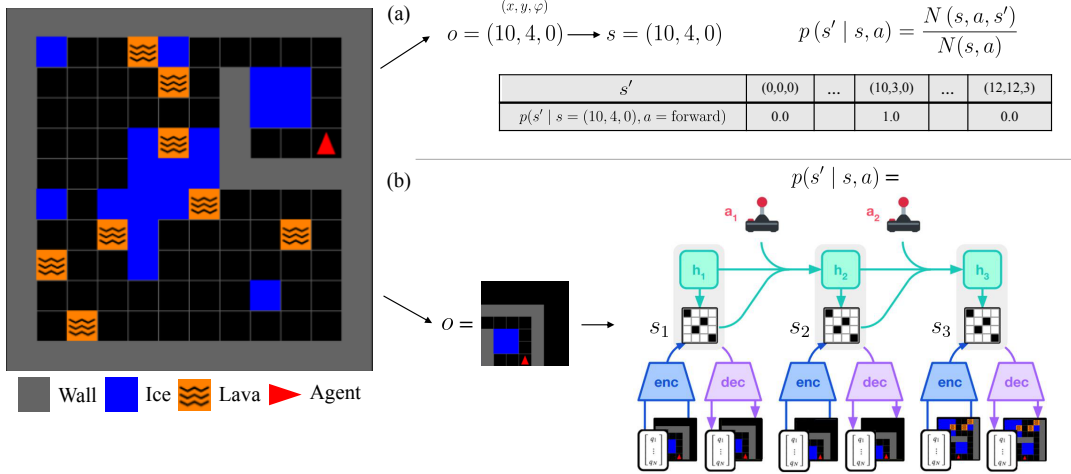


Figure 6.6: Two architectures, two feedback loops. (a) The *Tabular agent* maps the environment state directly to a fixed coordinate tuple $s = (x, y, \varphi)$ and maintains an explicit count-based transition model. The representation and dynamics model are interpretable and do not change with experience. (b) The *Dreamer agent* maps raw pixel observations to a compact latent state through a Recurrent State-Space Model (RSSM), maintaining a recurrent context h across time. The encoder, dynamics predictor, and decoder are learned jointly, so the latent state space and the intrinsic reward landscape evolve as the agent explores. The contrast between the two architectures is the experimental lever of this chapter: the tabular setting isolates each signal’s behavior; the Dreamer setting reveals what changes when motivation and representation co-evolve.

remaining intrinsic value of already-explored states; see [Section C.3](#) for details.

Dreamer agent

The Dreamer agent employs the DreamerV3 architecture (Hafner et al., 2023) ([Figure C.1](#)), a model-based RL system that learns a compact world model from raw pixel observations without task-specific supervision. Unlike the Tabular agent, the Dreamer agent has no fixed state abstraction: it simultaneously learns *what to represent* and *how the world works* in that representation. This co-learning is the defining property that makes [Section 6.5](#) theoretically informative: intrinsic reward signals computed over learned latent states are themselves functions of a changing representational substrate.

World model architecture. The world model is a Recurrent State-Space Model (RSSM) with three core components. The *sequence model* maintains a deterministic recurrent state h that serves as the agent’s working memory⁶, capturing temporal dependencies and resolving ambiguities arising from partial observability:

$$h' = f_{\phi}(h, s, a). \quad (6.4)$$

From this recurrent context, two complementary predictors operate in parallel. The *encoder* (posterior) computes a discrete latent state by incorporating the incoming observation:

$$s \sim p_{\phi}(s), \quad (6.5)$$

while the *dynamics predictor* (prior) generates a prediction of the same latent state *without* observational access:

$$\hat{s} \sim p_{\phi}(\hat{s} | h). \quad (6.6)$$

6: The RSSM is a continuous recurrent state, unlike an LSTM or GRU in that the deterministic path h and the stochastic path s are kept explicitly separate. This separation is what allows the dynamics predictor to generate predictions without observation access, which is analogous to a human predicting what happens next before looking.

During training, the KL divergence between encoder posterior and dynamics predictor prior is minimized, encouraging the world model to predict future latent states from past experience alone. The residual divergence⁷—the gap that remains after training—serves as the natural measure of model surprise and is the basis for the information gain reward (Section 6.3.4). Full architecture details, including the reward predictor, continue predictor, and decoder components, are given in Section C.1. Figure 6.6 contrasts the information flows of both architectures side by side.

Factorized discrete latent space. Dreamer represents each latent state s as a *factorized* collection of L independent categorical variables:

$$s = [s_1, s_2, \dots, s_L], \quad s_i \sim \text{Cat}(K) \text{ independently}, \quad (6.7)$$

where each factor s_i takes one of K discrete values⁸. This factorization has two consequences for reward computation. First, naive state counting over the full joint state s leads to combinatorial explosion. The K^L possible states are nearly all unvisited even after extended training, rendering every state perpetually novel and every count-based metric uninformative. Second, the factorized structure provides a principled resolution: compute each intrinsic signal separately per factor s_i and average across factors. We adopt this per-factor averaging uniformly for all three intrinsic signals, ensuring that novelty, information gain, and empowerment are computed over the same representational units and are therefore directly comparable.

7: The KL can be decomposed as $D_{\text{KL}}[p_\phi(s) | p_\phi(\tilde{s} | h)]$. DreamerV3 trains with a “free bits” threshold (Hafner et al., 2023): the KL is only penalized above a minimum value, preventing posterior collapse while still encouraging predictability.

8: In our implementation, $L = 32$ factors of $K = 32$ categories each

6.3.4 Intrinsic motivation signals

We now define the three intrinsic motivation signals studied in this chapter. Let $s \in \mathcal{S}$ denote the agent’s state representation, $a \in \mathcal{A}$ for the action. We write $p(s' | s, a)$ for the agent’s model of the transition dynamics and $N(s)$ for the visitation count of state s . We implement three intrinsic motivation signals, each computed over the agent’s internal state representation s rather than the true environment observations $o \in \mathcal{O}$.

Novelty

Novelty rewards the agent for visiting infrequently encountered states, operationalizing Berlyne’s collative property of the same name as the negative log-frequency of the current state.

Tabular.

$$r_{\text{nov}}^{\text{tab}}(s, a, s') = -\log \frac{N(s)}{\sum_{\tilde{s} \in \mathcal{S}} N(\tilde{s})}, \quad (6.8)$$

where $N(s)$ is the visitation count for state representation s . Novelty is agnostic about dynamics. It depends only on how often a state has been visited, not on what happens when the agent acts there.

Dreamer. Because Dreamer uses a factorized discrete latent space (see Equation (6.7) and the accompanying discussion), we track counts per factor and average:

$$r_{\text{nov}}^{\text{dream}}(s, a, s') = -\frac{1}{L} \sum_{i=1}^L \log \frac{N(s_i)}{\sum_{\tilde{s}_i} N(\tilde{s}_i)}, \quad (6.9)$$

Remark 6.1 (Novelty in learned vs. fixed feature spaces) Prior work scales count-based novelty to high-dimensional observations by projecting into a *fixed* random feature space (Tang et al., 2017) or by using the prediction error of a network trained to match a fixed random encoder (RND; Burda et al., 2018)). Because the feature space is static, visitation counts remain meaningful throughout training.

Our per-factor counts differ in two respects: the factors s_i are semantically structured by world-model training rather than randomly initialized, and they are *non-stationary*—as the model updates, the meaning of each s_i shifts, making counts computed at early training unreliable at later steps. This representational drift and its effect on novelty estimates is discussed in Section 6.5.4.

where $N(s_i)$ denotes how often factor i has taken the value s_i . This per-factor novelty is conceptually identical to the Tabular formula but applied to the learned latent vocabulary. As we will see in Section 6.5.4, this implementation interacts with an important pathology: because the latent representation itself changes during training, factor counts computed at one time step may not accurately reflect novelty at a later step when the meaning of each s_i has shifted.

Information gain

Information gain rewards the agent for performing actions that reduce uncertainty about the environment’s transition dynamics, operationalizing Loewenstein’s information-gap theory in information-theoretic terms. A key distinction separates the two agent architectures: the Tabular agent computes *predicted* (prospective) information gain—the expected reduction in uncertainty *before* observing an outcome—while the Dreamer agent uses a *retrospective* approximation computed *after* the transition is observed. This asymmetry has substantive consequences for exploration efficiency, discussed in Section C.2.

Tabular. For the Tabular agent, the transition model is a categorical distribution updated by Bayesian counts, which admits analytic computation of the predicted information gain:

$$r_{\text{IG}}^{\text{tab}}(s, a) = \mathbb{E}_{\hat{s}' \sim p(S' | s, a)} [D_{\text{KL}}(p(S' | s, a, \hat{s}') \parallel p(S' | s, a))], \quad (6.10)$$

where $p(S' | s, a, \hat{s}')$ denotes the updated posterior after observing the specific transition to $\hat{s}'$, and the expectation is taken over the current predictive distribution. Intuitively, information gain is high when the transition model is uncertain and a single observation would substantially sharpen it; it is zero when the model is already confident, and it eventually approaches zero even on stochastic ice cells once the aleatoric variance has been learned.

Dreamer. Computing the predicted information gain in a neural world model would require an intractable number of forward passes through the dynamics predictor. We therefore use a retrospective approximation: after observing a transition, information gain is measured as the divergence between the observation-informed posterior and the prior prediction of the dynamics model⁹:

$$r_{\text{IG}}^{\text{dream}}(s, a, s') = \frac{1}{L} \sum_{i=1}^L D_{\text{KL}}(p_{\phi}(s'_i | h', o') \parallel p_{\phi}(s'_i | h')). \quad (6.11)$$

This quantity is high when the world model is surprised by the observation—that is, when the encoder’s posterior diverges from the dynamics predictor’s prior. The per-factor averaging parallels the factorized novelty formulation, ensuring comparable magnitude across signals.

Empowerment

Empowerment measures the degree to which an agent’s actions influence subsequent states—the channel capacity of the agent’s actuation channel (Klyubin et al., 2005). Formally, empowerment at state s is:

$$\text{EMP}(s) = \max_{\omega(a | s)} \text{MI}(S'; A | s), \quad (6.12)$$

9: This retrospective KL formulation is related to two other traditions. Pathak et al. (2017)’s ICM computes the analogous divergence in a learned *feature* space, using prediction error as a proxy for surprise. The active-inference literature interprets the same quantity as the information-gain component of *expected free energy*: curiosity corresponds to seeking observations that reduce uncertainty, decomposable into information gain and expected preference (Schwartenbeck et al., 2013).

where $\text{MI}(S'; A | s)$ denotes the mutual information between the action A and the resulting next state S' , maximized over the agent's action distribution ω . Empowerment is high when the agent can reliably steer the environment into many distinguishable future states, and structurally penalized by stochasticity.

Tabular. In the Tabular agent, we compute one-step empowerment exactly using the Blahut-Arimoto algorithm (Dupuis et al., 2004), which iteratively optimizes the action distribution ω to maximize the mutual information:

$$r_{\text{emp}}^{\text{tab}}(s, a, s') = \text{EMP}(s) = \max_{\omega(a|s)} \text{MI}(S'; A | s). \quad (6.13)$$

Note that empowerment depends only on the current state, not on the specific action or outcome: it reflects the agent's *potential* for control at s , regardless of which action was actually taken.

Dreamer. Applying Blahut-Arimoto to the learned latent dynamics predictor is intractable in the general case. We approximate empowerment by (i) assuming a uniform source policy $\omega = \mathcal{U}(\mathcal{A})$ and (ii) computing the mutual information separately per latent factor (Seitzer et al., 2021):

$$r_{\text{emp}}^{\text{dream}}(s, a, s') = \frac{1}{L} \sum_{i=1}^L \text{MI}_{\mathcal{U}}(S'_i; A | s) = \frac{1}{L} \sum_{i=1}^L [\text{H}(S'_i | s) - \text{H}(S'_i | A, s)], \quad (6.14)$$

where $\text{H}(S'_i | s) = \text{H}\left[\frac{1}{|\mathcal{A}|} \sum_a p_{\phi}(s'_i | h', a)\right]$ is the marginal entropy of factor i under a random action policy, and $\text{H}(S'_i | A, s) = \frac{1}{|\mathcal{A}|} \sum_a \text{H}[p_{\phi}(s'_i | h', a)]$ is the action-conditioned entropy. Their difference is high when different actions lead to distinguishable next-latent distributions for factor i , and low when all actions produce similar or unpredictable transitions. Limitations of this approximation—including the single-step horizon, the uniform-policy lower bound, and its interaction with the learned representation—are discussed in [Section C.4](#).

Comparison

The three signals differ along three key axes ([Figure 6.7](#)):

What is rewarded: novelty rewards visiting infrequent states; information gain rewards transitions that update the world model; empowerment rewards occupying states with high causal influence over future states.

Sensitivity to stochasticity: novelty is blind to stochasticity and will reward re-visiting it indefinitely if the representation changes; information gain eventually extinguishes in stochastic regions once it has learned the aleatoric variance; empowerment is penalized by stochasticity as it reduces action-state mutual information.

Model dependence: novelty requires only a visitation counter; information gain requires an uncertainty-aware model of transition dynamics; empowerment requires a model of transition dynamics plus an optimization over multi-step action sequences.

These differences motivate distinct behavioral predictions. In our mixed-state grid world ([Figure 6.7](#)), novelty predicts equal interest in lava, ice, and open cells (all may be equally unvisited); information gain predicts high interest in unexplored ice cells (uncertain dynamics) but eventual abandonment once the stochasticity is learned; empowerment predicts avoidance of ice and lava, and preference for open deterministic regions where the agent's actions have reliable consequences.

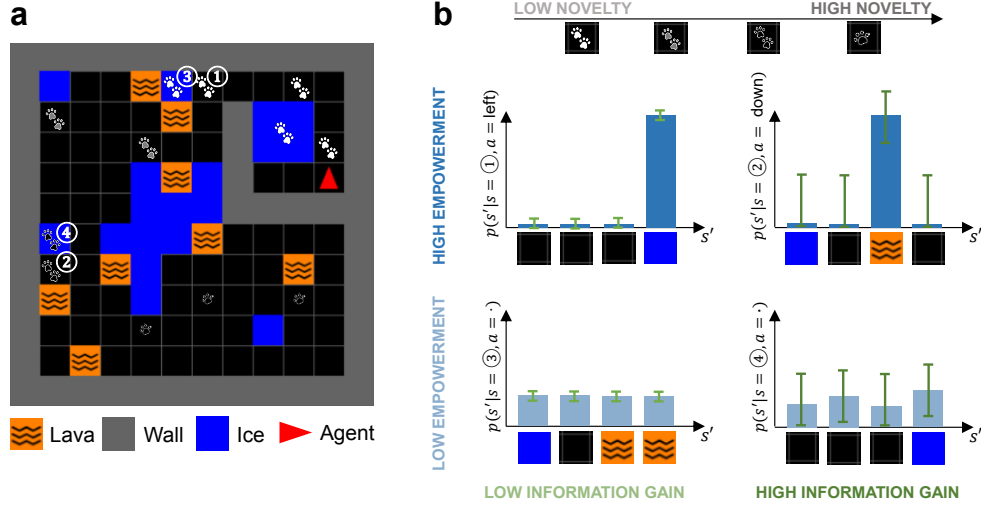


Figure 6.7: Task overview. (a) The mixed-state playground has three different cells: lava, wall, and ice. The agent (red triangle) starts from the upper right corner and has to explore the environment while avoiding death (falling into lava). (b) Novelty depends only on the visitation frequency for each state, regardless of the state types. Information gain is low for transitions which the agent has already learned to predict well (cells ① and ③), and high for transitions that the agent is less confident about (cells ② and ④). Empowerment is low in slippery ice cells where the agent cannot control where it will end up (cells ③ and ④), and high in states with many predictable action outcomes (cells ① and ②).

Hybrid motivations

To test whether curiosity and competence are complementary, we evaluate two simple hybrid reward signals that combine information gain and empowerment:

$$r_{\text{sum}}(s, a, s') = r_{\text{IG}}(s, a, s') + r_{\text{emp}}(s, a, s'), \quad (6.15)$$

$$r_{\text{prod}}(s, a, s') = r_{\text{IG}}(s, a, s') \cdot r_{\text{emp}}(s, a, s'). \quad (6.16)$$

We choose to combine information gain (rather than novelty) with empowerment for two reasons. First, from a theoretical perspective, information gain specifically targets improvement of the dynamics model—the same model used to compute empowerment—establishing a direct computational coupling between the two. Second, from an empirical perspective, initial experiments confirmed that information gain produces more thorough exploration than novelty in both architectures. All intrinsic reward signals are normalized to the $[0, 1]$ range using global min-max scaling before combination, ensuring the sum and product hybrids are not dominated by signals with larger magnitudes.

6.4 Experiment 1: Tabular Agent

We begin with the Tabular agent, where fixed state representations isolate the effect of each intrinsic motivation signal. Because the state vocabulary is fixed and transparent, any behavioral differences across conditions must be attributed entirely to the signal itself. This controlled setting allows us to answer a foundational question: what does each drive *ask for*, and what are the consequences?

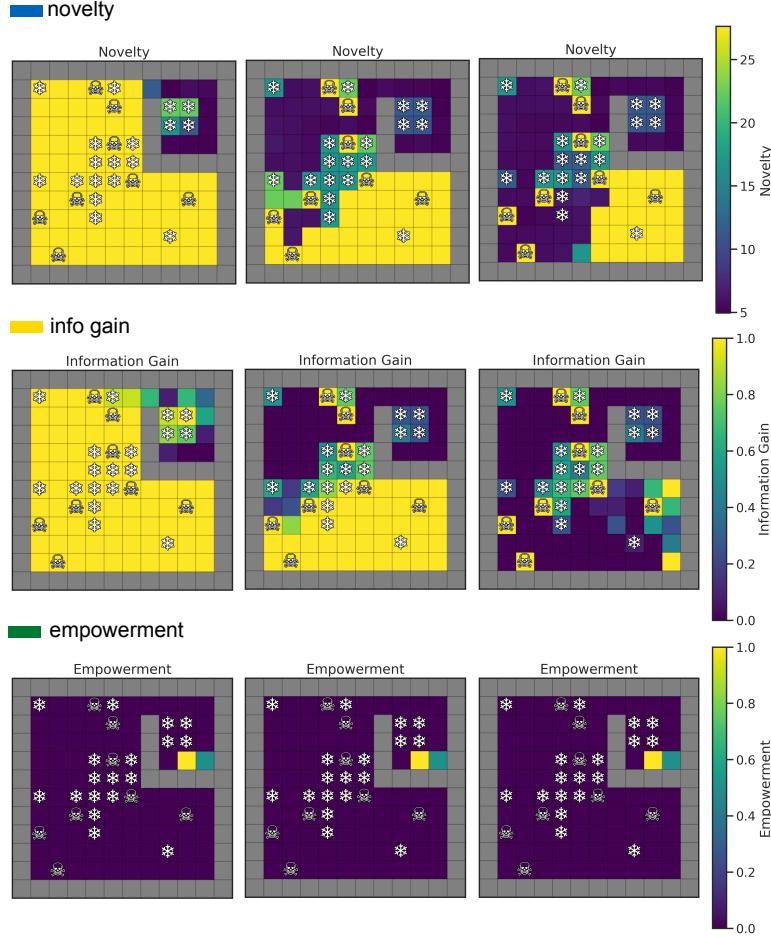


Figure 6.8: Intrinsic reward heatmaps in the mixed-state playground at 100, 1,000, and 2,000 training steps. Novelty remains broadly spread over unvisited states, reflecting its purely count-based nature. Information gain contracts toward uncertain regions but stays elevated on the stochastic ice patch, illustrating its attraction to irreducible uncertainty. Empowerment remains sharply localized near the controllable bottleneck, assigning little value to most unseen states. The three signals therefore differ both in where they focus attention and in how their reward landscapes evolve over learning.

6.4.1 Experimental setup

Each Tabular agent was evaluated over 10,000 steps and 10 independent random seeds, under one of five intrinsic reward conditions: novelty (Equation (6.8)), information gain (Equation (6.10)), empowerment (Equation (6.13)), sum hybrid (Equation (6.15)), and product hybrid (Equation (6.16)). A random-walk agent without any intrinsic reward serves as the baseline.

6.4.2 Motivation heatmaps: the geometry of attention

To visualize how each motivation signal allocates attention across the environment over time, we examine reward-landscape snapshots at three representative stages of training: 100, 1,000, and 2,000 steps (Figure 6.8).

The heatmaps confirm the theoretical predictions from Section 6.2. Novelty remains broadly distributed over unvisited cells, regardless of transition structure, because its value depends only on visitation counts. Information gain contracts toward uncertain regions, but stays high on the stochastic ice patch even after repeated visits, since its transitions remain inherently unpredictable. Empowerment shows the opposite pattern: it is sharply localized near the controllable bottleneck and changes little over training, assigning near-zero value to most unseen states.

The information gain heatmap reveals the precise mechanism of the *noisy TV problem*. Examining the state visitation counts alongside the reward

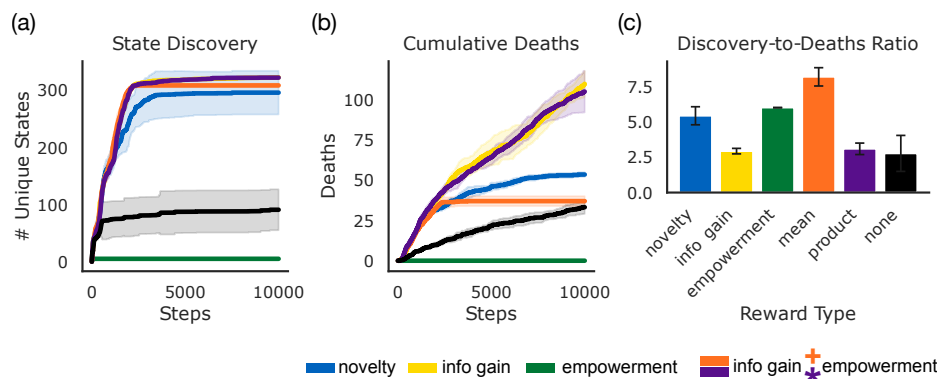


Figure 6.10: Exploration patterns of the Tabular agent on the mixed-state playground. (a) Total unique states discovered over time. Information gain achieves the broadest coverage; empowerment discovers almost nothing beyond the starting room. (b) Cumulative deaths. Information gain incurs the most deaths; empowerment incurs none. (c) Discovery-to-death ratio at step 10,000. The sum combination of information gain and empowerment achieves the best trade-off, exploring broadly while limiting deaths.

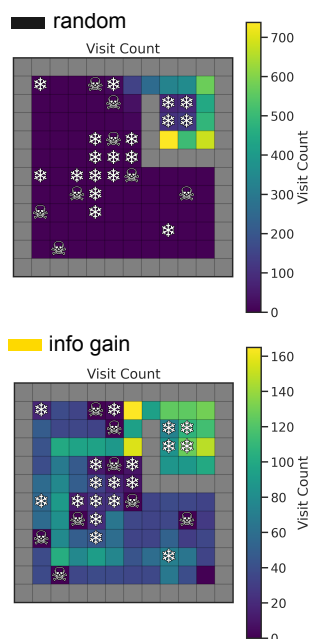


Figure 6.9: Visitation heatmaps for the mixed-state playground. The panels compare a random policy and the information-gain agent, showing the number of visits to each grid cell over training. The information-gain agent exhibits substantially broader spatial coverage, traversing the bottleneck and repeatedly visiting distant regions of the environment, whereas the random policy remains more uneven and locally concentrated.

landscape shows that the pure information-gain agent visits ice cells at coordinate (8,9) substantially more often than neighboring floor cells (Figure 6.9). This is not random: the Bayesian count-based model correctly identifies that stochastic transitions remain high-entropy even after many observations, and the information gain reward correctly computes this as non-zero uncertainty. The flaw is not in the computation but in the signal's inability to distinguish between uncertainty that can be reduced (epistemic, about deterministic but unexplored cells) and uncertainty that cannot (aleatoric, about inherently stochastic transitions).

6.4.3 Exploration coverage and safety

Beyond qualitative heatmaps, we also summarize the performance using *unique state coverage* (number of distinct states visited) and *cumulative deaths* (episodes terminated by entering a lava cell).

We report the discovery-to-death ratio as a composite measure of safe, efficient exploration. The results reveal a clear three-way dissociation among the motivation signals. The information-gain agent achieves the highest state coverage (Figure 6.10a), systematically visiting all accessible regions of the grid. This thoroughness comes at a cost: the information-gain agent also incurs the most deaths (Figure 6.10b), because it repeatedly enters ice cells adjacent to lava in an attempt to reduce its uncertainty about stochastic transitions that are, by design, irreducible. Unlike a random policy, the information-gain agent spreads its visits across nearly the entire accessible map (until half time of the training, Figure 6.9), including distant rooms beyond the bottleneck. We examine the mechanism of this fixation in Section 6.4.2.

The empowerment agent exhibits the opposite profile. It remains in the starting room, never traversing the bottleneck, and incurs zero deaths throughout. From the perspective of exploration, this agent's strategy is maximally conservative: it only occupies states in which it has high control, ignoring the rest of the environment entirely. Why it fails so completely requires a closer look at the interaction between its reward signal and its initial world model.

The novelty agent falls between these extremes, achieving moderate coverage with moderate risk. Its exploration is driven purely by visitation counts and is therefore insensitive to the *structure* of transitions—it neither seeks out informative dynamics nor avoids uncontrollable ones.

6.4.4 Why empowerment fails: the uniform prior problem

The complete failure of the pure empowerment agent is not immediately obvious from the reward definition. Empowerment rewards states where the agent's actions reliably produce distinguishable outcomes—a sensible objective for an agent that wishes to maintain control. Yet this agent never even attempts to explore. The explanation lies in the interaction between the empowerment signal and the initial state of the transition model.

Recall that before any transition has been observed, the count-based model (Equation (6.1)) assigns a uniform distribution over all possible next states for every state-action pair. A uniform distribution has *maximum entropy*: no action appears to lead to a distinguishable outcome. Consequently, the empowerment of every unvisited state is computed as *zero*—the agent believes it has no control anywhere outside the starting room. This creates a self-reinforcing trap (Figure 6.11). The agent computes empowerment using its current transition model. The model assigns zero empowerment to unseen states. Therefore the agent never visits unseen states. Therefore the model is never updated. The agent that most needs knowledge about the world to be effective is precisely the one that has no mechanism to acquire it¹⁰.

This computational insight has a direct psychological parallel: empowerment is not a primitive drive for exploration, but a control objective that depends on prior knowledge. An agent can seek control only over parts of the world it already models well enough to predict (Gopnik, 2024). A child cannot use an object strategically before learning its affordances; likewise, an agent cannot recognize a region as empowering before learning its dynamics. Empowerment therefore presupposes a minimal causal model of the environment. Without a complementary drive to acquire that model, empowerment alone cannot get started.

6.4.5 The complementarity of curiosity and competence

The most informative result concerns the hybrid conditions. Combining information gain and empowerment via a simple sum yields the highest discovery-to-death ratio (Figure 6.10c). This agent explores most of the environment—including regions the pure empowerment agent never reaches—while avoiding the repeated deaths that plague the pure information-gain agent.

Mechanism. The information gain reward is highest at the *beginning* of training, when the transition model is uncertain almost everywhere. As the agent explores and updates its model, information gain decays—rapidly in deterministic regions (where a few observations suffice to predict transitions confidently) and more slowly in stochastic regions. Empowerment, by contrast, converges to a stable *positive* value in deterministic floor cells: once the agent has learned that its actions lead to reliable outcomes there, these cells provide a constant reward for being in them. The combined agent therefore undergoes a natural motivational transition. Early in training, information gain dominates, driving the agent to explore unexplored regions. As the model matures, empowerment increasingly shapes the agent's attention: states where dynamics are known and controllable receive the highest combined reward, while ice cells—high information gain but zero empowerment—are penalized by the empowerment component. The result is information-gain-seeking *constrained* to empowering states.

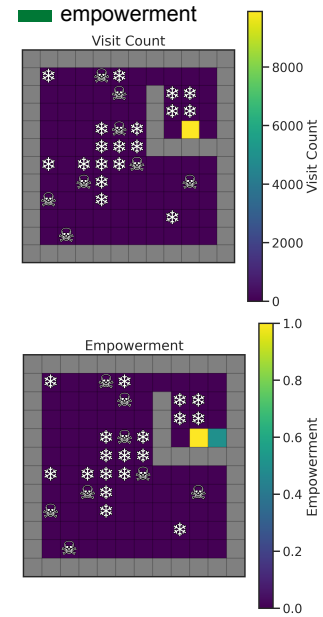


Figure 6.11: The uniform prior traps the empowerment agent. State visitation heatmap at the end of training: the empowerment agent never leaves the starting room. Empowerment estimates across the grid: because the transition model was never updated for unseen states, empowerment remains near zero everywhere outside the starting cell, giving the agent no gradient to explore.

10: This failure is therefore not a defect of empowerment per se, but of using empowerment alone before the agent has acquired a minimally informative model of the dynamics.

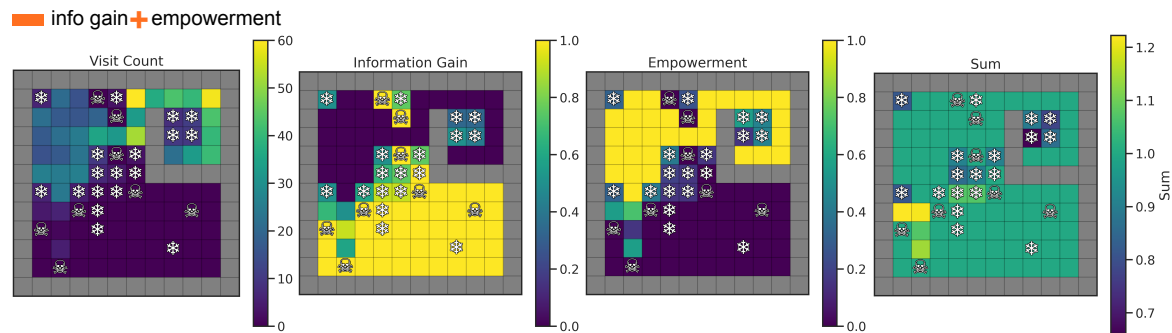


Figure 6.12: Mechanism of complementarity for the additive hybrid agent in the mixed-state playground. At training step 1,000, the visitation heatmap (left) shows broad exploration without the excessive concentration on hazardous stochastic states seen under pure information gain. The component heatmaps (middle) show that information gain favors uncertain states, whereas empowerment favors controllable ones. Their sum (right) therefore peaks in states that are both informative and controllable, suppressing attraction to stochastic but unempowering regions. This interaction explains the hybrid agent’s improved exploration-safety trade-off.

This mechanism is illustrated in Figure 6.12. The combined heatmap peaks in regions that are both controllable and not yet fully modelled—the frontier where representational investment pays the highest return. In states where information gain is high but empowerment is low (stochastic, uncontrollable), the empowerment component dampens the reward, naturally discouraging further visits once the agent recognises the lack of control.

Product vs. sum. The product combination ($r_{\text{prod}} = r_{\text{IG}} \cdot r_{\text{emp}}$) behaves differently and less favourably. Multiplying the two signals enforces a strict conjunction: the product is high only when *both* information gain and empowerment are simultaneously high, and goes to zero if either is zero. In practice, empowerment is zero in all unvisited states (the uniform prior problem), so the product reward cannot motivate initial exploration. The agent effectively reduces to a pure information-gain strategy until empowerment estimates become non-zero, losing the synergistic benefit. The sum, by contrast, allows either component to be informative and thus provides a gradient for exploration from the outset. We retain both combinations in our analyses for completeness, but the sum hybrid is the focus of interpretation.

6.5 Experiment 2: Dreamer Agent

We now turn to the setting where state representations are learned during exploration. Unlike the Tabular agent, the Dreamer agent is not given a hand-specified state space for the grid world. Instead, it must simultaneously discover *what to represent*—which features of pixel observations are informationally relevant—and *how the world works* in that learned representation. This closes the feedback loop that the Tabular setting deliberately holds open: the motivation signal shapes exploration, exploration shapes the training data, training data reshapes the world model, and the reshaped world model in turn changes the intrinsic reward landscape. The goal of this experiment is to make this feedback loop visible and to understand its consequences for each motivation signal.

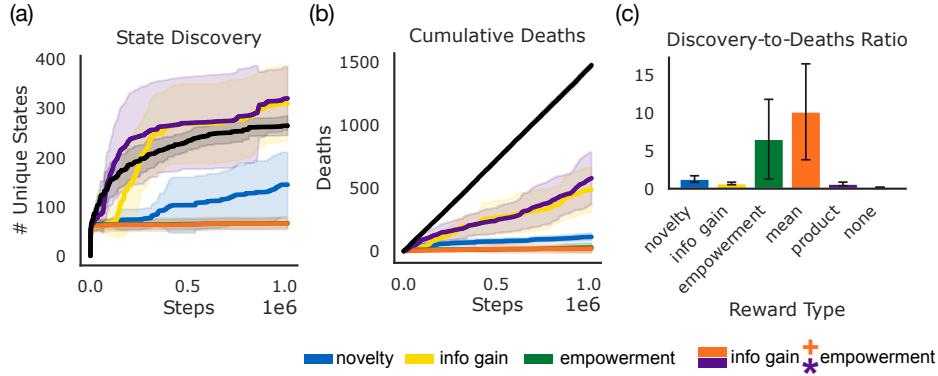


Figure 6.13: Exploration patterns of the Dreamer agent on the mixed-state playground. (a) Unique state discovery over time. The empowerment agent explores the starting room but cannot traverse the bottleneck, while information gain achieves the broadest coverage. (b) Cumulative deaths. The qualitative ordering mirrors the tabular setting. (c) Discovery-to-death ratio. The information gain-empowerment sum combination again achieves the best trade-off, though the complementarity effect is less pronounced than in Figure 6.10.

6.5.1 Experimental setup

Each Dreamer agent is trained for 10^6 steps from pixel observations on the mixed-state playground (Figure 6.4), with intrinsic rewards computed over the learned discrete latent states, namely the categorical latent representation inferred by the world model from pixel observations. We evaluate the same five intrinsic motivation (novelty, information gain, empowerment, and the two hybrids) conditions as in Section 6.4, plus a random baseline, and assess the same primary metrics: unique state coverage, cumulative deaths, and discovery-to-death ratio. A two-phase protocol is additionally used for the unary-state environments (Section 6.5.3) to test transfer beyond the original training layout: a pretraining phase with intrinsic rewards only, followed by a generalization phase with extrinsic rewards only in structurally similar but spatially novel environments¹¹.

¹¹: environments with the same local rules and object types, e.g., lavas, but different layouts.

6.5.2 Replication on the mixed-state playground

In the Dreamer setting, the qualitative ordering of intrinsic motivations on the mixed-state playground largely mirrors the Tabular case Section 6.4, but the differences between conditions become less cleanly separated. Information gain and the information gain-empowerment hybrids still achieve the broadest state coverage, whereas pure empowerment remains the most conservative strategy. However, unlike the Tabular agent, the Dreamer agent operates over a learned latent state space rather than a fixed ground-truth representation, so the intrinsic rewards are themselves shaped by a nonstationary world model. As a result, exploration trajectories are noisier, slower to stabilise, and less sharply differentiated across reward conditions.

Empowerment with learned representations. The most notable continuity with the Tabular setting is that pure empowerment still fails to support broad exploration. In both agents, empowerment does not by itself drive successful traversal of the bottleneck into the wider environment. But the nature of this failure differs. In the Tabular case, empowerment collapses almost completely because unseen states inherit zero value under the initially uninformative transition model, leaving the agent trapped near the start (Section 6.4.4). In the Dreamer case, the world model can generalise across visually similar observations, so the agent

learns that floor-like states within the starting room share predictable and controllable dynamics (Figure 6.13a). This allows the empowerment-driven Dreamer to explore the starting room more extensively than its Tabular counterpart. Even so, it still fails to cross the bottleneck reliably, because the model receives too little evidence about what lies beyond it. Thus, the Dreamer agent softens the Tabular failure, but does not remove its basic cause: empowerment alone depends on prior model knowledge and therefore struggles to bootstrap exploration into unseen territory.

Information gain: a moving target. Information gain also remains the strongest driver of broad exploration, but again with an important difference from the Tabular setting. In the Tabular case, the reward is computed from a fixed symbolic state space, so high information gain directly reflects uncertainty about known state-action transitions. In Dreamer, by contrast, the uncertainty estimate is tied to a learned latent representation that changes throughout training. This makes information gain a moving target. States may continue to generate intrinsic reward not only because they are objectively underexplored, but because the latent model itself is still reorganising how observations are represented and predicted. This helps explain why the Dreamer information-gain agent expands coverage more slowly and with greater variance than the Tabular agent, despite eventually reaching similarly broad regions of the environment.

Hybrid strategies. The Dreamer results still indicate a complementary interaction between information gain and empowerment, most clearly in the additive hybrid, which achieves the highest mean discovery-to-death ratio among the tested rewards. However, the complementarity effect is less pronounced than in the Tabular setting (Figure 6.13c). This suggests that the two motivations continue to play distinct and useful roles: information gain drives outward expansion into poorly modelled regions of the state space, while empowerment counteracts indiscriminate risk-taking by favouring states that preserve local control and future options. In the mixed environment, this produces a desirable behavioural balance, allowing the agent to maintain broad coverage without incurring the extreme death count associated with pure information gain. Relative to the Tabular case, however, the hybrid advantage is much noisier across random seeds. The most appropriate interpretation is therefore not that complementarity disappears, but that it becomes less clearly expressed once both reward components depend on a learned and evolving latent world model. Variability in representation learning, uncertainty estimation, and planning quality can all perturb how strongly each component influences behaviour in a given run. Thus, the Dreamer results preserve the same qualitative trade-off as the Tabular experiments, but with greater variance and weaker visual separation between conditions. A more principled solution to this nonstationarity issue—prospective reward computation and replay buffer management—is discussed in Section C.3.

6.5.3 Generalization across environment types

Coverage and death counts measure what the agent explores; they do not reveal whether the resulting representations support new tasks. An agent could explore broadly yet learn a world model too entangled with the specific training layout to transfer. Conversely, an agent with limited coverage might learn a compact, structure-capturing representation that generalizes well precisely because it discards irrelevant features. The

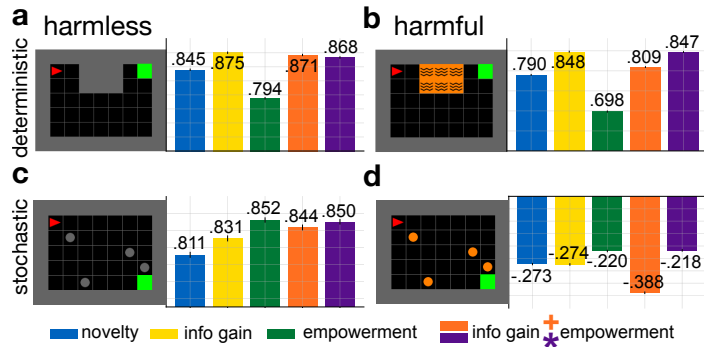


Figure 6.14: Generalization performance of the Dreamer agent across environment types. Environments are decomposed along two axes (harmful vs. harmless; deterministic vs. stochastic). Extrinsic reward is measured in novel environments after intrinsic-only pretraining. No single motivation dominates across all conditions. Novelty and information gain excel in deterministic settings; empowerment is more effective in stochastic settings. Hybrid approaches match or exceed the best single-motivation agent in each condition.

mixed-state playground cannot adjudicate between these possibilities because it conflates deterministic and stochastic dynamics, safe and hazardous cells, any performance difference between motivation conditions is attributable to any combination of these factors simultaneously. To isolate the representational consequences of each motivation, we instead evaluate agents on the four unary-state environments—pure instantiations of each combination of harmfulness and stochasticity—using a two-phase protocol in which intrinsic-reward pretraining is followed by extrinsic-reward generalization in a spatially novel layout of the same type.

Environment decomposition

We decompose the mixed-state playground along two orthogonal axes:

- **Harmless vs. harmful:** whether the environment contains lava and hazardous moving obstacles.
- **Deterministic vs. stochastic:** whether the environment contains stochastic transitions (moving walls, slippery ice patches).

This yields a 2×2 design (Figure 6.14) that isolates whether each intrinsic motivation’s advantages are specific to particular combinations of risk and unpredictability.

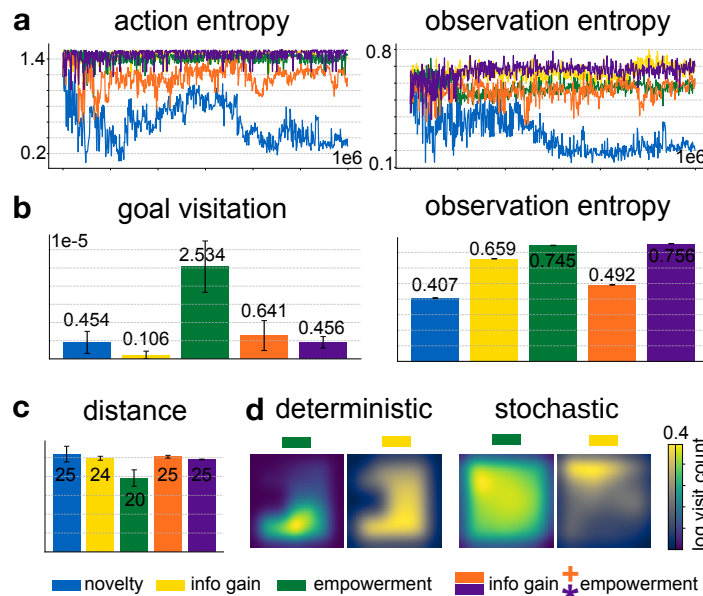
Generalization performance

The generalization results reveal a fundamental context-dependence: no single intrinsic motivation dominates across all environment types (Figure 6.14).

Deterministic environments. Novelty and information gain produce representations that support strong generalization in deterministic settings. Thorough exploration of deterministic dynamics yields a world model with broad and accurate coverage, learning how the agent’s actions propagate through the environment without the confound of irreducible stochasticity. The resulting representations capture the actionable structure of the environment efficiently.

Stochastic environments. Empowerment produces superior generalization in stochastic settings. An agent that learns to identify controllable states amid stochastic noise develops representations that foreground the *actionable* structure of the environment—precisely the structure that remains stable across layout changes. The representation learned by the empowerment agent, while covering fewer states, generalises better

Figure 6.15: Exploration patterns during pretraining. (a) Observation and action entropy over time: novelty produces the least diverse exploration, a signature of pseudo-novelty. (b) In the stochastic-harmless environment, information gain achieves high observation entropy but minimal goal visitation, reflecting fixation on irreducible uncertainty. (c) In deterministic environments, the empowerment agent maintains maximal distance from obstacles to preserve access to multiple future paths. (d) State visitation heatmaps confirm that empowerment agents prefer open regions in deterministic settings but roam continuously in stochastic settings.



because it discards uncontrollable stochasticity and focuses on reliable action-outcome relationships.

Hybrid motivations as a robust compromise. Hybrid motivations (sum and product of information gain and empowerment) achieve generalization performance equal to or better than the best single-motivation agent in each condition. This suggests that complementarity extends beyond exploration efficiency to the quality of the learned representations themselves: a hybrid exploration policy generates a data distribution that covers both learnable dynamics (via information gain) and controllable dynamics (via empowerment), producing a world model that captures environmentally stable and actionable structure.

Exploration diagnostics during pretraining

Examining exploration behavior during pretraining reveals why different motivations lead to different generalization outcomes.

Novelty: the lowest diversity. Counterintuitively, the novelty-driven Dreamer agent produces exploration with the lowest entropy of both observations and actions (Figure 6.15a). A novelty-seeking agent should visit novel states, which should produce diverse observations. The resolution of this paradox is the pseudo-novelty phenomenon we examine in depth in Section 6.5.4: representational drift in the latent space creates a subjective sense of novelty without requiring the agent to visit genuinely new locations.

Information gain: high entropy, wrong places. The information-gain agent produces high observation entropy—many diverse states visited—but achieves minimal extrinsic goal visitation in stochastic environments (Figure 6.15b). This pattern indicates that the agent fixates on inherently unpredictable transitions—moving walls, stochastic ice—rather than on the global structure of the environment. The agent is not failing to explore; it is exploring in the wrong places.

Empowerment: comfort zone and roaming. The empowerment agent reveals a striking context-dependence. In deterministic environments, it positions itself to maintain maximal access to future paths, preferring

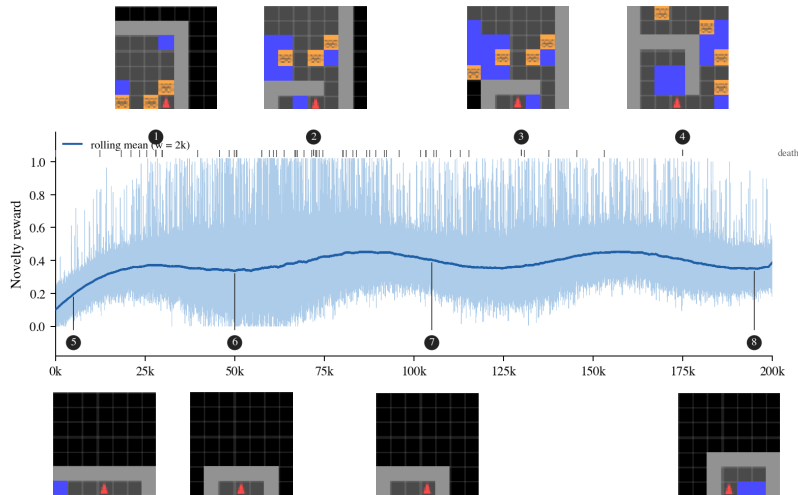


Figure 6.16: Pseudo-novelty in the Dreamer agent. Novelty rewards during the first 200,000 training steps. The dashed vertical lines indicate deaths. Despite spending most time in the starting room (annotated images), the agent’s perceived novelty remains high throughout training because representational drift in the learned latent space continuously shifts the encoding of familiar observations, registering as novel under the count-based metric. Compare with the Tabular setting, where novelty decays monotonically as states are revisited.

open areas far from walls and lava (Figure 6.15c). This conservative comfort zone behaviour limits exploration but reflects principled caution: the agent maximises its future action space. In stochastic environments, however, the empowerment agent abandons this comfort zone and roams continuously, because maintaining control in a stochastic world requires active repositioning. This context-dependent shift—from conservative to active exploration—illustrates that empowerment does not simply seek quiescence; it seeks *controllable* states, which in a stochastic world requires constant vigilance.

6.5.4 The representation-motivation feedback loop

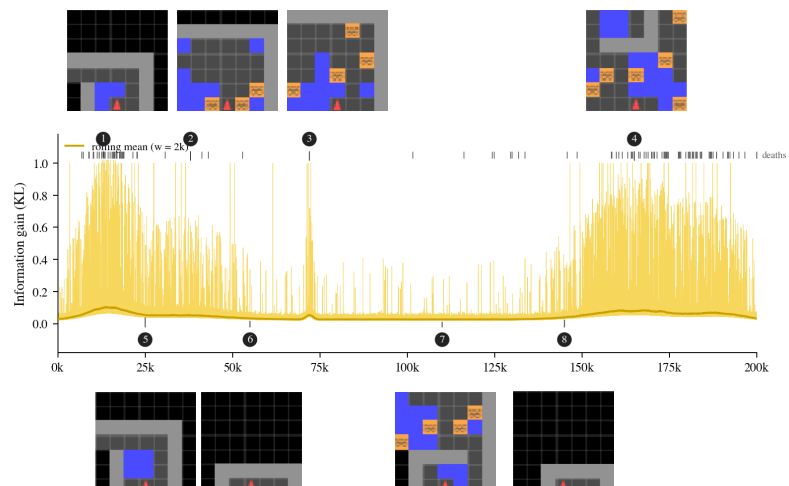
We now examine the feedback loop that is invisible in the Tabular setting: the bidirectional coupling between exploration and representation learning. In the Tabular agent, fixed representations decouple motivation from representation; changes in the reward landscape reflect only changes in the estimated model, not in the state vocabulary itself. In the Dreamer agent, both the model and the vocabulary change simultaneously. To make this feedback loop visible, we conduct trajectory analyses of each motivation signal over the course of training, examining how the quality and character of each reward signal evolves alongside the world model.

Novelty and pseudo-novelty

In the Tabular agent, the novelty reward decays predictably: each visit to a state increments $N(z)$, reducing the negative log-probability of that state. Eventually, all states are saturated and exploration terminates. In the Dreamer agent, this decay does not occur reliably—instead, the agent exhibits *pseudo-novelty*: a sustained high-novelty signal that does not correspond to genuine exploration.

At each training step, updates to the world model parameters ϕ change the encoder $p_\phi(z \mid h, o)$, so a familiar observation may be assigned a new discrete latent code $z'_i \neq z_i$. Because the corresponding count $N(z'_i)$ is low, the agent receives a high novelty reward despite acquiring no new environmental information. Figure 6.16 illustrates this failure mode: novelty remains elevated even as the agent revisits the same walls and floors, with peak rewards caused by shifted latent encodings

Figure 6.17: Information gain trajectory in the Dreamer agent. Per-step KL divergence (posterior vs. prior) during the first 200,000 steps. Information gain is high initially, decays for familiar transitions (e.g. walking into walls, annotated in bottom row), but persists at stochastic ice transitions whose aleatoric variance the world model cannot fully absorb. Eventually, the agent learns enough about the starting room to seek higher rewards outside, guiding it through the bottleneck (deaths visible in the final annotated episode).



12: Pseudo-novelty is a representation-learning artifact: it is absent in the Tabular setting, where fixed representations guarantee that revisiting a state always yields a lower novelty reward than the previous visit.

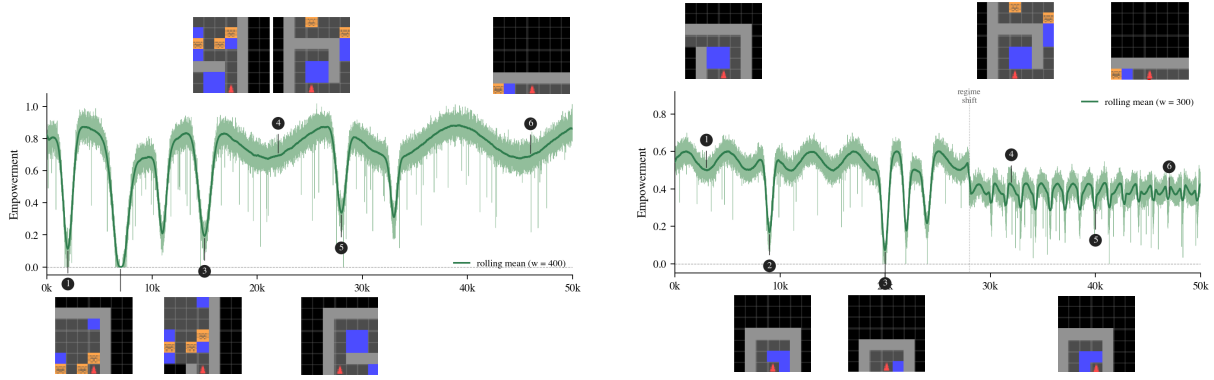
rather than genuinely new states. This creates a self-reinforcing loop in which representational drift produces illusory novelty, weakens true exploration, and limits data diversity, further destabilizing the model¹².

This pathology has direct implications for library curation. A curation policy based on visitation counts—“represent states that have been seen rarely”—operates correctly only when the representational vocabulary is stable. When the vocabulary is under active construction, a “rarely seen” latent encoding may simply be an *unstable* encoding of a familiar observation, not a genuinely new concept. A more robust curation criterion would need to be representation-invariant, or would need to explicitly account for the evolving representational substrate.

Information gain and aleatoric fixation

The information-gain trajectory in Figure 6.17 shows a subtler failure mode. Unlike the novelty agent, it eventually explores, but only after an indirect and costly phase. Early on, information gain is high everywhere because the world model cannot yet predict unfamiliar observations. With repeated exposure, predictable transitions are learned: the encoder posterior aligns with the dynamics prior, causing the KL reward to decay toward zero. This decay is fastest for deterministic transitions, such as walking into walls, and slower on stochastic ice tiles, where inherent variability keeps posterior and prior in disagreement longer.

Ice cells therefore retain elevated information-gain rewards long after repeated exposure. As shown by the annotated timesteps, the agent revisits the same starting-room ice tile from steps 25,000 to 60,000, trying to model dynamics that are inherently unmodelable. This is the noisy-TV problem in a learned-representation setting: unlike the tabular agent, whose information gain vanishes asymptotically even under stochasticity, Dreamer’s retrospective KL conflates irreducible aleatoric variance with genuine model uncertainty. The eventual escape is revealing. After roughly 70,000 steps, the starting-room dynamics are learned well enough that even ice transitions yield less information gain than genuinely unexplored regions beyond the bottleneck. The reward landscape shifts, guiding the agent out; in Figure 6.17, this appears as a cluster of deaths around steps 65,000–85,000 as it first navigates the dangerous bottleneck.



(a) Per-step empowerment during the first 50,000 steps. The reward signal is noisy: while some low-empowerment assessments are correct (annotated ice tile around step 7,000 receives near-zero reward), many empowering floor cells also receive low reward because the world model has not yet learned that actions reliably determine outcomes in those states. The agent defaults to near-random exploration during this early phase.

(b) Per-step empowerment during the final 50,000 steps of 10^6 total. The reward signal has stabilised: floor tiles consistently receive high empowerment and ice tiles near-zero. The agent has learned its comfort zone and exploits it, repeating the same empowering actions (annotated images: agent turning in place or walking in short loops on floor cells) rather than seeking new empowering regions.

Figure 6.18: Empowerment reward trajectory in the Dreamer agent. Early in training (a), the reward signal is noisy as the world model has not yet learned the controllability structure of the environment. By the end of training (b), the signal has stabilised but the agent converges to a comfort zone, repeatedly exploiting known empowering states rather than seeking new ones.

This analysis clarifies the fundamental limitation of retrospective information gain in a learned latent space: the reward tracks *model surprise*, which is partly a function of model quality. As the model improves on deterministic transitions, stochastic ones appear disproportionately surprising—not because they are informationally rich but because they resist prediction.

Empowerment across training

The empowerment trajectory reveals perhaps the sharpest illustration of the feedback loop. Unlike novelty and information gain, whose pathologies manifest in what the agent *seeks*, empowerment’s pathology manifests in what the agent *learns to avoid learning*.

Early training: the calibration problem. Figure 6.18a shows that empowerment is highly noisy during the first 50,000 steps. The world model has not yet learned a stable observation-to-latent mapping, and the factorized dynamics predictor $p_\phi(z'_i | h')$ does not yet capture the environment’s controllability structure. As a result, many genuinely empowering floor tiles receive low scores because the model predicts near-uniform next-latent distributions regardless of action. This mirrors the Tabular uniform-prior problem in Figure 6.11, but with a key difference: Dreamer’s neural dynamics predictor is updated through shared representations and regularization, even for unvisited regions. Its empowerment estimates can therefore improve without direct visitation, but are also noisier and less reliable early in training.

Late training: convergence to the comfort zone. By the end of training (Figure 6.18b), the empowerment reward has stabilized. The world model assigns high empowerment to deterministic floor cells and near-zero empowerment to ice cells, matching the qualitative structure of the Tabular agent’s heatmap in Figure 6.12. Yet the resulting behaviour is dysfunctional: the agent exploits familiar high-empowerment states by turning in place or following short loops on known floor tiles, collecting

reward without expanding exploration. This exposes the limitation of one-step empowerment as an exploration signal. The agent learns where its current action reliably controls the next state, but has no incentive to seek new empowering states whose value is still unknown.

The feedback loop as an object of study. Together, these three trajectory analyses render the representation-motivation feedback loop concrete. The novelty agent is trapped by the instability of the very representation it is trying to count. The information gain agent is delayed by the difficulty of distinguishing what it cannot predict from what is unpredictable. The empowerment agent cannot act on information it has not yet acquired, and once it acquires it, it stops seeking more. In each case, the pathology is not a flaw in the signal's definition but in the interaction between the signal and a changing representational substrate.

6.6 Discussion

6.6.1 Findings

The two experiments in this chapter converge on three claims about how intrinsic motivation shapes exploration and representation.

Complementary failure modes. Curiosity and competence fail in opposite ways. Information gain drives broad, thorough exploration but cannot distinguish epistemic from aleatoric uncertainty: the agent fixates on irreducibly stochastic transitions—the noisy TV problem—and pays for its thoroughness in deaths. Empowerment produces safe, conservative behavior but cannot bootstrap itself: the uniform prior assigns zero empowerment to unseen states, trapping the agent in a self-reinforcing loop where the knowledge it needs to act is precisely the knowledge it has no incentive to acquire. Novelty falls between these extremes, exploring moderately but blindly, indifferent to the structure of what it encounters.

Hybrid complementarity. Combining information gain and empowerment via summation consistently achieves the best exploration-safety trade-off—both in raw discovery-to-death ratio and in downstream generalization across environment types. The mechanism is a natural motivational transition: information gain dominates early, driving the agent outward; as the world model matures, empowerment increasingly constrains attention to controllable regions, dampening the attraction to irreducible stochasticity. The product combination fails because empowerment's initial zero values suppress exploration before it can begin. Crucially, the complementarity extends beyond exploration efficiency to representational quality: hybrid agents build world models that support stronger generalization because the training data covers both learnable and controllable dynamics.

The representation-motivation feedback loop. When representations are learned rather than given, each motivation signal interacts with a changing substrate in signal-specific ways. The novelty agent suffers from pseudo-novelty: representational drift creates the illusion of encountering new states, suppressing genuine exploration. The information-gain agent suffers from prolonged aleatoric fixation: the retrospective KL measure conflates model inadequacy with environmental stochasticity more severely than in the tabular case, because the model is simultaneously learning what to represent and how the world works. The empowerment agent faces a calibration problem: its reward signal is unreliable precisely

when the agent most needs guidance—early in training, when the world model has not yet learned which states are controllable. In each case, the pathology is not a defect in the signal’s definition but an emergent property of computing that signal over representations that are themselves under construction.

6.6.2 Limitations

Grid-world scope. The environments are small, discrete, and fully specified. While this enables precise experimental control over stochasticity and danger, it limits ecological validity. Whether the qualitative patterns identified here—the uniform prior trap, pseudo-novelty, aleatoric fixation—manifest in continuous, high-dimensional, or compositional environments remains an open question.

Unprincipled hybrid combinations. The sum and product combinations are ad hoc. A more satisfying approach would derive the weighting of curiosity and competence from a single normative objective—for example, a resource-rational trade-off between exploration cost and model utility.

Nonstationarity and replay. The Dreamer experiments are impaired by the incompatibility between nonstationary intrinsic rewards and the uniform replay buffer of DreamerV3. Experiences stored early in training carry reward values that are stale by the time they are replayed, introducing systematic bias. The results likely underestimate the potential complementarity between information gain and empowerment in the learned-representation setting. Addressing this through prospective reward recomputation or recency-weighted replay (see [Section C.3](#)) would provide a stronger test.

Human validity. The contribution of this chapter is normative: it characterizes what an optimal agent *would* do under each motivational regime, not what humans actually do. Validating these mechanisms against human behavior—in developmental studies of exploration (Giron et al., 2023; Gopnik, 2024; Nussenbaum and Hartley, 2019) or in controlled video game environments—remains essential to assess psychological plausibility.

6.7 Summary

This chapter showed that what an agent is motivated to seek determines what it learns to represent. Curiosity-driven agents build broad but noisy world models that waste resources on irreducible uncertainty; competence-driven agents build precise but narrow models that ignore everything beyond a learned comfort zone; and hybrid agents—combining information gain with empowerment—build models that are both broad and actionable, supporting generalization across environment types that neither drive achieves alone. When representations are learned alongside the policy, a feedback loop emerges: representational drift creates pseudo-novelty, retrospective surprise conflates model inadequacy with environmental stochasticity, and empowerment estimates depend on a world model that has not yet been built. The central lesson is that motivation and representation cannot be studied independently—each shapes the other, and the pathologies of each signal are emergent properties of this co-evolution. Translating these findings into the library framework of this thesis, several implications emerge.

Intrinsic motivation as a curation policy. Each intrinsic signal implicitly prescribes a different answer to the question “what should the library represent”. Novelty prescribes breadth: represent as many states as possible. Information gain prescribes uncertainty reduction: invest representational resources where the model is most uncertain. Empowerment prescribes utility: represent what the agent can control. The hybrid results show that none of these prescriptions is sufficient alone. Effective curation requires asking two questions jointly: *is this uncertainty reducible?* and *is the reduction actionable?*

The representation instability problem. Pseudo-novelty reveals a fundamental obstacle for curation in systems with learned representations. Any curation criterion defined over a changing vocabulary can be misled by representational drift: a latent state that appears newly encountered may simply be an unstable encoding of a familiar observation. Robust curation criteria must either be representation-invariant or must explicitly discount states whose apparent novelty coincides with periods of rapid representational change.

The retrospective-prospective gap. The tabular agent’s prospective information gain guides exploration *before* visiting a state, enabling planning toward jointly informative and controllable regions. The Dreamer agent’s retrospective KL can only reward transitions *after* they occur, providing no gradient toward unexplored high-information territory. This gap explains the weaker complementarity in the learned-representation setting: the synergy between curiosity and competence is strongest when the agent can anticipate which experiences will be both informative and controllable.

The need for internal empowerment. Empowerment in this chapter measures influence over *external* states—positions in a grid. But for library curation, the relevant question is whether the agent can control *what it represents*. An agent might have high external empowerment (many reachable grid positions) but low representational empowerment (the world model fails to distinguish those positions in a stable, action-sensitive way). The empowerment calibration analysis provides direct evidence for this dissociation: the Dreamer agent’s external behavior and its internal empowerment estimates diverge substantially while the world model is being built. This motivates the shift, taken in the next chapter, from external empowerment to a representation selection principle defined over the agent’s internal dynamics.

The next chapter closes this gap. Rather than asking which grid positions give the agent the most control, it asks which *internal representational structures* maximize the agent’s capacity to predict, distinguish, and act on the consequences of its own decisions. This reframing transforms empowerment from an exploration bonus into a representation selection principle—one that operates directly on the library’s internal organization rather than on the external world the library describes.

Not everything is worth pursuing: agent-centric representation curation

7.

“You have power over your mind—not outside events. Realize this, and you will find strength.”

— Marcus Aurelius, *Meditations*, 8.47

Chapter 6 examined how intrinsic motivation signals shape the representations they are computed over, and showed that the value of a representation, in those models, is read off its consequences in the external environment. Information gain over a world model and mutual information between actions and reachable states both ask what the agent can predict or control out there. Neither asks what the agent’s library makes possible in here.

This chapter separates three objects that are often conflated, namely the inaccessible world W , the environment-specific model M the agent constructs from finite interactions, and the persistent library $\mathcal{L}$ that carries reusable elements across tasks. Within this $W \rightarrow M \rightarrow \mathcal{L}$ pipeline, we develop *representation empowerment* as a construction-and-curation criterion that scores candidate elements by the marginal future capacity to model and plan they confer on the agent. The criterion keeps the channel-capacity template of classical empowerment but redirects both endpoints inward, from external states to the agent’s own representational state. We then ground the criterion in observable behavior through a goal-achievement signature, which is what allows an interacting agent to score it.

We operationalize the criterion in a hierarchical Actor-Curator architecture and test it across three experiments. Section 7.4 reports a causal-learning task with $n = 70$ human participants in a closed vocabulary of observation granularities. Section 7.5 replaces the participant with a normative agent and varies the construction criterion directly, isolating representation selection rather than smarter exploration as the dominant lever. Section 7.6 lifts the closed-vocabulary assumption and tests an LLM-augmented Curator in two grid-world families with PDDL libraries.

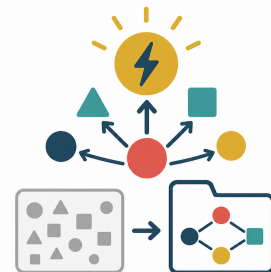
This chapter addresses the curation stage of the library lifecycle, the second of the three symptoms of the representational dynamics gap (Section 2.3.3). Construction (Part II) supplies representational material by building new library entries from experience. The compression chapter (Section 3.6.4) argued that the criterion every empirical chapter wants is prospective, scoring representations against goals the agent has not yet faced rather than against data already in hand. Representation empowerment is the prospective proxy this chapter develops.

7.1 Introduction

There is no ready-made vocabulary for modeling the world. Concepts such as force, mass, and acceleration in physics were *constructed* to explain, predict, and intervene in the world, and the resulting vocabulary, e.g., Newton’s laws, is not the world itself but a compressed representation sufficient for prediction and control. The vocabulary is also *curated* over

7.1 Introduction	125
7.2 Background	127
7.3 Methods	129
7.4 Experiment 1: Closed- vocabulary causal learning in humans	134
7.5 Experiment 2: Matched simulations	137
7.6 Experiment 3: Open- vocabulary symbolic construction	138
7.7 Discussion	141
7.8 Summary	143

Code and experiments available at the
Github repository [meta-empowerment](#)



centuries (Fogarty et al., 2015), with Aristotelian categories giving way to Newtonian mechanics and then to relativity. Each transition was driven by phenomena the old vocabulary could not support, yet some reusable abstractions persisted because they continued to support reasoning in new domains. A child learning to cook follows the same pattern at a smaller scale. She does not build a complete chemical model of food; she constructs just enough (e.g., bigger food needs to cook longer) to achieve her goals, and across recipes she keeps what transferred and discards what was specific to last Tuesday’s dinner.

This is the loop cognitive science has long described, namely that people plan with simplified, task-specific models rather than reconstructing the full complexity of the world (Chen et al., 2026; Ho et al., 2022; Nagy et al., 2025b; Wong et al., 2025), consistent with resource-rational accounts of bounded agents (Gershman et al., 2015; Lieder and Griffiths, 2020; Mantiuk* et al., 2025) and with children entertaining candidate causal structures rather than full reconstructions (Gopnik and Wellman, 2012). What is missing is an operational criterion for when a representational element should be selected, refined, or pruned.

The chapter starts from this question of representational vocabulary. Instead of asking *how do I discover the true model?*, we ask *what representational vocabulary is worth building and maintaining?* Out of the many ways to carve up an environment, only some solutions end up being worth committing to, and we are interested in what governs that choice. This is logically prior to what most of the learning literature addresses. Continual learning asks how to preserve knowledge across tasks but treats the representation as fixed (Kirkpatrick et al., 2017; Rolnick et al., 2019; Serra et al., 2018). World-model learning asks how to discover dynamics but starts from a fixed vocabulary of state variables or latent factors (Ha and Schmidhuber, 2018a; Hafner et al., 2025). Causal discovery recovers edges among given variables (Bramley et al., 2017; Fränken et al., 2022; Ke et al., 2021). LLM-based agents accumulate symbolic knowledge but without principled selection (Tang et al., 2024; Wang et al., 2023). Across these lines of work, attention has been on the process of learning within a representation rather than on deciding which representations are worth building and maintaining.

The $W \rightarrow M \rightarrow \mathcal{L}$ pipeline. We separate three objects that are often conflated. W denotes the inaccessible ground truth of the whole world, M the environment-specific model the agent constructs from finite interactions, and $\mathcal{L}$ the library of reusable elements carried across tasks (Nagy et al., 2025b; Wong et al., 2025). Within a task, the agent performs *construction* ($W \rightarrow M$), proposing and testing representational elements until the model is deemed sufficient for the agent’s current goals¹. Across tasks, it performs *curation* ($M \rightarrow \mathcal{L}$), deciding which elements should be preserved into a library that carries over to the next task².

Representation empowerment. We formalize *representation empowerment* (RepEmp) as a construction criterion that quantifies the marginal increase in goal-achieving capacity from adding a new element to the current model. A resource-rational learner should construct task models M and curate libraries $\mathcal{L}$ that maximize future goal-achieving capacity. This is not the standard definition of empowerment (Klyubin et al., 2005; Modirshanechi et al., 2025; Salge et al., 2014), which is defined in terms of the reachability of external states. RepEmp keeps the same channel-capacity template and redirects both endpoints inward, scoring the agent’s control over its own representational state (Zhou et al., 2025).

1: Throughout, “model” does not mean an increasingly faithful mirror of W . It means an action-supporting compression whose adequacy is evaluated relative to an agent’s goals, resources, and future transfer opportunities.

2: Construction and curation can disagree. A representational element may be useful enough to solve the current task, yet too idiosyncratic to preserve; conversely, an element may be only weakly useful now but valuable because it opens many future model-building options.

What the chapter does. We instantiate the framework as a hierarchical Actor-Curator architecture and test it across three experiments. A finite-vocabulary causal-learning task asks human participants ($n = 70$) to construct causal models at varying abstraction granularities, and their granularity choices are best explained by RepEmp over information-gain or novelty alternatives. The same participants compress toward a goal-sufficient abstraction rather than reconstructing W . Matched simulations replace the human with a normative agent and isolate RepEmp-guided construction, rather than smarter exploration, as the dominant lever for both within-task discovery and cross-task transfer. An open-vocabulary planning experiment lifts the closed-vocabulary assumption: an LLM-augmented Curator builds compact symbolic libraries in PDDL across two grid-world families, with stronger backward and held-out transfer than PPO (Schulman et al., 2017), Motif (Klissarov et al., 2023), and WorldCoder (Tang et al., 2024) baselines, and at substantially lower training cost. Ablating RepEmp (the Hoarder baseline) eliminates the gains.

7.2 Background

7.2.1 The environment-centric trap

The reinforcement-learning paradigm rests on the reward hypothesis (Sutton, Barto, et al., 1998) and on a second commitment that has shaped its theoretical development equally strongly, namely that the agent's primary modeling target is the external environment. Internal representations are instrumental in this view: they matter to the extent that they support reward maximization in a given environment. The framework is extraordinarily productive for narrow task mastery, but it creates a systematic blind spot for the general intelligence this thesis is concerned with. An agent optimized for a specific reward in a well-defined environment learns representations that are tuned to one set of statistical regularities and one reward function, and those representations are organized for prediction and control in that environment rather than for reuse, composition, or transfer.

Intrinsic motivation was introduced precisely to address this limitation. Task-agnostic learning signals encourage exploration and skill acquisition in the absence of extrinsic objectives, and Chapter 6 showed that they produce richer exploration and more varied representational repertoires than extrinsic reward alone. Intrinsic motivation as typically formulated, however, inherits the environment-centric bias. Novelty drives the agent to visit all environmental states. Information gain drives the formation of a more accurate model of the current environment's transition dynamics. State-level empowerment, despite its task-agnostic formulation, measures the agent's influence over external states. The result is that intrinsic motivation behaves as a more sophisticated version of the environment-centric paradigm. The agent learns richer representations, but those representations are still shaped by the regularities of the training environment rather than by properties (compositionality, modularity, transferability) that would support lasting adaptability.

7.2.2 Environmental empowerment and its structural limit

Among intrinsic-motivation signals, state-level empowerment occupies a distinguished position. It is not tied to any specific task or reward function and instead measures a structural property of the agent’s situation, namely the degree to which its actions can influence its future. For an agent at state $s \in S$ with action sequence $a_{1:T}$ and resulting state s' , environmental empowerment is the channel capacity between actions and futures:

$$\text{EnvEmp}(s) = \max_{p(a_{1:T})} \text{MI}(s'; a_{1:T} | s). \quad (7.1)$$

High $\text{EnvEmp}(s)$ means the agent can reliably steer the environment from s into many distinct future states. This is the formal expression of the instrumental convergence argument: regardless of what the agent ultimately needs to achieve, having the capacity to bring about diverse and controllable futures is almost always useful (Turner et al., 2019). Chapter 6 demonstrated that EnvEmp provides a natural safety signal (the agent avoids regions where it has little control) and complements curiosity-based exploration (curiosity reduces uncertainty; empowerment seeks reach).

The same chapter exposed a structural limitation that motivates the rest of this chapter. Equation (7.1) is computed over raw environmental states. Where the state representation is given and appropriate, this works well; where the state representation is itself learned, EnvEmp ’s value depends on the quality of those learned states, which the criterion is in no position to evaluate. The issue is not that the agent fixates on irreducible stochasticity, as in the noisy-TV setup (Burda et al., 2018); it is that the agent optimizes for control over the wrong substrate. An agent that maximizes its capacity to reach many grid cells has high state-level empowerment, but it may never discover that keys open doors, that containers hold objects, or that fire requires fuel, because these causal abstractions live in the agent’s representation rather than in the raw state.

A complementary gap concerns curation. Even where EnvEmp does drive the discovery of useful abstractions, it provides no criterion for deciding which to retain. An agent that has discovered both “keys open doors” and “the blue key is usually in the northeast corner of room 7” has no principled basis under Equation (7.1) for preferring the former (a general operator) over the latter (an environment-specific coincidence). Both expand the set of reachable states in their respective training environments. Only a criterion that evaluates representations across a distribution of tasks can distinguish them, and Equation (7.1) is not such a criterion.

7.2.3 Internalizing the criterion

The two limitations above point to a specific remedy. Rather than asking in which environmental configurations the agent has the most control, we ask which internal representational structures give the agent the most control over its own future cognitive states. The reframing has three immediate consequences for library management.

First, it makes the library itself the object of optimization. EnvEmp treats the library as a tool for predicting and controlling the world. A criterion defined over the agent’s representational state treats the library as the

primary asset, valuable not only because it enables good predictions about one environment but because it can be reconfigured (composed, abstracted, specialized) to address a wide range of future tasks.

Second, it provides a principled curation criterion. A library entry is worth retaining when it contributes to the diversity and controllability of the representational configurations the agent can achieve. Entries that are redundant (replaceable by compositions of other entries), unreliable (their addition produces unpredictable behavior), or environment-specific (useful in one setting but not across a distribution of tasks) lose their place under such a criterion.

Third, it connects the curation problem to the resource-rational frame of [Chapter 2](#) and the rate-distortion view of [Section 3.6.4](#). The compression chapter argued that the criterion every empirical chapter wants is prospective, scoring representations against goals the agent has not yet faced rather than against data already in hand. Representation empowerment, formalized in the next section, is the prospective proxy this chapter develops. What it cannot fix is the residual gap to genuinely unseen environments. What it does fix is the gap between scoring representations on observed data and scoring them on the goal-achieving capacity they would confer, which is the closest available stand-in.

The remainder of the chapter makes this concrete. [Section 7.3](#) defines representation empowerment over the agent’s representational state, grounds the criterion through goals the Curator can probe, and develops the Actor-Curator architecture that operationalizes it. [Sections 7.4 to 7.6](#) then test the framework in human, simulated, and open-vocabulary settings.

7.3 Methods

We turn the lifecycle into a concrete agent. The chapter’s claim is that an entry earns its place in the library by the future capacity to model and plan that it adds; the work of this section is to give that claim a precise object, a tractable scoring rule, and an implementable architecture. We separate three things that prior accounts tend to conflate, define representation empowerment as a criterion over the agent’s own representational state, and show that the criterion can be grounded in the environment through a signature of which goals the model achieves.

7.3.1 The $W \rightarrow M \rightarrow L$ pipeline

Most accounts of model-based learning treat the agent’s model as a single object to be optimized against the world (Ha and Schmidhuber, 2018a; Hafner et al., 2025). A finite-resource agent cannot do this. It cannot recover the world directly; it must maintain a small environment-specific model that supports current goals, and it must keep selected pieces of that model around so that the next environment is not started from scratch (Nagy et al., 2025b; Wong et al., 2025). We therefore separate three objects.

Let W denote the inaccessible ground-truth generative process of the world, partitioned into environments $e_1, e_2, \dots \sim \mathcal{E}$ with environment-specific ground truths W_e that we do not assume are recoverable under finite resources. Inside an episode in environment e , the agent maintains

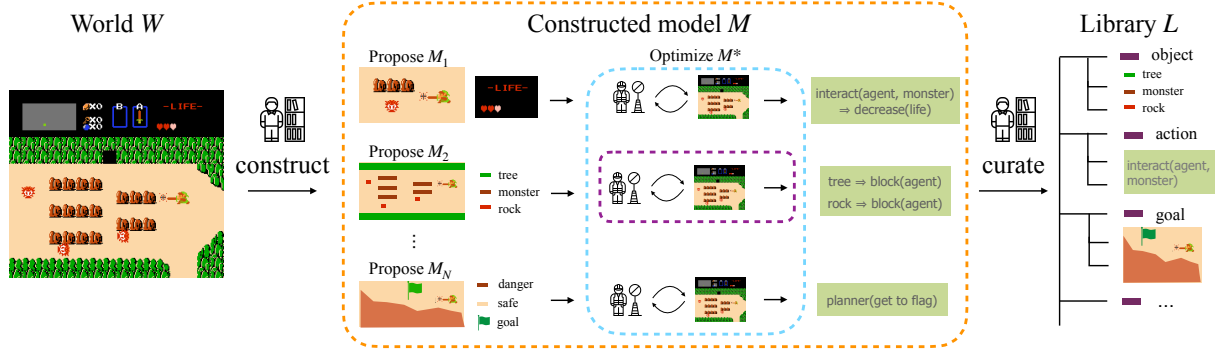
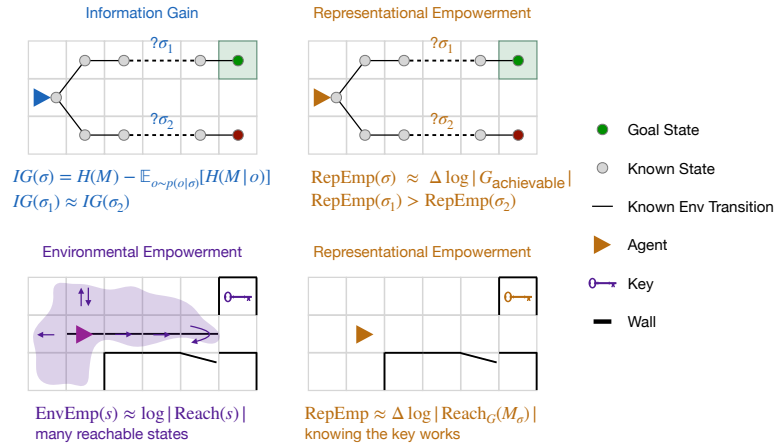


Figure 7.1: Continual model construction as a $W \rightarrow M \rightarrow L$ pipeline. The agent is unable to recover the full world W and instead constructs candidate task models $M_1, \dots, M_N$ optimized against current goals, curating reusable elements into a persistent library $\mathcal{L}$ across environments. (a) Existing world-model learning fits a single monolithic M to all observations. (b) Continual and meta-learning reuse M across environments while leaving the representational vocabulary fixed. (c) The framework of this chapter treats the choice of M and the curation of $\mathcal{L}$ as the primary objects of optimization.

Figure 7.2: Representation empowerment redirects environmental empowerment inward. Information gain rewards uncertainty reduction over the current model’s parameters. Environmental empowerment rewards control over external state visitations. Representation empowerment keeps the same template but redirects both endpoints inward, scoring control over the agent’s own representational state $Z_t = \{M_t, \mathcal{L}_t\}$ rather than over the environment.



an *environment-specific model* $M_e \subseteq \Sigma$ used to plan in the current e . Across episodes, the agent maintains a *persistent library* $\mathcal{L} \subseteq \Sigma$ that carries selected elements forward. Here Σ is a candidate set whose elements $\sigma \in \Sigma$ are units of representational commitment, e.g., an observation granularity, a typed predicate, an operator schema, or a causal edge. The lifecycle stages of Section 2.3.3 have a precise location in this picture. *Construction* adds elements to M_e within an episode, before it is known which will support the agent’s current goals. *Curation* updates $\mathcal{L}$ at the environment boundary by retaining elements that earned their place across M .

Remark 7.1 (Three targets, three errors) Scoring M against W asks for impossible world fidelity. Scoring $\mathcal{L}$ against the current episode overfits to what may not recur. Scoring $\mathcal{L}$ by future goal reach asks the curation question directly: what should be carried forward?

The three objects do different work and live on different timescales, and conflating them is what makes the curation problem look intractable. A criterion that scores M against W asks for fidelity to a target the agent does not see and demands a representational vocabulary it does not yet have. A criterion that scores $\mathcal{L}$ against current data scores it against the very episode whose recurrence it is supposed to outlive. The criterion we want should score elements by the future capacity they add, evaluated through the agent’s own representational state.

7.3.2 Representation empowerment over the agent's representational state

Empowerment, in the classical formulation of Equation (7.1), is a channel-capacity quantity: it scores a state by how reliably the agent can steer the environment from that state into many distinct futures (Klyubin et al., 2005; Modirshanechi et al., 2025; Salge et al., 2014). The construction holds the agent's representation fixed and varies the world; we keep its template and reverse the move, varying the agent's representation while holding the world fixed.

Let $Z_t = (M_t, \mathcal{L}_t)$ denote the agent's representational state at step t . Let Ω be a finite alphabet of *representational edits* on Z_t (e.g., ADD, REMOVE, or REFINE; details in Section D.1.1). An edit sequence $u = \omega_{1:T} \in \Omega^T$ induces a kernel K_T over the resulting representational state Z_T .

Definition 7.1 (Representation empowerment) The *representation empowerment* of a representational state z is the channel capacity between edit sequences and the resulting representational state, $z \in \mathbb{L}$:

$$\text{RepEmp}(z) = \sup_{\pi \in \Delta(\Omega^T)} \text{MI}_{\pi K_T}(U; Z_T \mid Z_0 = z). \quad (7.2)$$

The decomposition makes the two desiderata that RepEmp jointly encodes explicit:

$$\text{MI}(U; Z_T \mid Z_0 = z) = \underbrace{H(Z_T \mid Z_0)}_{\text{diversity}} - \underbrace{H(Z_T \mid Z_0, U)}_{\text{uncontrollability}}. \quad (7.3)$$

The first term measures how many functionally distinct representational states are reachable from z under the edit alphabet, capturing the breadth of the library's future self. The second term is the residual uncertainty about Z_T even after the edit sequence is specified, capturing the unreliability of the edits themselves. A library that is rich in principle but whose edits produce unpredictable outcomes scores poorly under Equation (7.2), and so does one whose edits are reliable but reach only a narrow set of states. RepEmp is high only when the edits are both wide in their reach and predictable in their effect.

The contrast with EnvEmp is summarized in Figure 7.2. Information gain rewards the agent for reducing uncertainty about the parameters of the model it already has (Lindley, 1956; Loewenstein, 1994; Mantiuk* et al., 2025). EnvEmp rewards control over external state visitations (Lidayan et al., 2025; Modirshanechi et al., 2025). RepEmp keeps the same template as EnvEmp but redirects both endpoints inward, scoring the agent's control over its own ability to model the future. The shift from external to internal substrate is what allows the criterion to evaluate libraries across environments rather than within one.

Grounding the criterion in the environment. Equation (7.2) ranges over the agent's representational state Z_T , but the agent has no behavior-free oracle for representations. Two libraries that produce identical actions under all relevant goals are observationally indistinguishable, and the only thing we can observe about a representation is what it lets the agent do in an environment e . We therefore project the criterion through behavior.

Example 7.1 (An edit channel) Suppose the Curator can either add a predicate `openable(x)` or add an operator `unlock(x, y)`. These edits are valuable only if they lead to different future models that unlock different probe goals. If both edits leave the same goals achievable, the grounded channel cannot distinguish them.

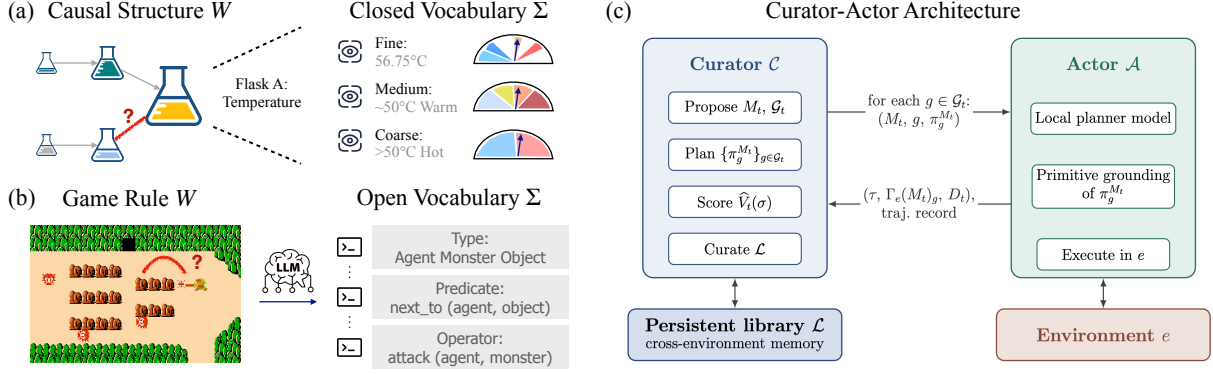


Figure 7.3: The Curator-Actor architecture across closed- and open-vocabulary settings. (a) Closed vocabulary (Section 7.4, Section 7.5): the candidate set Σ is a fixed menu of model granularities, variables, and properties. (b) Open vocabulary (Section 7.6): Σ is an LLM-proposed space of PDDL types, predicates, and operators; promoted elements enter $\mathcal{L}$ and seed M on the next task. (c) The Curator proposes $(M_t, \mathcal{G}_t)$ together with a plan $\pi_g^{M_t}$ per goal, scores candidates by the coverage-gain proxy $\hat{V}_t(\sigma)$, and curates $\mathcal{L}$ at the environment boundary. The Actor grounds each $(M_t, g, \pi_g^{M_t})$ to environment primitives and returns the trajectory τ together with the achievement bit $\Gamma_e(M_t)_g$. The dashed arrow is the only interface between them.

Equip each environment e with a finite family of probe goals $\mathcal{G}_e \subseteq \mathcal{G}$, generated by the Curator (Section 7.3.4), and define the *goal-achievement signature*:

$$\Gamma_e(M) = (\mathbb{1}[g \text{ achieved by planning with } M \text{ and executing in } e])_{g \in \mathcal{G}_e}. \quad (7.4)$$

The signature records which probes the model achieves; a richer model expands which entries of $\Gamma_e(M)$ become 1. The grounded approximation to RepEmp replaces Z_T in Equation (7.2) with the signature it induces:

$$\widehat{\text{RepEmp}}(z; e) = \sup_{\pi \in \Delta(\Omega^T)} \text{MI}_{\pi K_T}(U; \Gamma_e(Z_T) | Z_0 = z). \quad (7.5)$$

By the data-processing inequality, $\widehat{\text{RepEmp}}(z; e) \leq \text{RepEmp}(z)$ (Section D.1.2), and the gap consists of capacity over behaviorally equivalent representations (Proposition D.3). The agent is free to compress this gap away without losing reach over goals. Goal projection is the move that turns RepEmp from an in-principle criterion into one that an interacting agent can score.

7.3.3 The Actor-Curator architecture

Equation (7.5) has an inherent asymmetry. Edits live in the agent’s internal representational state, while Γ_e is realized only through interaction with the environment. The asymmetry motivates a hierarchical split (Figure 7.3c). A high-level *Curator* $\mathcal{C}$ manipulates the representational state $Z = (M, \mathcal{L})$ by proposing, scoring, and selecting elements into $\mathcal{L}$. A low-level *Actor* $\mathcal{A}$ realizes Γ_e in the environment by planning and executing a trajectory τ for each probe goal g and returning the per-goal achievement bit $\Gamma_e(M_t)_g$.

At each step the Curator emits a working model and a probe set,

$$\mathcal{C} : (\mathcal{L}, o_e, \mathcal{D}_t) \mapsto (M_t, \mathcal{G}_t), \quad (7.6)$$

where o_e is the current observation, $\mathcal{D}_t$ is the trajectory evidence accumulated so far, M_t is the Curator’s evolving hypothesis (assembled from

elements of $\mathcal{L}$ relevant to e together with newly proposed candidates), and $\mathcal{G}_t \subseteq \mathcal{G}_e$ is a set of probe goals expressible under M_t . For each $g \in \mathcal{G}_t$ the Curator also produces a symbolic plan $\pi_g^{M_t} = (a_1, \dots, a_k)$ over the operator schemas of M_t . The Actor consumes this and returns trajectory and achievement,

$$\mathcal{A}: (M_t, g, \pi_g^{M_t}) \mapsto (\tau, \Gamma_e(M_t)_g), \quad g \in \mathcal{G}_t. \quad (7.7)$$

The dashed arrow in Figure 7.3c marks the only interface between them.

The architecture mirrors hierarchical reinforcement learning (Dayan and Hinton, 1992; Sutton et al., 1999) with one substitution. Where standard hierarchical RL has the high-level controller emit subgoals over environment states, the Curator emits representational edits, and the high-level reward is a change in Γ_e on the probe set rather than a change in environmental return (Ellis et al., 2021). From the Curator’s perspective the Actor is a black-box realization of Γ_e . From the Actor’s perspective the Curator is a non-stationary source of representations. Two operations close the loop. Compression prunes elements whose removal leaves the signature unchanged (Section D.1.4). Library updates fire at the environment boundary, where the Curator re-scores retained elements against the cross-environment history and updates $\mathcal{L}$ greedily under the same criterion.

7.3.4 The Curator

The Curator has three responsibilities, each tied to a quantity in Sections 7.3.1 and 7.3.2. It *proposes* a task model and probe set (instantiating Σ and $\mathcal{G}_e$). It *scores* candidates by a tractable estimator of $\widehat{\text{RepEmp}}$. And it *selects* elements into $\mathcal{L}$ at the environment boundary.

Propose: $(\mathcal{L}, o_e, \mathcal{D}_t) \rightarrow (M_t, \mathcal{G}_t)$. The Curator’s task model M_t is assembled from two sources, those reused from $\mathcal{L}$ as relevant to e and fresh candidates drawn from a domain-appropriate vocabulary Σ as hypotheses about e . Probe goals are chosen to discriminate M_{t-1} from M_t , which means goals previously blocked but newly expressible under the additions, together with previously solved goals that act as regression checks. Probes are how the Curator measures $\Gamma_e(M_t)$ in practice, and they are the goals it hands to the Actor.

Score: $(M_t, \mathcal{G}_t) \rightarrow \mathbb{R}$. Equation (7.5) is not computable in closed form, since the supremum ranges over Ω^T , the conditioning is on the full representational state, and $\mathcal{G}_e$ may be unbounded. We estimate it with a one-step coverage-gain proxy $\widehat{V}_t$ grounded in execution rather than in surface features of M_t . The Curator queries $\mathcal{A}$ on each probe $g \in \mathcal{G}_t$ and uses the empirical achievement rates of $\Gamma_e(M_t)_g$ as the probabilities $\widehat{p}_t(g \mid M_t)$. For a candidate element σ ,

$$\widehat{V}_t(\sigma) \approx q_t(\sigma) \sum_{g \in W_t(\sigma)} (\widehat{p}_t(g \mid M_t \cup \{\sigma\}) - \widehat{p}_t(g \mid M_t)) - \lambda c_\sigma, \quad (7.8)$$

where $q_t(\sigma)$ is the probability that σ is valid, $W_t(\sigma) \subseteq \mathcal{G}_t$ is its *witness set* of currently unachievable goals that adding σ could unlock, c_σ is the cost of carrying σ , and $\lambda \geq 0$ is a cardinality penalty. A candidate is rewarded only when it produces a measurable change in Γ_e on the probe set, which keeps the score grounded in execution.

Remark 7.2 (What the proxy preserves) The proxy drops the full channel-capacity computation but keeps the load-bearing test: does adding σ reliably flip currently unreachable goals into reachable ones? This is the singleton case of grounded representation empowerment.

The same estimator covers both regimes the chapter studies. When Σ is a fixed model set and $\mathcal{G}_e$ is enumerable, the gain is computed exactly by enumerating Σ , and under deterministic-edge beliefs it coincides with the mutual-information form of Equation (7.5). When Σ is unbounded and $\mathcal{G}_e$ must be sampled, the Curator scores one element at a time and the cardinality penalty λc_σ ($c_\sigma \equiv 1$) ensures elements are retained only when their coverage gain exceeds their cost; the limit $\lambda \rightarrow 0$ recovers the closed-vocabulary form. For multi-candidate proposals we score bundles additively, $\widehat{V}_t(\Delta M) \approx \sum_{\sigma \in \Delta M} \widehat{V}_t(\sigma)$, which is exact when the witness sets $W_t(\sigma)$ are disjoint and otherwise an approximation (Section D.1.3).

Select: $\mathcal{L} \rightarrow \mathcal{L}'$. At the environment boundary, each retained element is re-scored against the rest of the library to remove redundancy: $\widehat{V}_t(\sigma)$ now measures the marginal coverage that σ contributes given $\mathcal{L} \setminus \{\sigma\}$. Re-running the Actor on prior environments would be expensive; instead, $\widehat{p}_t$ is estimated from the trajectory buffer (Ellis et al., 2021) (Section D.1.3). Under a cardinality budget B_{lib} , $\mathcal{L}$ is updated greedily, demoting elements that are subsumed by others or whose reach has degraded as new environments accumulate.

7.3.5 The Actor

Given a symbolic plan $\pi = (a_1, \dots, a_k)$ from the Curator, the Actor's job is to realize π in the current environment and report whether each subgoal was achieved.

Primitive execution. The Actor grounds each operator a_i into environment primitives by heuristic search over a *local planner model*, an agent-egocentric forward model $(\text{state}, \text{primitive}) \rightarrow (\text{state}', \Delta \text{predicates})$ learned online from the Actor's own trajectories. The search heuristic counts unmet effects of a_i under predicted predicates, and the environment verifies the resulting predicate flips. Every executed primitive contributes a training example to the local model, so the local model improves monotonically with interaction at no extra environment budget (Section D.5.1).

Failure attribution. Beyond the realized trajectory and achievement signature, the Actor returns a record of which operator (if any) failed to achieve its effects. This lets the Curator assign blame³. An operator whose effects do not materialize is evidence about the symbolic library, e.g., a missing or miscalibrated predicate. A primitive search that exhausts its budget without reaching those effects is evidence about the local model rather than about the library. Separating the two failure modes is what allows the same coverage-gain score to be diagnostic of representation rather than of execution.

3: Without attribution, a failed probe is ambiguous: the symbolic library may be wrong, or the Actor may simply have failed to ground a good symbolic plan. The architecture needs this split so that coverage gain remains evidence about representation rather than execution noise.

7.4 Experiment 1: Closed-vocabulary causal learning in humans

We test whether RepEmp predicts and produces the behavior the theory describes in a setting where each candidate criterion can be scored against the same enumerable alternatives. The first experiment uses human participants ($n = 70$) in a causal-discovery task with a fixed menu of observation granularities. The closed vocabulary is the methodological lever: a difference in predictions across criteria reflects the criterion itself

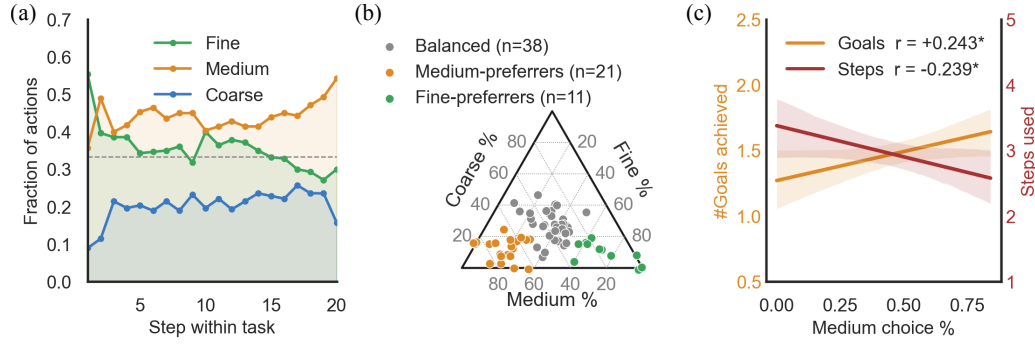


Figure 7.4: Participants compress toward an empowering medium granularity. (a) Fraction of steps at each observation granularity, averaged across participants and plotted against step number within a task. Participants begin with frequent fine use and shift toward medium and coarse as the task progresses. (b) Ternary plot of per-participant granularity-use shares; *k*-means recovers three stable clusters (balanced, medium-preferring, fine-preferring) that persist across tasks even as the dominant granularity adapts. (c) Across participants, a higher medium share predicts both more goals achieved and fewer steps to solution in the held-out tasks. Medium buys reach without overspecifying W .

rather than the noise of approximation. Section 7.5 replaces the human with a normative agent and varies the selection criterion directly, so that human and simulation evidence can be read against the same setup.

The closed-vocabulary causal-learning environment. We adapt the multi-variable causal-discovery environment of Ke et al. (2021). The world W is a fixed structural equation model over eight latent variables. Participants never see W directly; they see overlapping windows of it through tasks. Each step is a two-part action. The participant first picks a *granularity* $\ell \in \{\text{coarse}, \text{medium}, \text{fine}\}$ that sets observation resolution, and then chooses an *intervention* (v, r) on a variable-property pair (e.g., flask A 's temperature or flask B 's weight). The intervention's effects propagate through W and are reported back to the participant at the chosen ℓ . The granularity is the closed-vocabulary realization of the candidate set Σ in Section 7.3.1. The task model M_t is the causal model the participant assembles at the granularity in use, and the persistent library $\mathcal{L}$ is the structure that carries across windows. Every variable additionally carries a *distractor* property: an observable feature that reduces uncertainty about the variable's identity but has no causal effect on other variables. The distractor instantiates the goal-irrelevant element of Section D.1.4, which information gain rewards but representation empowerment correctly ignores. Full environment details, including the SEM, the distractor structure, and the four-task curriculum, are in Section D.2.

Procedure. Participants ($n = 70$; see Section D.3 for recruitment and exclusion criteria) completed four causal-discovery tasks. On each step they picked a granularity ℓ and an intervention (v, r) , observed the outcomes at ℓ , and at the end of each task reported their inferred causal graph. Three questions organize the analysis. (A) Is the granularity choice a structured policy, or is it random or evidence-reactive? (B) Among candidate criteria, which best predicts participants' joint choice of granularity and intervention? (C) Does the constructed model M buy participants goal reach as a compression of W , or do they instead try to recover W ?

7.4.1 Results

Granularity choice is a structured policy. Participants repeat their previous granularity on 67% of consecutive steps, 20.5% above shuffled

Table 7.1: Human granularity-and-intervention choice predictions ($n = 70$). Normalized rank ($< .5$ beats uniform; lower is better) and held-out per-participant cross-validation ΔLL relative to a random baseline (higher is better). The factored model selects granularity with and intervention with ; REP EMP and ENV EMP apply a single criterion to the joint action space.

Family	Model	Norm rank ↓	ΔLL ↑
Empowerment	Factored	0.442	+111.49
	RepEmp	0.482	+91.92
	EnvEmp	0.474	+59.78
Uncertainty	InfoGain	0.556	+38.41
	Novelty	0.522	−791.98
	Random	0.500	0

controls ($p < .001$), which rules out a random policy. A multinomial regression finds that the previous granularity predicts the next ($p < .001$) while the previously observed causal effect does not (Section D.3), which rules out an evidence-driven policy. Consistent with these regressions, k -means clustering on participants’ granularity-choice vectors recovers three clusters: balanced, medium-preferring, and fine-preferring (Figure 7.4b). Cluster membership is stable across tasks (54.5% vs. 42.8% chance, $p < .001$), even as each cluster drifts toward medium. Participants reuse *how* they construct the model while adapting *which* granularity dominates.

Empowerment predicts choices. Each step decomposes into two sub-decisions, a granularity ℓ and an intervention (v, r) . We score each alternative under empowerment-family and uncertainty-family models and fit a per-participant softmax with held-out cross-validation. Table 7.1 summarizes the comparison. The empowerment family dominates, with a factored model (selecting granularity by RepEmp and intervention by EnvEmp) the overall best predictor, ahead of either criterion applied jointly to the flat action space (granularity together with intervention). Uncertainty-family models perform well below chance. The split maps cleanly onto the theoretical interpretations of the two forms of empowerment. RepEmp governs the granularity of abstraction, and EnvEmp informs the agent how to act on the world at that granularity.

Medium is the empowering compression of W . If participants were trying to recover W , they would always select the finest granularity. They do not. Fine selection declines within tasks while medium choices increase (Figure 7.4a, with the per-task and per-step breakdowns in Section D.3.4). When participants re-probe the same causal variable at a different granularity, they asymmetrically coarsen more often than they refine (Section D.3.4). The choice is not arbitrary. A higher share of medium interventions correlates with more goals achieved ($r = +.24$, $p < .05$) and fewer steps ($r = -.24$, $p < .05$; Figure 7.4c). A higher share of fine interventions correlates in the opposite direction, with fewer goals achieved ($r = -.29$, $p < .05$) and more steps ($r = +.31$, $p < .05$). Participants behave as efficient model-constructors: they form policies over M , score modeling actions by their effect on the goal signature Γ_e , and converge on abstractions that support goal achievement rather than try to capture every detail of W . The pattern identifies M rather than W as how participants plan in this environment, and empowerment rather than uncertainty as the criterion under which they evaluate representations. The evidence is correlational at the participant level, however. The next section uses controlled simulations to isolate representation empowerment as the causal lever for the same outcomes.

Family	Model	F1 $\uparrow$	SHD $\downarrow$
Empowerment	Factored	0.75 $\pm$ 0.01	0.48 $\pm$ 0.03
	RepEmp	0.72 $\pm$ 0.01	0.51 $\pm$ 0.03
	EnvEmp	0.52 $\pm$ 0.02	0.97 $\pm$ 0.03
Uncertainty	InfoGain	0.24 $\pm$ 0.005	5.64 $\pm$ 0.26
	Novelty	0.24 $\pm$ 0.005	5.58 $\pm$ 0.26
	Random	0.23 $\pm$ 0.01	4.70 $\pm$ 0.22

Table 7.2: Causal-discovery simulation results, 150 instances per condition. F1 score and structural Hamming distance (SHD) on the recovered graph at the end of training. Mean $\pm$ standard error across instances. Switching the granularity rule from EnvEmp to RepEmp improves F1 by 0.20 ($t(149) = 2.98$, $p < .001$); the factored combination adds a further 0.03. Uncertainty-family criteria collapse to the random baseline.

7.5 Experiment 2: Matched simulations

The human evidence of Section 7.4 establishes that participants behave as RepEmp-guided model-constructors at the population level. What it cannot establish is a causal claim about *why* those choices produce better task outcomes, since participant-level correlations are open to several mechanistic readings. We complement the human study with controlled simulations that share the closed-vocabulary environment of Section 7.4 but replace the participant with a normative agent. Holding everything else fixed and varying only the construction criterion isolates RepEmp-guided granularity selection from smarter exploration over causal variables.

Setup. The simulation mirrors Section 7.4 at scale, with 10 variables and three properties each, a randomly generated DAG, and a procedural curriculum of five training tasks followed by five held-out test tasks across 30 seeds (per-task budget and DAG generation in Section D.4). Each agent observes the same task curriculum and is allowed the same number of interventions; what varies between agents is the criterion that scores the next granularity-and-intervention pair. Five criteria are compared at the granularity slot: random, EnvEmp (Equation (7.1)), RepEmp (Equation (7.2)), Bayesian information gain over candidate DAGs, and a novelty-information-gain hybrid. A factored variant pairs RepEmp at the granularity slot with EnvEmp at the intervention slot, matching the model that best predicted human choices in Section 7.4. We evaluate causal-graph recovery via F1 against the ground-truth DAG and structural Hamming distance (SHD).

7.5.1 Results

Representation selection is the primary contributor. Table 7.2 reports F1 and SHD on the recovered graph at the end of training. Switching the granularity rule from EnvEmp to RepEmp improves F1 by 0.20, from 0.52 to 0.72, a 38% relative gain. Combining RepEmp at the granularity slot with EnvEmp at the intervention slot in the factored model adds a further 0.03 (0.72 $\rightarrow$ 0.75, $t(149) = 2.98$, $p < .001$). The uncertainty family collapses to the random baseline, with Bayesian information gain at 0.24 and novelty at 0.24 against random’s 0.23. The contrast between RepEmp at 0.72 and these alternatives near 0.24 identifies grounded RepEmp (Equation (7.5)) as the mechanism. Smarter exploration only helps when the granularity is efficient and grounded in goal achievement, and the gain RepEmp produces is not matched by heuristics like information gain or novelty that score against external environments.

Why medium wins, and why it supports transfer. The advantage of medium reflects a structural property of the environment rather than the scoring rule. An identifiability analysis (Section D.4.3) shows that under

the per-task intervention budget, only 2.2% of edges are theoretically distinguishable at fine, against 71.1% at medium and 44.4% at coarse. Medium is therefore favored by RepEmp in Equation (7.5) not because of any scoring artifact but because it lies on the empowerment frontier the environment exposes. The same lever drives transfer. Across the held-out task sequence, the factored model improves F1 from 0.66 to 0.77 ($p < .001$). RepEmp alone improves it from 0.66 to 0.73 ($p < .05$). EnvEmp alone shows no transfer ($p = .79$). Selecting the right abstraction is the lever that generalizes, in the sense formalized by Section D.1.5. This is the mechanistic evidence the human study cannot provide: representation selection, not exploration, is the dominant cause of both within-task discovery and cross-task transfer.

The closed-vocabulary studies of Sections 7.4 and 7.5 hold the candidate set Σ fixed by design. Section 7.6 lifts that assumption and asks whether the same selection criterion can build a transferable library when the vocabulary itself must be constructed online.

7.6 Experiment 3: Open-vocabulary symbolic construction

Sections 7.4 and 7.5 held the candidate set Σ fixed by design. This section lifts that assumption. Σ is now generated online from interaction with the environment, and the question becomes whether the same selection criterion can build a transferable library when the vocabulary itself must be chosen. We instantiate the Curator-Actor architecture of Section 7.3.3 with LLM-augmented components in two grid-world families. The environment-specific model M is a PDDL (Aeronautiques et al., 1998) domain (typed predicates, operator schemas, type hierarchy) together with grounding code, and the persistent library $\mathcal{L}$ carries promoted elements across levels.

Three findings organize the section. First, RepEmp-scored curation outperforms baselines on backward and held-out transfer in both environment families, with the strongest gains where library reuse compounds. Second, removing the empowerment criterion (the Hoarder baseline) collapses these gains, which isolates selection from proposer volume. Third, the Curator does its work at substantially lower LLM token cost than the strongest baseline, because the compactness pressure built into the criterion prevents library accretion.

Environments and protocol. We use two grid-world families. In BabyAI (Chevalier-Boisvert et al., 2018), training proceeds along a curriculum ordered by minimum operator-chain length: GoToLocal (1 operator), Pickup (3), OpenRedDoor (3), UnlockLocal (6+). Each environment introduces new operator requirements that compose those from earlier environments. A held-out split tests compositional transfer to configurations that did not appear during training. In Zelda we use the layout from EMPA (Tsividis et al., 2021), with the first three environments for training and the last two held out for testing. The environments naturally become more difficult as new monster types are introduced. Full environment specifications are in Section D.5.2.

Evaluation. After training, $\mathcal{L}$ is frozen. We report *forward transfer* (freezing $\mathcal{L}$ at environment i and evaluating on $i+1$), *backward transfer* (re-evaluating the final $\mathcal{L}$ on each prior training environment), and *held-out generalization* on the held-out split. We compare the Curator against four baselines,

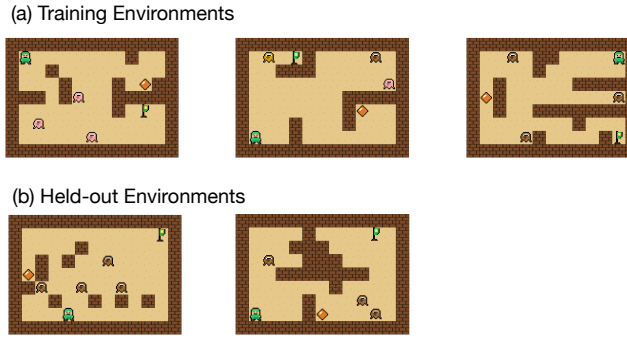
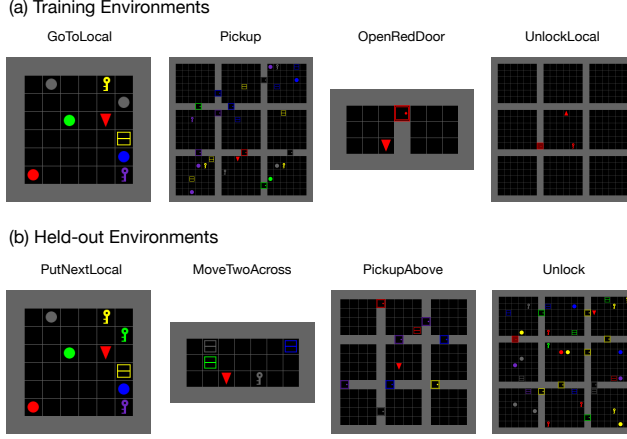


Figure 7.5: Open-vocabulary grid worlds. (a) BabyAI (Chevalier-Boisvert et al., 2018), with a curriculum ordered by minimum operator-chain length: GoToLocal (1), Pickup (3), OpenRedDoor (3), UnlockLocal (6+). Each environment introduces new operator requirements that compose those from earlier environments. (b) Zelda, with the EMPA (Tsividis et al., 2021) layout. The first three environments are used for training, the last two are held out, and difficulty grows as new monster types are introduced.

Table 7.3: Open-vocabulary symbolic construction. (a) Transfer success on BabyAI and Zelda (mean $\pm$ std over three seeds). Forward (Fwd): freezing the library at environment i and evaluating on $i+1$. Backward (Bwd): re-evaluating the final library on each prior training environment. Held-out (H.O.): generalization to environments unseen during training. (b) Training-time LLM cost (total tokens, in millions) and library compactness (carried-forward source code in KB, averaged across the curriculum).

(a) Transfer success							(b) Training cost & footprint		
Method	BabyAI			Zelda			Method	Tokens (M)	Library (KB)
	Fwd.	Bwd.	H.O.	Fwd.	Bwd.	H.O.			
PPO	.29 $\pm$.13	.33 $\pm$.15	.01 $\pm$.00	.03 $\pm$.01	.08 $\pm$.05	.03 $\pm$.02	BabyAI		
Motif	.04 $\pm$.03	.23 $\pm$.10	.12 $\pm$.05	.19 $\pm$.03	.40 $\pm$.05	.18 $\pm$.02	WorldCoder	6.50	18.3
WorldCoder	.32 $\pm$.11	.40 $\pm$.10	.35 $\pm$.10	.35 $\pm$.02	.36 $\pm$.02	.38 $\pm$.02	Curator	5.14	8.0
Hoarder	.20 $\pm$.11	.24 $\pm$.10	.03 $\pm$.01	.07 $\pm$.04	.12 $\pm$.02	.06 $\pm$.00	Zelda		
Curator	.51$\pm$.19	.61$\pm$.07	.47$\pm$.13	.24$\pm$.07	.51$\pm$.06	.48$\pm$.01	WorldCoder	4.13	7.1
							Curator	2.94	4.4

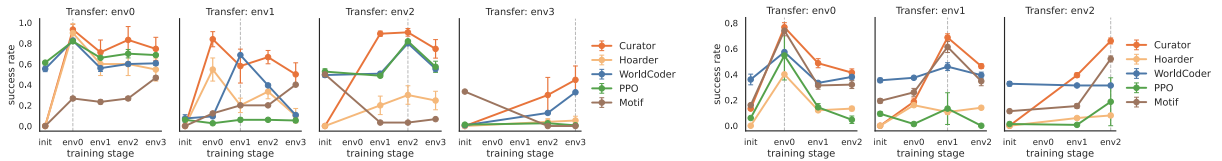


Figure 7.6: Backward transfer across BabyAI and Zelda. Re-evaluating the final library on each prior training environment shows where library reuse compounds across the curriculum. The Curator’s gain is concentrated in backward transfer, where later abstractions retroactively improve performance on earlier environments. Bars are mean over three seeds; error bars are standard deviation.

matched on interaction budget where applicable. PPO (Schulman et al., 2017) learns a primitive-action policy from extrinsic reward. Motif (Klissarov et al., 2023) uses LLM-shaped intrinsic reward but stays at the policy level. WorldCoder (Tang et al., 2024) writes executable world-model code from traces without empowerment-based curation. Hoarder ablates RepEmp by retaining every verified element regardless of its empowerment score, holding proposer volume and verification logic constant. Baseline details are in Section D.5.4.

Transfer. Table 7.3a reports success rates aggregated across three seeds.

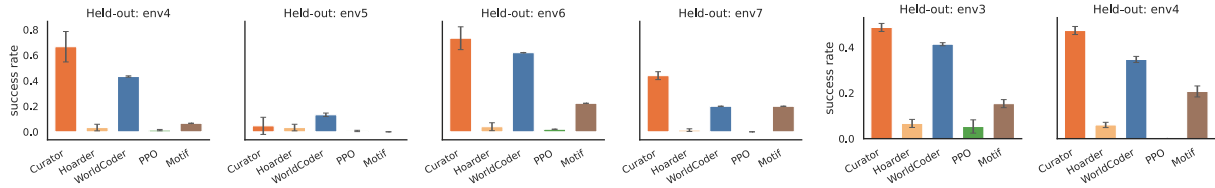


Figure 7.7: Held-out generalization across BabyAI and Zelda. Held-out environments share the same primitives as training but differ in layout and object instantiation. The Curator’s compact library transfers to these unseen configurations more reliably than policy baselines and the Hoarder ablation. Bars are mean over three seeds; error bars are standard deviation.

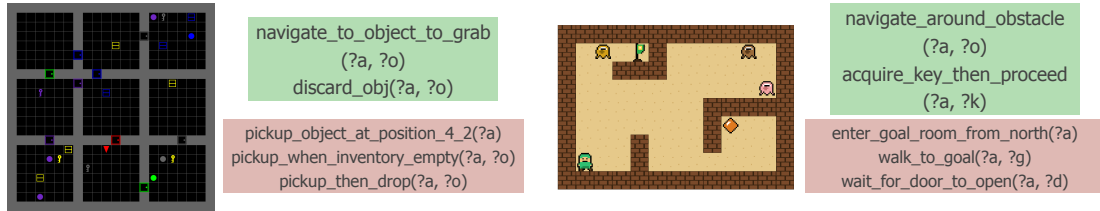


Figure 7.8: Promoted vs. discarded operators in BabyAI (left) and Zelda (right). Green operators are promoted to $\mathcal{L}$ after passing the threshold. Red candidates are discarded despite being reliable on individual probes, because their witness sets are too narrow to earn persistence. General schemas (navigate_to_object_to_grab, acquire_key_then_proceed) earn high , while position- or context-bound candidates (pickup_object_at_position_4_2, enter_goal_room_from_north) are pruned.

The Curator achieves the best transfer and held-out success in most settings, with the strongest gains on backward and held-out transfer in both families. PPO and Motif transfer poorly across environments because their policies are tied to the environment they were trained on. WorldCoder remains competitive and even outperforms the Curator on Zelda forward transfer (.35 vs. .24), but the gap reverses on backward and held-out columns. The Curator’s backward transfer reaches .61 on BabyAI and .51 on Zelda against WorldCoder’s .40 and .36. On the held-out split, the Curator reaches .47 and .48 against WorldCoder’s .35 and .38. Backward and held-out transfer are where compositional reuse should pay off, and the Curator’s abstractions retroactively improve performance on earlier environments and prospectively generalize to unseen ones.

Figure 7.8 illustrates what survives the criterion. General schemas like navigate_to_object_to_grab and acquire_key_then_proceed earn high RepEmp scores, while position- or context-bound candidates such as pickup_object_at_position_4_2 and enter_goal_room_from_north are discarded despite being individually reliable on the probes that exposed them. Their witness sets are too narrow to earn persistence. Absolute success rates remain modest, so we read these numbers as a directional effect of the criterion rather than as a benchmarking claim.

LLM cost and library compactness. The Curator selects sparingly and curates reusable elements at the environment boundary, so both LLM token costs and library size stay bounded across the curriculum. Table 7.3b reports total tokens consumed during training and the size of the carried-forward library given to the planner. WorldCoder’s reward model grows monotonically with the number of environments. On BabyAI it triples from environment 0 to environment 3 because each new environment requires a new branch and prior branches cannot be deleted without breaking earlier goals. The Curator can rebind the same handful of proven typed predicates and operator schemas to new objects, so the library stays roughly 3× smaller and approximately constant in size across the curriculum (per-environment breakdown in Section D.5.7).

The compactness is not a side effect of stinginess; it is what the cardinality penalty λc_σ in Equation (7.8) buys.

Hoarder: empowerment vs. verification. The Hoarder baseline retains every operator that the Curator’s pipeline verifies, without applying the RepEmp score. Table 7.3a shows that the resulting library transfers worse than the empowerment-curated one on every column of both environment families, with held-out success collapsing to .03 on BabyAI and .06 on Zelda. Two distinct costs of uncurated accumulation appear in the trace. First, planning over a library of K operators searches a space exponential in plan length, and irrelevant operators raise the branching factor and the rate of plans that succeed in form but fail in execution. Second, occasional grounding errors compound. An incorrect operator that passes surface-level verification may execute partway through a plan and leave the environment in a state that triggers further failures downstream. The Curator’s RepEmp threshold cuts both costs by demoting operators whose witness sets do not earn their cost. The Hoarder data make the point sharply: the bottleneck is not that the proposer fails to generate the right operators but that no criterion is filtering them.

Failure case. On BabyAI’s held-out split, every method including the Curator drops below 15% success. The held-out environment requires achieving a conjunction of binary inter-object spatial relations of the form “move object A next to B and C next to D ”. Yet every training-environment goal that the LLM proposed was either unary or agent-indexed (e.g., holding, unlocked, agent-at-location). Even when the proposer occasionally generated a candidate next-to($?o1$, $?o2$) predicate, no probe goal in the witness set exercised compositional structure. The criterion behaves correctly given the probes it sees. The probes do not sufficiently cover the space of structurally distinct goals that the held-out environment demands. The bottleneck here is goal diversity in the LLM proposer (Davidson et al., 2022), and lifting it is a limitation we leave to future work.

Takeaway. Across the two grid-world families, the candidate set is unbounded. Keeping every verified element fails (Hoarder), and so does the policy-level approach that ignores symbolic structure (PPO and Motif). RepEmp-scored curation supports transfer in each domain at lower LLM cost than WorldCoder and with structural properties no policy baseline can match. The Hoarder ablation isolates empowerment, distinct from reliability, as the mechanism that produces a transferable library. Absolute success rates remain modest, so we read these results as a directional effect of the criterion rather than as a benchmark. Broader environment families are needed to test how far the principle extends.

7.7 Discussion

The three experiments address complementary aspects of the framework, and their joint implications are richer than any one in isolation. Section 7.4 establishes that human granularity choices are better predicted by RepEmp than by information gain or novelty, with participants compressing toward a goal-sufficient abstraction rather than reconstructing W . Section 7.5 replaces the human with a normative agent and shows that representation selection, not smarter exploration, is the dominant cause of both within-task discovery and cross-task transfer. Section 7.6 extends the framework to an open vocabulary, where an LLM-augmented Curator builds compact PDDL libraries with stronger backward and

held-out transfer than baseline approaches and at substantially lower training cost.

7.7.1 Summary of findings

Goal projection is the load-bearing move. RepEmp in Equation (7.2) is defined over the agent’s representational state and is not directly computable. The agent has no behavior-free oracle for representations; two libraries that produce identical behavior on all relevant goals are observationally indistinguishable. The grounded approximation in Equation (7.5) replaces Z_T with the goal-achievement signature $\Gamma_e(M)$, and the data-processing inequality bounds the gap to behaviorally equivalent representations. The same goal projection is what makes the closed- and open-vocabulary settings amenable to a single estimator. In the closed vocabulary, Γ_e is enumerable and the coverage-gain proxy reduces to mutual information under deterministic-edge beliefs. In the open vocabulary, Γ_e must be sampled and the cardinality penalty in Equation (7.8) becomes load-bearing. Both regimes use the same Actor-Curator interface and the same scoring rule.

Curation beats accumulation, and the difference is not verification. The Hoarder baseline retains every operator that the same pipeline verifies, with the only difference being the absence of the RepEmp score. Table 7.3 shows the resulting library transfers worse than the empowerment-curated one on every column of both grid-world families, with held-out success collapsing to .03 on BabyAI and .06 on Zelda. Two costs of uncuration appear in the trace. Search complexity grows as plans must be selected against a larger operator pool, with irrelevant operators raising the branching factor and the rate of plans that succeed in form but fail in execution. Grounding errors compound when an operator that passes surface verification executes partway through a plan and leaves the environment in a state that triggers further failures downstream. Curation cuts both costs by demoting operators whose witness sets do not earn their cost. The signal is present in the Hoarder library but is buried in noise.

7.7.2 Limitations

The proposer caps the criterion’s reach. The Curator can only score what its proposer generates. The failure case shows what this means in practice. The held-out BabyAI environment requires a conjunction of binary inter-object spatial relations, but every training-environment goal that the LLM proposed was either unary or agent-indexed. Even when the proposer generated a candidate `next-to(?o1, ?o2)` predicate, no probe goal in the witness set exercised compositional structure. The criterion behaved correctly given the probes it saw, but the probes did not sufficiently cover the relevant goal structure. Lifting this limit requires improvements in goal-proposal diversity (Davidson et al., 2022) that are orthogonal to the curation logic developed here.

Two environment families are a directional effect, not a benchmark. The open-vocabulary results span BabyAI and Zelda, both grid worlds with similar primitive vocabularies. Absolute success rates remain modest across all methods, and we read the empowerment-curated library’s advantage as a directional effect of the criterion rather than as a benchmarking claim. Whether the same advantage holds in continuous-action

or partially-observable environments, where the local planner model itself becomes a moving target, is open.

Behavioral correlation, not process-level claim. The human evidence in [Section 7.4](#) is correlational at the participant level. Higher medium share predicts more goals achieved and fewer steps, and the factored RepEmp/ model predicts choices better than uncertainty alternatives, but no part of the data identifies RepEmp as the mechanism participants compute. The mechanistic support comes from the matched simulation in [Section 7.5](#) rather than from the human data alone, so the human study should be read as a behavioral signature consistent with RepEmp rather than as evidence about how the participants compute it.

Computational cost of estimating RepEmp. [Equation \(7.5\)](#) is intractable in closed form, and the coverage-gain proxy in [Equation \(7.8\)](#) is exact only when the witness sets $W_t(\sigma)$ are disjoint. The bundled additive form is otherwise an approximation, and tighter estimators that account for witness-set overlap (e.g., variational bounds on mutual information) would refine the score the Curator uses. The proxy was sufficient for the regimes we tested, but it is not a tight bound on the underlying quantity.

The edit alphabet is fixed. The current framework treats the set of representational edits Ω as given. A more cognitively realistic agent's edit repertoire should itself evolve, with simple edits available early in learning and richer ones (compose, abstract, analogize) emerging from the library that the simple edits have already produced. Treating Ω as a second-order library that co-evolves with the primary library would produce a ratchet, where each improvement in the edit set enables more sophisticated curation, which in turn motivates further refinement of the edit set. We do not address this here.

7.8 Summary

This chapter framed continual model construction as a $W \rightarrow M \rightarrow \mathcal{L}$ pipeline and introduced representation empowerment (RepEmp) as the criterion that decides what enters the environment-specific model M and persists into the library $\mathcal{L}$. An element earns its place to the extent that it expands the plannable goal space per unit of representational cost, and we evaluated this objective at three scales. In a closed-vocabulary causal-learning task, RepEmp predicted human granularity choices better than information-gain or novelty alternatives, and participants compressed toward a goal-sufficient abstraction rather than reconstructing W . A matched simulation isolated representation selection, rather than smarter exploration, as the dominant factor for both within-task discovery and cross-task transfer. In an open-vocabulary symbolic-planning setting, the same objective allowed an LLM-augmented Curator to build compact, transferable PDDL libraries, with the gain concentrated in backward and held-out transfer rather than in single-task success.

Curation as the second lifecycle stage. The curation stage of the library lifecycle is now precise. [Part II](#) produced library entries from experience. The criterion developed here decides which entries earn long-term residency, and it does so by supplying a prospective proxy for the future-distortion gap that the compression chapter left open ([Section 3.6.4](#)). The previous chapter combined information gain and empowerment with hand-tuned weights; representation empowerment subsumes both inside

a single channel-capacity objective whose distortion is goal-achievement rather than reconstruction.

Forward to communication. Diversity and controllability are also the properties that make a library teachable. An abstract operator with predictable effects can be demonstrated, explained, and transferred; a monolithic procedure tied to environment-specific contingencies cannot. The criterion that decides which entries earn long-term residency therefore also shapes which entries can be communicated, and at what cost. **Part IV** takes up this problem, treating explanation as resource-rational communication aimed at library alignment.

Part IV.

Communicate: how structured knowledge
is shared

Explaining how: communication as action enablement

8.

“In time, those Unconscionable Maps no longer satisfied, and the Cartographers Guilds struck a Map of the Empire whose size was that of the Empire, and which coincided point for point with it.”

— Jorge Luis Borges, *On Exactitude in Science*, 1946

The preceding chapters established two faces of individual intelligence: *construction*—how agents build structured representational libraries through inductive learning, and how the order of experience leaves lasting traces in what is acquired, and *curation*—how agents must actively manage these libraries their internal representations under the pressure of resource constraints and intrinsic objectives. Both accounts focused on what happens inside a single agent.

But we do not learn in isolation. Much of what makes human cognition powerful is the ability to transmit knowledge (or representational structure) from one mind to another—through teaching, advice, demonstration, and explanation. When an expert explains how to navigate a building, she is not merely describing a sequence of actions; she is attempting to reshape the listener’s planning process so that the listener can act competently in situations the expert may never observe. This is the third face of the library lifecycle: *communication*. The gap that motivates this chapter is precisely what construction and curation leave unanswered—how does an agent get the right representation into another agent’s head?

The question is, beyond whether an explanation conveys true facts, about whether it equips the recipient to act. We study this through *procedural explanation* in a setting where the asymmetry between explainer and listener is precise: the explainer holds a complete map; the listener navigates with only a local field of view. A useful explanation, we argue, is one that can be compiled into executable guidance—a program over the listener’s planning process that reduces replanning, improves path efficiency, and keeps failures rare. Two experiments test whether this utility criterion predicts human judgment and whether feeding it back to an LLM as an optimization signal generates better explanations.

This framing connects naturally to the thesis arc. If construction is about inducing representational libraries from experience, and curation about deciding what is worth representing, then explanation is about transferring representations deliberately—*library alignment*: the act of installing in another agent’s planning process the abstractions that make good actions easy to find. The same resource-rational logic applies: communicative effort, like representational effort, is finite, and should be invested where it most reduces the recipient’s expected cost of action.

The chapter is organized as follows. [Section 8.1](#) motivates the technical problem and connects it to the field. [Section 8.2](#) reviews related work on pragmatic communication, Bayesian pedagogy, and language grounding. [Section 8.3](#) presents the computational framework. [Section 8.4](#) reports Experiment 1. [Section 8.5](#) presents Experiment 2. [Section 8.6](#) discusses the findings and their implications.

8.1 Introduction	148
8.2 Background	149
8.3 Methods	152
8.4 Experiment 1: Explanation Utility	155
8.5 Experiment 2: Explanation Generation	159
8.6 Discussion	162
8.7 Summary	163

Code and experiments available at the
Github repository [explaining-how](#)



8.1 Introduction

Before you open the box, a friend who has assembled this wardrobe twice gives you a two-minute briefing—not from the manual, but from experience: the two cam locks that must go in before the backing panel, the step that looks optional but will jam the drawer, the wobble that means a dowel is missing. What makes this useful is that it prepares you to act—it gives you the right checkpoints, at the right moments, in a form suited to someone working inside a half-built cabinet.

This is an instance of *procedural explanation*—telling someone *how* to do something, rather than explaining *what* something is or *why* something is the case. Explaining what (“the wardrobe has two cam locks and a backing panel”) or why (“the panel must go in last because it stabilizes the frame”) may be accurate, yet still leave a listener unable to proceed mid-assembly. What the listener needs is not just facts or reasons, but *guidance*: what to do next, when to stop, and which mistakes to anticipate. A procedural explanation succeeds not when it produces correct beliefs, but when it enables effective action under the listener’s actual constraints.

This changes how explanation should be evaluated. Rational models of communication (Frank and Goodman, 2012; Goodman and Frank, 2016) and Bayesian pedagogy (Ho et al., 2016; Shafto et al., 2014) treat explanation as belief updating or hypothesis-space restructuring. These are powerful accounts of conceptual communication, but they assess success primarily in epistemic terms. What is missing is a decision-theoretic account of procedural explanation—one that evaluates explanations by their downstream behavioral consequences for a listener with particular abilities, a particular task, and a particular view of the world.

This chapter proposes such an account. We formalize procedural explanation in a controlled navigation domain: a grid-based dungeon in which an explainer has global knowledge and an explainee must navigate to a hidden goal under partial observability. We evaluate explanations with a utility function defined over their behavioral consequences for a simulated planning agent. The listener is modeled as an agent that converts a natural-language explanation into program-like guidance via a large language model (Brown et al., 2020) acting as a stochastic compiler, then follows this guidance while planning under a POMDP. Explanation quality is measured along three dimensions: how often the listener must revise its plan (*replanning*), whether successful paths are efficient (*path efficiency*), and whether the listener reaches the goal at all (*success rate*).

We test this framework in two experiments. **Experiment 1** collects a corpus of human-written explanations, obtains human helpfulness judgments, measures navigation performance with and without explanations, and evaluates whether the utility model predicts both the subjective judgments and the objective outcomes. **Experiment 2** asks whether the utility framework can do more than evaluate—whether it can *generate*: we use the utility signal as an optimization criterion to guide LLM explanation generation, testing whether the framework produces explanations that are competitive with or superior to the best human-written ones.

The chapter is organized as follows. [Section 8.2](#) reviews the related literature on pragmatic communication, Bayesian pedagogy, and language grounding. [Section 8.3](#) presents the computational framework. [Section 8.4](#) reports Experiment 1. [Section 8.5](#) presents Experiment 2. [Section 8.6](#) discusses the findings and their implications.

8.2 Background

8.2.1 The philosophy and psychology of explanation

What is an explanation? The question has occupied philosophers and psychologists for over a century, but the dominant cases are usually *why*-explanations: accounts that make an event intelligible by locating it within a broader structure of causes, laws, or reasons. By contrast, this chapter is concerned with *how*-explanations: explanations meant to enable action¹.

Explaining why: from laws to causes. The dominant twentieth-century framework, the deductive-nomological (DN) model of Hempel and Oppenheim (1948), treats explanation as logical derivation: an event is explained by showing that it follows from covering laws and initial conditions. A stone falls because all unsupported masses fall, and this mass was unsupported. Despite its influence, the DN model was criticized on multiple fronts. Salmon (2020) argued that causal relevance, not logical entailment, is central: to explain an event is to identify the processes that produced it. Kitcher (1981) emphasized *unification*, reducing the number of independent patterns needed to account for many phenomena. Woodward (2005) developed an interventionist view, according to which explanation identifies what would change under intervention. A parallel tradition addresses *what*-explanations: placing an entity within a category or taxonomy and identifying its distinguishing properties (Barnes, 1994). These traditions differ in detail, but they share a common assumption: explanations are primarily evaluated by correctness, causal adequacy, or conceptual fit.

The psychology of explanation-seeking. In human cognition, explanation-seeking is a basic epistemic drive. Children ask “why?” early and often, and adults evaluate explanations with systematic sensitivity to causal form (Frazier et al., 2009; Kelemen, 2012). For example, Lombrozo (2006) shows that laypeople often prefer *mechanistic* explanations for physical phenomena, but *teleological* explanations for artifacts and biological traits. They also tend to prefer simpler explanations, a bias that can be normatively sensible under some priors (Lombrozo, 2007). For surprising events, explanations citing abnormal conditions are often judged more informative than those citing background conditions (Hilton and Slugoski, 1986).

These preferences matter not only for judgment, but also for learning. Children given explanations for novel patterns acquire more generalizable rules than children given matched descriptions without explanatory language (Walker et al., 2014). Similarly, the *self-explanation effect* (Chi et al., 1994) shows that generating explanations during learning improves comprehension and transfer, likely by promoting integration with existing causal models². At the same time, people often overestimate the depth of their own understanding. Rozenblit and Keil (2002) termed this the *illusion of explanatory depth*: people mistake knowing *that* something works for knowing *how* it works mechanistically.

The underspecified case: how-explanations. One important form of explanation, however, lacks a widely adopted normative framework centered on successful action: *explaining how*—telling someone how to do something: An explanation may be causally precise yet practically useless if it is too abstract, assumes unavailable skills, or omits relevant contingencies. Conversely, an explanation may be imperfectly precise

1: We use “explanation” rather than “instruction” because the object of interest is not merely an imperative utterance, but a communicative act that changes the listener’s internal model in a way that supports successful action.

2: Self-explanation is not the same phenomenon as interpersonal explanation, but it supports the broader claim that explanations matter because they reorganize internal representations in ways that improve later inference and transfer.

Table 8.1: Three common explanatory targets. They differ in the question they answer and in their standard of success. This chapter focuses on *how*-explanations, whose adequacy is determined by downstream action rather than truth-conditional content alone.

Type	Canonical question	Success criterion	Example
<i>Why</i> -explanation	Why did this happen?	Causal adequacy, lawful support, counterfactual relevance	"The glass broke because the force exceeded its tolerance."
<i>What</i> -explanation	What is this?	Correct characterization or categorization	"A whale is a mammal, not a fish."
<i>How</i> -explanation	How do I do this?	Action enablement under the listener's constraints	"At the red door, turn left, then follow the corridor to the stairwell."

yet highly effective if it is well matched to the listener's skills and environment.

Empirical work on *wayfinding* and *procedural instruction* identifies several features of effective how-explanations (Denis, 1997; Tversky, 1996). Route descriptions are often hierarchical, emphasize decision points, and rely on landmarks rather than abstract coordinates. Action-oriented instructions ("turn left at the red door") also outperform purely spatial descriptions ("the room is to your left"). But these findings are largely descriptive: they tell us what useful procedural explanations often contain, without specifying the criterion that makes them useful. This chapter argues that the relevant criterion is *decision-theoretic utility*³: how much an explanation reduces the expected cost of acting under partial observability for a bounded listener.

3: Here "utility" is used in the formal sense: the expected reduction in downstream action cost under a model of the listener, not merely informal helpfulness.

8.2.2 Pragmatic communication

A central insight in pragmatics is that speakers choose utterances strategically, anticipating how listeners will interpret them⁴. Grice's cooperative principle and its associated maxims—be informative, relevant, brief, and clear—capture this idea informally: effective communication requires tailoring an utterance to the listener and the conversational context (Grice, 1975).

The Rational Speech Act (RSA) framework gives this intuition a probabilistic form (Frank and Goodman, 2012; Goodman and Frank, 2016). In the canonical version, a literal listener L_0 updates a prior over world states w using the truth conditions of an utterance u ⁵:

$$L_0(w \mid u) \propto \llbracket u \rrbracket(w) P(w), \quad (8.1)$$

where $\llbracket u \rrbracket(w) \in \{0, 1\}$ indicates whether u is true in w . A pragmatic speaker S_1 then chooses utterances by balancing communicative utility against utterance cost $C(u)$:

$$S_1(u \mid w) \propto \exp(\lambda U(u, w) - C(u)), \quad (8.2)$$

where, in the standard referential case, $U(u, w)$ is often identified with how strongly u leads the listener toward the correct world state. A pragmatic listener L_1 can then reason about such a speaker, yielding recursive pragmatic inference.

RSA has been used to model scalar implicature, reference, vagueness, and metaphor (Goodman and Stuhlmüller, 2013; Lassiter and Goodman, 2017). For the present chapter, two features matter most. First, RSA treats communication as *utility-guided inference*: utterances are chosen for their expected effects on a listener. Second, it makes listener modeling explicit: successful communication depends on reasoning about what the listener knows and how the listener will update.

4: We will use speaker/explainer and listener/explainee interchanably. But we try to make sure there are consistent with the context.

5: This truth-conditional listener is the simplest RSA instantiation. Many extensions enrich the semantics or relax the literal and pragmatic distinction, but the basic speaker-listener recursion is the key idea used here.

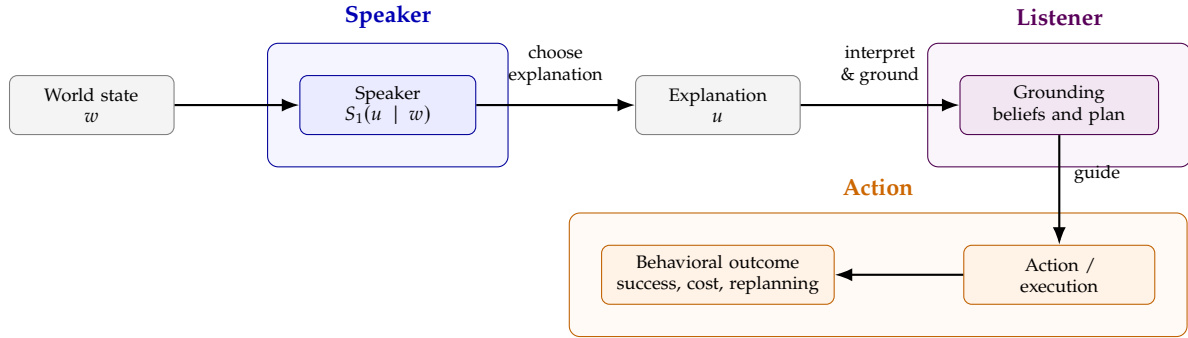


Figure 8.1: Our framework extends the RSA pipeline (Figure 3.2) beyond interpretation. The speaker chooses an explanation for its effect on the listener’s grounded guidance representation, which in turn shapes downstream action and behavioral utility.

However, the utility in standard RSA is usually *epistemic*. The speaker aims to help the listener identify the correct referent, world state, or interpretation. Procedural explanation requires a different target. Here, the goal is not merely that the listener *believe* the right thing, but that the listener acquire an executable guidance representation that supports successful action under partial observability⁶. This chapter therefore extends the RSA intuition from belief-oriented pragmatics to *decision-theoretic* pragmatics: explanations are evaluated by how they change downstream behavior, not just by how they change beliefs.

6: In this setting, what matters is not only which world state the listener infers, but which action-relevant abstraction, policy prior, or plan the utterance induces.

8.2.3 Grounding language in action

Once explanation is treated as action-oriented communication, a further question becomes unavoidable: how does an utterance become executable guidance for the listener? Work in AI, robotics, and cognitive science addresses this grounding problem by studying how natural language is mapped onto actions (Tellex et al., 2011). Early work mapped linguistic structure onto formal action representations in constrained domains. More recent approaches use large language models to translate instructions into code, plans, or subgoal specifications (Ahn et al., 2022; Huang et al., 2022; Liang et al., 2023). In navigation, vision-and-language systems similarly treat language as high-level guidance for action selection (Anderson et al., 2018; Fried et al., 2018).

For the present work, grounding matters because it determines how an explanation is actually used. An explanation may be correct in content yet fail in practice if the listener cannot convert it into an actionable plan given their observations, abilities, and planning limits⁷. This motivates our use of an LLM-based stochastic compiler: explanation quality depends not only on semantic adequacy, but on whether the utterance can be transformed into guidance that improves downstream behavior.

7: In this sense, grounding is the intermediate step between linguistic meaning and behavioral utility: it determines what executable guidance the explanation induces in the listener.

8.2.4 Research gaps

Taken together, these literatures suggest that procedural explanation should be understood as *utility-guided communication grounded in action*. What remains missing is an integrated framework that combines these insights: one that treats explanations as speaker-chosen utterances, grounding as the transformation from language to executable guidance,

and explanation quality as the downstream behavioral utility those utterances produce for a partially informed listener. The next section develops such a framework.

8.3 Methods

We now formalize the problem of procedural explanation in a navigation domain. The framework has four components: (i) a partially observable planning problem that the explaineer must solve; (ii) a translation mechanism that converts natural-language explanations into program-like guidance; (iii) a utility function that scores explanations by their downstream behavioral consequences; and (iv) a speaker model that selects explanations to maximize expected utility.

8.3.1 Navigation under partial observability

Let w denote a fully specified world instance—a map with a layout, a start location, and a goal (treasure). The world defines a state space $\mathcal{S}$, an action space $\mathcal{A}$, and deterministic transitions $P(s_{t+1} \mid s_t, a_t, w)$. The explaineer does not observe w in full; instead, at each time step t , the explaineer receives a local observation $o_t = \mathcal{O}(s_t, w)$ revealing only a neighborhood around its current position. Navigation is thus modeled as a partially observable Markov decision process (POMDP) parameterized by w :

$$\langle \mathcal{S}, \mathcal{A}, P, \mathcal{O}, R, \gamma \rangle, \quad (8.3)$$

where $R(s, a)$ provides a reward upon reaching the goal and a step cost otherwise, and γ is a discount factor. Without any prior guidance, the explaineer must explore under uncertainty, a costly process in environments with dead ends, corridors, and ambiguous structure.

This setup captures the core asymmetry that defines procedural explanation: the explainer has access to w (global knowledge), while the explaineer must act based on o_t (local observations) and whatever guidance the explanation provides. The explanation's value lies precisely in bridging this information gap.

8.3.2 The Rational Speech Act framework for procedural domains

In the standard RSA framework (Section 8.2.2), the speaker's utility is defined over the listener's *beliefs*—the probability that the listener identifies the correct referent or world state. We extend this framework by replacing epistemic utility with *decision-theoretic utility*: the speaker's goal is not to induce correct beliefs, but to enable effective action.

Let $e \in \mathcal{U}$ denote an explanation and $w \in \mathcal{W}$ a world instance. We define explanation utility as:

$$U(e, w) = -w_r \cdot \text{REPLAN}(e, w) - w_l \cdot \text{LEN}_{\min}(e, w) + w_s \cdot \text{SUCC}(e, w), \quad (8.4)$$

where $w_r, w_l, w_s \geq 0$ weight three components that capture distinct aspects of explanation quality:

REPLAN measures the average number of times the listener must abandon and re-derive a plan during execution, reflecting the cost of ambiguous or underspecified guidance:

$$\text{REPLAN}(e, w) := \frac{1}{N} \sum_{i=1}^N R_i(e, w), \quad (8.5)$$

where $R_i(e, w)$ is the number of replanning events in attempt i , and N is the number of independent simulation attempts.

LEN_{min} measures the shortest successful path length, capturing the efficiency of the guidance when it works:

$$\text{LEN}_{\min}(e, w) := \min_{i: S_i(e, w)=1} L_i(e, w), \quad (8.6)$$

where $L_i(e, w)$ is the path length in attempt i and $S_i(e, w) \in \{0, 1\}$ indicates success. If no attempt succeeds, $\text{LEN}_{\min}$ is set to the episode budget as a failure penalty.

Succ measures the fraction of attempts that reach the goal:

$$\text{Succ}(e, w) := \frac{1}{N} \sum_{i=1}^N S_i(e, w). \quad (8.7)$$

The pragmatic speaker then selects explanations according to a softmax over utility:

$$S(e | w) \propto \exp(\lambda \cdot U(e, w)), \quad (8.8)$$

where λ is an inverse temperature parameter⁸. The three utility components correspond to distinct failure modes of procedural explanation: *ambiguity* (high replanning), *inefficiency* (high path length), and *unreliability* (low success rate).

8: The softmax speaker (the S_1 speaker in RSA (Frank and Goodman, 2012)) selects explanations proportional to exponentiated utility. When $\lambda \rightarrow 0$, selection is uniform over all explanations; when $\lambda \rightarrow \infty$, only the highest-utility explanation is chosen. λ thus captures the speaker's sensitivity to utility differences — analogous to a rationality parameter.

8.3.3 LLM translation as a stochastic compiler

A critical component of the framework is the mechanism by which an explanation e is *grounded* into executable guidance for the planning agent. We use a large language model (LLM) as a stochastic compiler that translates natural language into a structured guidance tuple $\tau = (\hat{\pi}_\tau, \hat{V}_\tau)$, where:

- $\hat{\pi}_\tau(a | s)$ is a *policy prior*: a distribution over actions at each planner state, derived from procedural content such as “go left, then turn right”;
- $\hat{V}_\tau(s)$ is a *value map*: an estimate of proximity to the goal at each state, derived from goal-oriented content such as “the treasure is in the northwest corner”.

The planner states $s = \phi(o, w)$ are mapped from local observations o in world w . Because LLM decoding is stochastic, we treat compilation as a conditional distribution $p(\tau | e, w)$, approximated by running the same prompting procedure K times to obtain K compilations $\tau^{(k)} \sim \mathcal{T}(e, w)$.

At each time step t , the agent selects an action by consulting the current guidance:

$$a_t \sim \text{PLAN}(\tau_t, s_t), \quad s_t = \phi(o_t, w). \quad (8.9)$$

If the current compilation fails to yield usable guidance—because it specifies an impossible action or leads into a dead end—the agent triggers a replanning event:

$$\tau_t \leftarrow \mathcal{T}(e, w) \quad \text{whenever} \quad \text{FAIL}(\tau_t, o_t) = 1. \quad (8.10)$$

Each replanning event contributes to the REPLAN cost in Equation (8.5).

This design serves three purposes. First, it makes the grounding step *explicit and evaluable*: we can measure the reliability and variance of interpretation directly. Second, it connects to the program induction framework (Chapter 4): the LLM’s translation from natural language to structured guidance is a form of language-conditioned program synthesis. Third, it captures a real property of explanation use: the same explanation can be interpreted in multiple ways, and some of this variability reflects genuine ambiguity while some is noise.

See Section E.2 for full prompting protocol and examples of compiled guidance tuples.

8.3.4 The dual content of effective explanations

The decomposition of guidance into a policy prior $\hat{\pi}_\tau$ and a value map $\hat{V}_\tau$ reflects a distinction that emerges naturally from how people explain navigation routes. Explanations typically combine two kinds of content:

Policy content. Procedural, step-by-step instructions that specify what to do at each point: “go straight for three steps, then turn left at the corridor.” This content directly constrains action selection, reducing deliberation at each decision point. When accurate and contextually applicable, it produces efficient paths with minimal planning overhead.

Value content. Goal-oriented or landmark-based information that conveys where the objective is relative to the listener’s current position: “the treasure is in the northwest corner” or “head toward the wooden crates.” This content does not specify actions directly but biases search toward promising regions. It is more robust to local deviations—if the listener takes a wrong turn, value information still points in the right direction—but less immediately actionable.

As we show in the empirical results (Section 8.4), the most effective explanations combine both: policy content reduces the cost of immediate action, while value content provides a fallback when policy content is locally inapplicable. This combinatorial structure mirrors the library framework’s emphasis on representations that support both direct execution and flexible recomposition.

8.3.5 From belief update to action support

It is worth pausing to articulate how this framework relates to the thesis-level argument about representational libraries. In the standard RSA setting, the speaker helps the listener identify the correct world state—a problem of *inference*. In our setting, the speaker helps the listener *act* effectively—a problem of *planning under uncertainty*. This shift has several consequences.

First, the relevant representation is no longer a posterior distribution over world states, but a *guidance program* that can drive action: a policy prior and a value map. The explanation’s value lies in its capacity to install useful structure in the listener’s planning process, not in its capacity to convey true propositions.

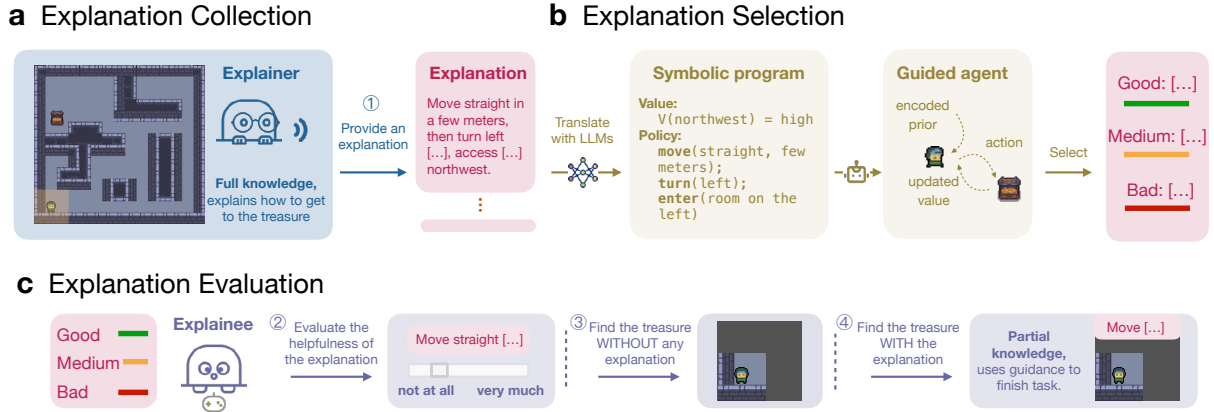


Figure 8.2: Modeling language-guided navigation and the four-study experimental pipeline. (a) Explanation collection: an explainer with full knowledge of the map writes a natural-language message for a partially informed partner. (b) Explanation modeling and selection: free-form explanations are translated into executable guidance and scored by the simulated listener model. (c) Behavioral evaluation: model-ranked explanations are tested through helpfulness judgments, baseline navigation without explanations, and navigation with explanations of varying quality.

Second, the quality of an explanation depends on the *listener's capabilities*—what observations are available, what actions are feasible, and what planning resources can be deployed. This means explanation quality is not an intrinsic property of the utterance, but a relational property that depends on the match between what is communicated and what the listener can do with it. This relational view connects directly to the thesis framework: the same communicated abstraction has different value depending on the agent's existing representational and computational resources.

Third, the framework makes the *cost of communication failure* concrete and measurable. When an explanation is ambiguous, the listener must replan; when it is misleading, the listener follows inefficient routes; when it is incomplete, the listener may fail entirely. These costs are observable behavioral signatures, which we measure in the experiments below.

8.4 Experiment 1: Explanation Utility

Experiment 1 tests the computational framework through four preregistered studies that share a common navigation domain. We first collect a corpus of human-written explanations, use the model to rank them by predicted utility, and then ask whether these rankings predict both perceived helpfulness and actual navigation performance.

8.4.1 Data collection and environments

Navigation environment. The shared task is a grid-based dungeon navigation problem with 24 maps organized into 12 near-matched pairs (Figure E.1). Each map contains a fixed start location and a hidden treasure. Explainers see the full map, whereas explainees act under partial observability with only a local field of view around the current position. Figure 8.3 shows example paired maps and the corresponding behavioral patterns. The full environment specification and the map statistics are given in Section E.1.

a Map Examples



b Rating and Path Length of Participants

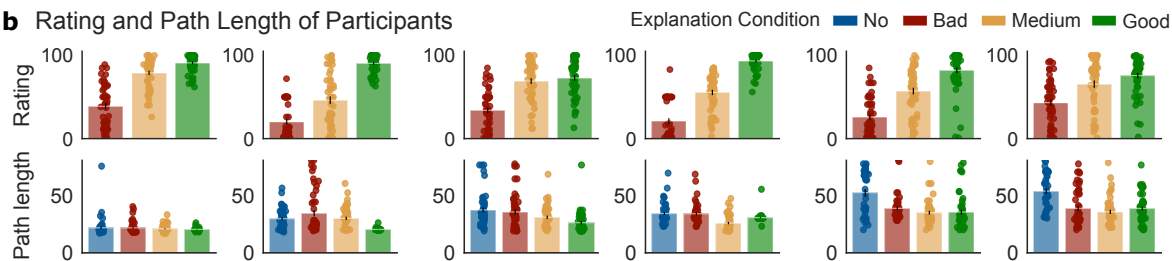


Figure 8.3: Paired map examples and behavioral effects of explanation quality. (a) Example paired maps with matched global layouts but small local structural edits. (b) Path length and helpfulness as a function of condition, illustrating that higher-quality explanations are judged more helpful and tend to support shorter navigation paths.

Study overview. We first collect a corpus of free-text explanations from fully informed explainers and rank them by predicted utility. We then ask a separate participant sample to judge how helpful model-ranked explanations would be for a partially informed partner. Next, we establish a no-explanation baseline by measuring navigation performance under partial observability alone. Finally, we test whether model-ranked explanation quality affects actual navigation performance when explanations are provided. Complete preregistration information can be found in [Section E.3](#).

Model-based explanation ranking. After collecting the explanation corpus, we rank explanations by predicted utility using the compiler and listener model introduced in [Section 8.3](#). Briefly, each explanation is compiled multiple times into executable guidance and scored by the simulated listener using $U(e, w)$. These utility scores are then used to select Good, Medium, and Bad explanations for the downstream behavioral studies.

8.4.2 Corpus description and selection

Collected explanations We recruited $N = 50$ participants to act as explainers. On each trial, participants saw the full map and wrote a free-text message intended to help a partner find the treasure. Each participant wrote one explanation for each of the 24 maps in randomized order. The final corpus contained 1,200 explanations (50 per map), with mean length 21.16 ± 13.68 words. Qualitatively, explanations ranged from value-focused goal descriptions (“top-left corner”), or policy-focused descriptions of either step-by-step action sequences (“go up twice”) or high-level contingencies (“if you see X, do Y”). In a keyword-based coding (e.g., “center” for value, “steps” for low-level policy), both types were often mentioned non-exclusively, such that mixed explanations were most common⁹.

⁹: value: 75.4%; low-level policy: 74.7%; high-level policy: 96.8%; both: 58.3%

To relate explanation style to map structure, we computed graph properties of the reachable state space from the start: (i) shortest-path distance to the goal; (ii) brittleness, measured by the fraction of dead ends (degree = 1) and corridors (degree = 2); (iii) and openness, measured by the

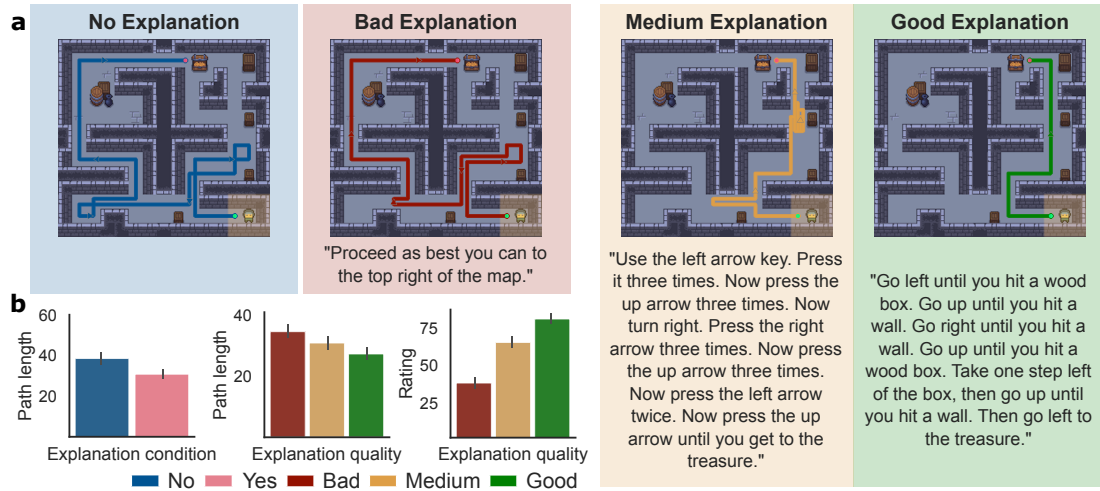


Figure 8.4: Explanations improve navigation efficiency and subjective helpfulness. (a) Example trajectories by condition: no explanation yields inefficient exploration; low-quality explanations are vague procedural; high-quality explanations emphasize relevant landmarks and structure, producing more direct paths. (b) Path length by condition (No/Bad/Medium/Good): higher quality yields shorter paths. Helpfulness ratings increase monotonically with quality. Bars show means; black lines bootstrapped 95% confidence intervals.

fraction of emptiness (degree ≥ 3) (Table E.1). Across maps ($n = 24$), explanation length was unrelated to shortest-path distance¹⁰, but increased strongly with brittleness¹¹, whereby maps with more corridors and dead ends elicited longer explanations. Intuitively, dead ends and narrow corridors make deviations costly, encouraging more explicit, step-by-step guidance. Oppositely, maps with more decision points had shorter explanations¹², consistent with the idea that when many alternative routes are available, goal-level guidance suffices because mistakes are easy to recover from.

Stimulus selection for behavior analysis We ranked all 50 explanations for each map by utility $U(e, w)$ and selected three explanations per map at controlled quality levels: *Good*, *Medium*, and *Bad*. This yielded 72 explanations (3 per map across 24 maps), with mean length 20.98 ± 14.78 words, for use in the subsequent behavioral studies.

10: Spearman $\rho = .08, p = .69$

11: Spearman $\rho = .73, p < .001$

12: Spearman $\rho = -.74, p < .001$

8.4.3 Behavioral analysis

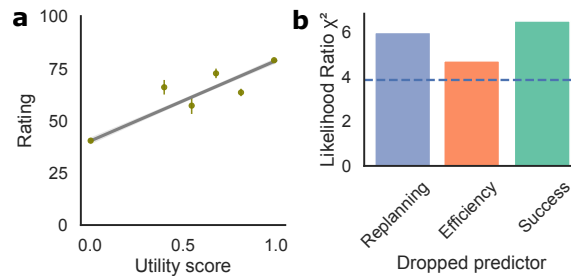
To evaluate model-ranked explanations, we analyzed both subjective judgments of helpfulness and objective navigation performance. Across analyses, we test whether explanation quality (Equation (8.4)) corresponded to human judgments and produced measurable improvements in navigation performance under partial observability.

Explanation utility aligns with human judgments.

We first tested whether explanations categorized by the computational model as *bad*, *medium*, or *good* differed in perceived helpfulness. Figure 8.4b shows averaged ratings by explanation quality. Consistent with preregistered predictions, explanations labeled as *good* were rated as more helpful than *medium* explanations, which in turn were rated higher than *bad* explanations, yielding a clear monotonic ordering¹³. To estimate the size of this effect on a common scale while accounting for repeated measurements, we fit a Bayesian linear mixed-effects model predicting ratings from ordered explanation quality (Bad = 0, Medium = 1, Good = 2),

13: One-sided Welch's t-tests: Good > Medium: $t(49) = 13.25, p < .001, d = 1.86$; Medium > Bad: $t(49) = 18.48, p < .001, d = 2.59$

Figure 8.5: Model comparison and component ablations. (a) Binned mean helpfulness ratings versus utility score; line shows the mixed-effects partial regression (utility effect controlling for base-lines). (b) Leave-one-component-out ablations of the utility model. Bars show the likelihood-ratio statistic (χ^2) comparing each lesioned model to the full model (higher values indicate a larger loss in fit); the dashed line marks the $p = .05$ threshold. All components contribute, so the full model is preferred.



14: $\beta = 17.87$, 95% CI [17.02, 18.74]

with random intercepts for participants and maps. Helpfulness ratings increased with quality¹⁴. Overall, the utility model captures psychologically meaningful variation. Explanations predicted to be more useful for a partially informed listener are consistently judged as more helpful.

Higher-quality explanations produce more efficient navigation.

15: $\beta = -7.73$, 95% CI [-9.05, -6.12]

We tested whether navigation efficiency varies with explanation quality. Relative to no-explanation trials, providing explanations substantially reduced path length overall¹⁵, showing that even imperfect guidance helps under partial observability. Within explanation trials, we fit a Bayesian mixed-effects model with explanation quality entered as an ordinal score (Bad= 0, Medium= 1, Good= 2) and random intercepts for participants and maps. Higher-quality explanations predicted shorter paths¹⁶. Together, these results suggest that explanation quality, defined by our computational model, affects downstream behavior under partial observability. Better explanations improve both perceived helpfulness and objective performance in the navigation task.

16: $\beta = -3.65$, 95% CI [-4.25, -3.00]

8.4.4 Computational model evaluation

We next evaluated whether the full utility model predicts human helpfulness judgments better than simpler alternatives. We scored explanations by simulating how a partially informed listener would translate the explanation into executable guidance and then act.

We compared the full utility model to two variants. *Length-only*: As a simple processing heuristic, we scored explanations by negative word count, $U_{\text{LEN}}(e) \propto -\text{LEN}(e)$. *Direct-action (non-program)*: To test whether an explicit program-like intermediate representation is necessary, we prompted the LLM to output the agent's next action directly from the current observation and explanation, without producing an explicit policy and value representation, and without a replanning loop. We evaluated each explanation over N attempts per map and scored it using success and path length (capped on failure), but *without* replanning cost: $U_{\text{DIRECT}}(e, w) := \delta \text{Succ}(e, w) - \alpha \mathbb{E}[L(e, w)]$, with fixed $\alpha, \delta \geq 0$ across models.

Analyzing what makes a good explanation.

For comparability, we normalized each model's score to [0, 1] within each map and analyzed only explanations that were rated by participants. We then fitted a mixed-effects regression, predicting helpfulness from

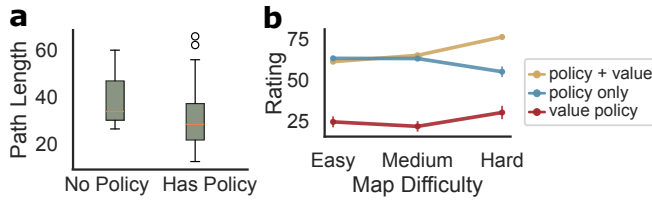


Figure 8.6: Policy vs. value content in explanations. (a) Navigation path length for explanations that contain policy guidance vs. those that do not. (b) Mean helpfulness ratings by map difficulty for explanations containing policy+value information, policy-only information, or value-only information. Error bars show standard errors.

the full model's utility score, a length-only score, and a direct-action score entered simultaneously, with random intercepts for participants and maps. The utility score uniquely predicted higher helpfulness¹⁷. In contrast, the length-only and direct-action predictors were weaker¹⁸. Figure 8.5a shows helpfulness against utility score, along with the partial regression line from the joint model (i.e., the effect of utility holding length and direct-action constant at their means). Overall, helpfulness reflects more than brevity: participants prefer explanations that the utility model predicts will support reliable and efficient guidance. The utility function in Equation (8.4) decomposes explanation quality into three functional terms: replanning, efficiency, and success. To test whether each term contributes to predicting human judgments, we ran leave-one-component-out ablations of the full model. Dropping any term reduced predictive power (Figure 8.5b), with the largest losses coming from removing REPLAN (Equation (8.5)) and Succ (Equation (8.7)). This suggests that people like explanations less that require frequent reinterpretation during execution or that are unreliable. Removing path efficiency had a smaller but reliable impact, suggesting that participants also value explanations that enable direct paths rather than circuitous exploration.

Linguistic strategies and structure.

Lastly, we analyzed rated explanations to characterize strategies associated with higher utility. We coded whether an explanation primarily provided a *policy* (procedural actions), a *value* signal (goal- or landmark-oriented direction), or both. Policy information improved navigation efficiency: trials with policy content yielded shorter paths than trials without policy content (Figure 8.6a)¹⁹. However, helpfulness depended on combining strategies. Explanations integrating policy and value cues were rated highest overall²⁰, although the difference between policy+value and policy-only was not significant ($t = 1.614$, $p = .106$). Importantly, the advantage of combining policy and value increased with map difficulty (Figure 8.6b): policy-only guidance degraded on hard maps, whereas policy+value remained robust.

Overall, these analyses provide converging evidence that our model's notion of communicative utility—grounded in executable guidance for a partially informed listener—captures both the *behavioral* and *linguistic* signatures of what people consider to be good explanations of how to do something.

8.5 Experiment 2: Explanation Generation

Experiment 1 established that the computational utility model captures variation in human-written explanation quality. A stronger test of the framework is whether it can do more than evaluate—whether the utility signal is rich enough to *generate* good explanations when fed back

$$17: \beta_{\text{UTIL}} = 38.47, 95\% \text{ CI} = [36.24, 40.17]$$

$$18: \beta_{\text{LEN}} = .85, 95\% \text{ CI} = [-1.23, 2.94], \\ \beta_{\text{DIRECT}} = .08, 95\% \text{ CI} = [-2.10, 2.28]$$

$$19: 25.09 \pm 7.36 \text{ vs. } 33.83 \pm 12.52; t = 3.20, \\ p < .005, d = .851$$

$$20: \text{policy+value: } 64.23 \pm 31.50; \text{policy-} \\ \text{only: } 62.47 \pm 30.41; \text{value-only: } 24.99 \pm \\ 25.80$$

to the LLM as an optimization criterion. If the utility function truly captures what makes a procedural explanation effective, then using it as a generative target should produce explanations that are competitive with, or better than, the best human-written ones. Experiment 2 tests this hypothesis through a utility-guided explanation generation procedure.

8.5.1 Motivation

The generative test matters for two reasons. First, it provides stronger evidence for the utility framework than evaluation alone: a metric may predict human judgments while relying on corpus-specific surface correlations. If the same utility signal can guide generation toward better explanations across new maps and starting points, it suggests that the signal captures a more substantive property of explanation quality. Second, it is practically useful. Since human explanations are costly to collect and vary in quality, a utility-guided pipeline that reliably generates strong explanations could support tutoring, navigation assistance, and other procedural instruction settings.

8.5.2 Utility-Guided Explanation Generation

We compare three explanation sources on the same 24 maps from Experiment 1:

Human-written (from Exp. 1a). The 50 explanations per map collected from Prolific participants, serving as the human baseline. We report the full distribution of utility scores and the performance of the top- k human explanations (those in the top 10% of utility for each map).

LLM zero-shot. We prompt the same DeepSeek-R1 (32B) model used as the stochastic compiler to now *generate* explanations, without any utility feedback. For each map, we generate $M = 50$ zero-shot candidate explanations under the same decoding settings (temperature = 0.7, top- $p = 0.9$). The prompt is: “You have full knowledge of the following dungeon map. Write a helpful message to a partner who can only see a small area around them. Help them find the treasure.” The map layout is provided in the same format as for the compiler.

LLM utility-guided. Starting from the zero-shot candidates, we run a fixed number of iterative refinement rounds. In each round, we score all current candidates using $U(e, w)$, select the top- K candidates (we use $K = 5$), and prompt the LLM to generate M/K new variants of each selected candidate using utility feedback:

This explanation scored {score} out of 1.0. It failed primarily because: {failure reason derived from the utility components}. Please revise it to improve the listener’s ability to navigate successfully. Specifically: {targeted feedback per utility component}.

The failure reason and targeted feedback are derived automatically from the utility components: high REPLAN triggers feedback about ambiguity and decision-point coverage; high $\text{LEN}_{\min}$ triggers feedback about route efficiency; low SUCC triggers feedback about completeness and obstacle warnings. We run 3 refinement rounds, yielding a final pool of candidates from which we select the highest-utility explanation per map.

8.5.3 Evaluation

Since this is an exploratory simulation, We first evaluate explanations along the computational utility before actually collecting human data. We compute $U(e, w)$ for all candidate explanations using $K = 10$ compilation runs (matching Experiment 1). We report: (i) mean utility across maps for each explanation source; (ii) the top- k utility achieved per map; and (iii) the fraction of maps on which the LLM utility-guided approach matches or exceeds the best human-written explanation.

8.5.4 Predictions

Prediction 1: Utility-guided generation outperforms zero-shot. Iterative refinement with utility feedback will produce explanations with higher mean utility than zero-shot generation, because the feedback directly targets the failure modes (ambiguity, inefficiency, incompleteness) that the utility function measures.

Prediction 2: Utility-guided generation is competitive with top human explanations. The LLM utility-guided pipeline will match or exceed the best human-written explanation on a majority of maps. We expect LLMs to reliably cover both policy and value content (which human writers sometimes omit), producing more consistently complete guidance, while the iterative refinement corrects the LLM’s tendency to produce generic or over-specified instructions.

Prediction 3: Failure-mode feedback drives content-specific improvements. Analysis of the revision trajectories will show that feedback targeting REPLAN leads to more explicit landmark and decision-point coverage, feedback targeting Succ leads to longer explanations with more route-contingency warnings, and feedback targeting $LEN_{\min}$ leads to more direct routing with fewer unnecessary detours mentioned.

8.5.5 Simulation results

In the simulation, utility-guided generation improved explanation quality relative to both the zero-shot LLM baseline and the full distribution of human-written explanations. Across the 24 maps, mean utility was lowest for the full human corpus ($M = .46$, $SD = .21$), higher for zero-shot LLM generations ($M = .57$, $SD = .16$), and highest after utility-guided refinement ($M = .74$, $SD = .11$). The top 10% of human-written explanations remained strong ($M = .71$, $SD = .09$), but the utility-guided LLM matched or exceeded the best human explanation on 15 of 24 maps.

The refinement trajectory showed diminishing returns. Starting from the best zero-shot candidate per map ($M = .62$), mean utility rose to .70 after one refinement round, .73 after two rounds, and .74 after three rounds. The first round mainly reduced replanning cost by making decision points and landmarks more explicit. Later rounds produced smaller gains by removing unnecessary detours and adding warnings about dead ends. This pattern is consistent with the idea that the utility signal is most useful when it identifies a concrete execution failure rather than simply rewarding longer messages.

The component scores showed the same pattern. Relative to zero-shot generation, utility-guided explanations had higher simulated success (.82 vs. .69), shorter paths conditional on success (22.8 vs. 26.4 steps),

and lower replanning cost (1.9 vs. 3.4 replans per episode). The gain was largest on brittle maps, where wrong turns were difficult to recover from. On open maps, zero-shot explanations were already close to the human top- k baseline, and refinement mainly shortened or clarified them.

The generated content also shifted toward the strategy observed in the best human explanations from Experiment 1. Zero-shot generations were often either goal-focused (“the treasure is in the upper-left area”) or over-specified as complete action strings. After refinement, explanations more often combined policy and value information: they gave a local route, named diagnostic landmarks, and included the goal region as a fallback if the listener deviated. In a content coding, policy-plus-value explanations increased from 52% of zero-shot candidates to 79% of utility-guided candidates.

8.6 Discussion

8.6.1 Findings

Procedural explanation quality cannot be measured only by surface features like brevity or fluency, instead, it is more of *pragmatic utility*—how well the explanation can be compiled into executable guidance that supports efficient and reliable action. The full utility model we developed, which accounts for replanning costs, path efficiency, and success rates, predicted human helpfulness judgments substantially better than a length-only heuristic or a direct-action baseline that bypasses the program-like intermediate representation. This confirms that people are also sensitive to the downstream consequences of an explanation.

Dual-content structure of effective explanations. The advantage of combining policy-level procedural instructions with value-level goal information is functionally motivated. Policy content reduces deliberation when conditions are favorable; value content provides robustness when local conditions deviate from what the explainer anticipated. The advantage of combining both strategies increases with environmental complexity, as predicted by the framework.

Resource-rational communicative investment. The utility model predicts different explanations for different map structures. On brittle maps, where a wrong turn is costly to recover from, higher-utility explanations are longer and specify more local decisions. On open maps, where errors are easier to correct, higher-utility explanations can be shorter and focus more on the goal than on step-by-step guidance. Explanation length is therefore not a global marker of quality. Its value depends on how much additional detail improves downstream action in the current environment.

8.6.2 Limitations

Scope of the task domain. We used grid-based dungeons to gain experimental control over partial observability and planning difficulty. This makes the task analytically clean, but it limits external validity. Real procedural explanation often involves richer perception, continuous action, and hierarchical goals, and the same computational logic should be tested in more naturalistic settings and in other procedural domains such as troubleshooting, cooking, assembly, and programming.

Single-shot communication. Our experiments restrict explanation to a single written message. The explainer cannot ask follow-up questions, demonstrate actions, point to locations, or adapt the message once the listener begins to act. Real procedural explanation is often multi-turn and multi-modal. Modeling that richer setting would require extending the framework from one-shot text communication to interactive dialogue and other communicative actions.

Explanation and library construction. A deeper question, left for future work, is how receiving explanations changes the listener's library over time. In the present framework, each explanation is compiled and used within a single episode. But in human learning, explanations often have lasting effects: they introduce new abstractions, establish new conventions, and reshape the listener's representational repertoire for future tasks. Modeling this long-term impact would close the loop between the "Communicate" and "Construct" stages of the lifecycle.

8.6.3 Towards listener-adaptive explanation

The present framework assumes a fixed listener within each episode. That assumption is useful for identifying the utility of a message, but it omits a central feature of real explanation: the explainer is usually uncertain about what the listener knows, and the listener's state changes during the interaction. The next step is therefore to treat the listener's knowledge state as a latent variable to be inferred online rather than as a fixed condition of the task.

A concrete extension would maintain a posterior over listener states, update it from observable signals such as local actions, hesitations, or requests for clarification, and choose explanations against that posterior rather than against a single assumed listener. This turns explanation into a dual problem. The explainer must decide both what guidance is useful under the current belief and whether it is worth spending effort to reduce uncertainty about the listener before giving more guidance. In some cases the best action is immediate instruction. In others it is a short probe, a diagnostic landmark, or a conditional explanation whose main value is to reveal what the listener does not yet understand.

This extension changes the computational role of explanation itself. An explanation will function as an intervention on the listener's knowledge state and as a test of what structure the listener can already use. That is the setting in which explanation becomes fully adaptive.

8.7 Summary

This chapter showed that procedural explanation is a mechanism for communicating representational structure: an effective explanation installs in the listener's planning process a guidance program—a policy prior and value map—that reshapes what the listener finds easy to do. The quality of that guidance is not a surface property of the utterance but a decision-theoretic one, measured by three complementary behavioral consequences: how often the listener must replan, how efficiently successful paths are executed, and whether the goal is reached at all. Experiment 1 confirmed that the computational utility model captures psychologically meaningful variation: model-ranked explanations predict both human helpfulness judgments and objective navigation performance,

outperforming surface baselines (brevity) and non-program-grounded alternatives. Experiment 2 validated the utility framework as more than an evaluator: iterative utility-guided LLM generation produced explanations competitive with the best human-written ones, confirming that the utility signal is informative enough to serve as a generative optimization criterion.

Explanation as library communication. Within the thesis argument, this chapter instantiates the *Communicate* stage of the library lifecycle. Just as construction involves inducing reusable subroutines from experience (Part II), explanation grounding involves inducing executable programs from linguistic input—the same act of program synthesis, now conditioned on language rather than behavioral data. Just as curation asks what is worth representing (Part III), explanation asks what is worth transmitting, and the answer depends on the cost structure of the listener’s task. The chapter establishes that library communication is governed by the same resource-rational principles as library management, and that the quality of a transmitted representation is measured by its downstream functional consequences for the recipient.

The listener as unknown. Both experiments assumed that the listener’s capabilities (field of view, prior knowledge, movement abilities) are either fixed or known to the explainer. But effective explanation in the real world requires more: the explainer must model a listener whose capabilities are not directly observed, must decide when to seek clarification, and must produce guidance robust to residual uncertainty about who is receiving it. These questions—how to infer the listener’s type, when to probe, and how to adapt under uncertainty—extend the utility framework to a Bayesian model of listener-type inference and active probing, and they are the subject of the future work discussed in Part V.

Part V.

Conclusion & Future Directions

General discussion

9.

This thesis opened with a gap. Major traditions in computational cognitive science—symbolic, connectionist, and Bayesian—have formalized learning, inference, and decision, yet all defer the same question: how does new representational structure arise from experience, and how does it reshape what becomes learnable next? This deferral is structural. Computation requires a fixed representational format, allowing learning within that format to be explained while leaving its origin and change unaccounted for. Turing made the computational assumptions precise, but not the representations themselves. This unfinished problem is the representational dynamics gap this thesis addresses.

The thesis adopted a specific theoretical object—the generative library—to make the gap tractable, and studied its lifecycle as the alternation of construction, curation, and communication. Seven empirical chapters instantiated the lifecycle across domains ranging from melodic learning in a controlled experiment to mathematics education at scale, and from simulated agents in grid worlds to humans reasoning about natural language explanations. Each chapter addressed a different aspect of the gap. Together they demonstrate that representational dynamics are a first-class object of scientific inquiry with its own formal structure and empirical predictions, rather than a residual nuisance left over after the rest of cognitive science is done.

This concluding chapter does four things in sequence. [Section 9.1](#) summarizes the work along the construct-curate-communicate axis and states the thesis-level commitment they jointly license—the relocation of the unit of analysis from representations to the processes that manage them. [Section 9.2](#) names what the commitment leaves unresolved at the thesis level. [Section 9.3](#) lays out the research programme the framework opens, organized as three axes that intersect in four flagship projects. [Section 9.4](#) offers a closing reflection.

9.1 What the thesis established	167
9.2 Where current paradigms fall short	174
9.3 The future directions . . .	176
9.4 Closing reflection	183

9.1 What the thesis established

The chapters of this thesis each address a distinct problem: program induction in melodic learning ([Chapter 4](#)), knowledge tracing in educational data ([Chapter 5](#)), intrinsic motivation in exploration ([Chapter 6](#)), representational empowerment in curation ([Chapter 7](#)), and utility-guided explanation ([Chapter 8](#)). Each chapter contributes a model, experiment, or formal framework to its domain. Taken together, they show how representations are constructed, curated, and communicated: how new vocabularies are induced from experience, how they are organized to support agency and learning, and how they are translated for others. This section summarizes these contributions across the three stages and then draws out the common principles that connect them.

Remark 9.1 (The common empirical object) Across chapters, the measured object changes form: programs, prerequisite graphs, world-model latents, curated operators, and explanation-guidance maps. The thesis claim is that these are not separate objects, but different empirical views of representational vocabulary under construction, curation, and communication.

9.1.1 Construction is path-dependent and consequential

The two construction chapters converge on a single claim. The order in which a learner encounters experience determines the representational

vocabulary it builds, and this vocabulary in turn determines which hypotheses are cheap next.

The melodic sequence learning study (Chapter 4) observed this path dependence directly under controlled conditions. Different orderings of training melodies produced different libraries, which in turn supported different patterns of generalization and composition. The fitted grammar recovered learners' internal programs, showing that identical melodic patterns could have different internal representations depending on learning (i.e., construction) history. The curriculum analysis further showed that information-theoretic orderings induce more reusable libraries, turning path dependence from a confound into a design variable. The knowledge tracing study (Chapter 5) observed the same phenomenon at scale and in a naturalistic setting. The prerequisite graph inferred from response histories is itself a statement about construction order: certain knowledge components must be in place before others can be efficiently acquired. Individualized cognitive traits—forgetting rate, transfer ability, prior knowledge—further modulate how strongly that order-dependence binds a particular learner.

Together, these chapters locate path dependence at two complementary levels: within the learner's evolving representational library and within the structure of the curriculum or domain itself. Across melodies and mathematical knowledge components, construction emerges as an active sequencing problem: earlier acquisitions shape the space of later ones, making representational order a mechanism of learning rather than a background condition. This is what makes construction consequential for the rest of the lifecycle. The library a learner ends up with is also the object that the curation and communication stages must work with. Curation cannot reorganize what construction never built, and communication cannot align libraries whose entries do not factor compatibly.

9.1.2 Curation is a primary objective, not a side-effect

The construction chapters show that libraries are built incrementally, but construction alone cannot organize them: a mechanism is still needed to decide what to keep, discard, or reorganize. The curation chapters argue that this active management requires a distinct objective with its own formal criterion—and that the criterion must be internal to the library, not external to the environment.

The intrinsic motivation study (Chapter 6) showed why this internal-external distinction matters. What an agent seeks determines what it learns to represent, yet environment-facing motivation signals can misguide this process. Changing representations can make familiar states appear novel, producing *pseudo-novelty*; retrospective surprise conflates model error with environmental stochasticity; and empowerment over external states assumes the world model the agent is still learning. In each case, the signal is defined over external events, while the object being managed is the internal library.

Curation therefore requires a library-internal criterion: one that scores representations by how they expand the future uses of the library. Chapter 7 formalizes this as *representational empowerment*, the channel capacity between modification operations on the library and the resulting library states. Unlike standard empowerment over external states, this inward measure captures both *diversity*—many functionally distinct configurations are reachable—and *controllability*—the operations

that produce them have predictable effects. Across human experiments, Bayesian causal-discovery simulations, and an LLM-augmented curator, representational empowerment produced libraries that transferred more effectively to novel tasks, especially when environmental empowerment operated at the wrong substrate.

The two chapters thus interlock: the curiosity chapter diagnosed why environment-centric drives cannot guide representational structure, and the empowerment chapter responded by shifting the locus of optimization inward. The hybrid reward of the curiosity chapter—information gain plus empowerment—falls out as a special case of representational empowerment, with information gain reducing the uncontrollability term and environmental empowerment increasing the diversity term. What was an empirical hyperparameter in one chapter recovers as a normative criterion in the next.

9.1.3 Communication extends the lifecycle across agents

Construction and curation concern how an individual learner builds and manages a representational library. Communication introduces the multi-agent version of the same problem: how one agent can reshape another agent's library so that it supports better inference, planning, and action.

The explanation study (Chapter 8) formalizes this as the transmission of guidance programs. An explanation is useful not because it is complete, but because it installs structure that the listener can use: a policy prior and value map that make the right actions cheaper to find. Across human judgments, navigation performance, and LLM-generated explanations, the chapter shows that effective communication depends on aligning what is transmitted with the listener's current library and task demands.

Communication therefore completes the lifecycle. Representations are not only constructed from experience and curated for future use; they are also transferred and aligned across agents.

9.1.4 What the framework implies for the fields

Across the three stages, the dynamics of representational library—what was built, what is being maintained, what is being transmitted—are evaluated according to its functional consequences rather than by its surface form. Treating dynamics as the primary unit of analysis, rather than any particular representation it produces, is the lens I now turn on the three fields the thesis bridges.

Implications for cognitive science

Dissolving the format debate. Cognitive science has long debated which representational format minds use, with classical-symbolic, connectionist, and probabilistic traditions each defending different candidate vocabularies (Fodor and Pylyshyn, 1988; Griffiths et al., 2010; Lake et al., 2017; McClelland et al., 2010; Smolensky, 1988). Our lifecycle account suggests that this debate stalls when format is treated as a premise of the theory rather than as something to be explained. Representational format is not fixed once and for all; it is produced by construction under resource constraints, curation under a representational objective, and pressure

toward communicability. Asking which format is correct is therefore like asking which form is the butterfly: caterpillar, chrysalis, and adult are all real forms, but each is produced by a different developmental stage and serves different functional demands. The more relevant question is which processes and pressures produce a given library. Long-running disputes about modularity (Fodor, 1983), the language of thought (Fodor, 1975; Quilty-Dunn et al., 2023), and the symbolic-versus-subsymbolic divide (Marcus, 2003) can then be recast as questions about the dynamics that yield modular, compositional, or distributed libraries, rather than as commitments to a single static format defended in the abstract.

A generative account of individual differences. Computational cognitive science often models individual variation through participant-specific parameters—priors, learning rates, exploration bonuses, or decision temperatures—within a shared model class (Daw, 2011; Wilson and Collins, 2019). The lifecycle account adds a complementary source of variation: differences in the representational libraries on which those parameters operate. Because library construction is path-dependent, learners who encounter the same evidence in different orders may compile different abstractions, making different hypotheses cheap thereafter. Chapter 4 demonstrates this mechanism experimentally, while Chapter 5 shows that analogous structure can be recovered from naturalistic response data through prerequisite inference and individualized cognitive traits. Individual differences are therefore measurable not only as parameter differences, but also as differences in representational structure¹. This reframes the relation between idiographic and nomothetic accounts of cognition (Molenaar, 2004): shared environments, constraints, or curricula can produce population-level regularities, while divergent histories produce person-specific libraries.

1: This is not meant to replace parameter-based accounts of individual variation. Rather, it distinguishes variation in how a fixed representational library is used from variation in the library itself.

A normative theory of intrinsic motivation. Developmental and motivational psychology have accumulated a catalogue of drives—curiosity, mastery, competence, agency, play—each defended by its own behavioral signatures and each mostly silent on what gets represented as a result (Gopnik, 2012; Kidd and Hayden, 2015; Oudeyer and Kaplan, 2007; Ryan and Deci, 2000). Reinforcement learning has produced a parallel catalogue—novelty, prediction error, information gain, environmental empowerment—selected by benchmark performance rather than derived from a principle (Burda et al., 2018; Klyubin et al., 2005; Pathak et al., 2017; Schmidhuber, 2010). Representational empowerment subsumes both catalogues as special cases of a single optimization process, whereby intrinsic motivation is the policy that selects which library entries should become expressible and controllable. On this view, hypothesis-testing play (Cook et al., 2011; Schulz and Bonawitz, 2007), causal probing (Bonawitz et al., 2011; Gopnik et al., 2004), and preference for moderately complex stimuli (Berlyne, 1960; Kidd et al., 2012) can be understood as forms of representational investment. Such behaviors are valuable when they improve the agent’s future ability to express, distinguish, or control task-relevant possibilities. The framework also makes the role of representational primitives explicit: different libraries may value the same exploratory act differently because they make different future distinctions possible.

A computational-level theory whose object is representation. Marr’s computational level is usually used to specify the task an agent solves, while representational format is assigned to the algorithmic level and justified by behavioral fit (Anderson, 1996; Griffiths et al., 2015; Lieder and Griffiths, 2020; Marr, 2010). The lifecycle account shifts the object of normative analysis: not only which task should be solved, but

which representational library should be built, maintained, and shared. Construction, curation, and communication are the operations; the library is the state variable; and representational empowerment supplies a bounded-resource criterion for evaluating how useful that state can become². This extends resource-rational analysis (Gershman et al., 2015; Lieder and Griffiths, 2020), which asks which computations are worth performing when computation is costly. The lifecycle asks the parallel question for representational resources: which primitives are worth constructing, retaining, or transmitting when time, memory, and search are limited³? This makes format an explanandum rather than an assumption. Symbolic, distributed, modular, or program-like representations can then be treated as outcomes of constructive, curatorial, and communicative pressures, with consequences for which computations are cheap, which hypotheses are expressible, which errors persist, and which coordination problems become solvable.

2: In Marr’s terms, the claim is not that representations cease to have algorithmic realizations, but that the choice of representational resources can itself be analyzed at the computational level.

3: A value-of-computation objective typically evaluates whether to run a computation; the lifecycle generalizes this logic to whether to change the library on which future computations will operate.

Implications for machine learning

An intrinsic target for transfer. Machine learning has known that end-to-end representations transfer poorly outside the training distribution, and the field has responded with different fixes—domain randomization (Tobin et al., 2017), meta-learning (Finn et al., 2017), invariant risk minimization (Arjovsky et al., 2019), and foundation-model pretraining (Bommasani et al., 2021)—each evaluated by held-out benchmark performance rather than by an intrinsic property of the learned representation. This is appropriate because transfer relates a learned representation to future use cases, but it raises a harder question: can an agent estimate a representation’s future usefulness before downstream tasks are specified? Representational empowerment offers one intrinsic proxy, evaluating a library by the distinctions and modifications it makes expressible or controllable under the agent’s own operations, rather than by performance on a specific benchmark. **Chapter 7** shows that, under its witness distribution, optimizing this objective yields libraries that generalize to novel tasks better than reward- or novelty-driven baselines. However, this depends on how the witness set is generated: a poorly calibrated witness distribution may preserve the wrong capacities. Thus, the machine learning contribution is not a task-independent transfer guarantee, but an intrinsic objective that can be optimized before target tasks are known and tested as a predictor of transfer under distribution shift.

Pretraining-data curation needs a representational target. Foundation-model pretraining depends on data-curation pipelines that decide what enters the training corpus. These pipelines use deduplication (Lee et al., 2022), classifier-based quality filtering (Penedo et al., 2023; Wenzek et al., 2020), toxicity or topic exclusion (Gao et al., 2020; Soldaini et al., 2024), and mixture ratios tuned by held-out perplexity on selected evaluation suites (Albalak et al., 2024). These choices are usually evaluated through properties of the corpus or downstream model performance while the construction and curation suggest a complementary target—**Part II** shows that what a learner can represent depends on which experiences are admitted and in what order; **Part III** shows that the usefulness of an induced library can be evaluated by its representational empowerment. Together, these arguments recast data curation as a library-shaping problem: deduplication thresholds, quality cutoffs, and curriculum order should be evaluated not only by input-side statistics or held-out loss, but also by the representational resources they help produce.

4: The claim is not that better reward specification is unnecessary, but that reward specification and representational structure are distinct intervention points.

Reward hacking as a missing-vocabulary pathology. Reinforcement learning has repeatedly shown that agents can exploit reward signals in unintended ways: reward hacking (Amodei et al., 2016), specification gaming (Krakovna et al., 2020), and Goodhart-style failures of learned reward models (Everitt et al., 2021; Pan et al., 2022) are now treated as recurring phenomena rather than isolated bugs. Most responses improve the target being optimized—through reward modeling, RLHF, constitutional feedback, or debate (Bai et al., 2022; Christiano et al., 2017; Irving et al., 2018; Leike et al., 2018). The lifecycle account points to a complementary target: the representational vocabulary through which optimization is carried out⁴. When a scalar reward both defines success and shapes the features used to pursue it, an agent can learn the narrowest vocabulary that exploits the proxy. Human learning is less tightly coupled: representational resources are also shaped by exploration, development, social interaction, and communication, so task rewards operate over a vocabulary built by multiple pressures (Chu and Schulz, 2020; Gopnik, 2020). The lifecycle reframes reward hacking as partly a missing-vocabulary problem. An agent fails not only because the reward is imperfect, but because its library may lack the distinctions needed to pursue the intended objective rather than the proxy. The implication is that alignment can act upstream of reward maximization—by designing curricula, curation objectives, and communicative feedback that build reusable and controllable representations, we may make agents less dependent on any single reward signal and better equipped to generalize beyond it.

Communicability as a trainable design pressure. The previous section located a missing constraint on reward-driven agents in the processes that shape their representational vocabularies. Communication supplies one such constraint: a representation usable by another agent cannot be optimized only for private predictive efficiency. Much interpretability work operates post hoc on representations shaped by prediction loss, reward, or preference optimization: probes test what information is recoverable from internal states (Olah et al., 2020); sparse autoencoders decompose activations into interpretable features (Bricken et al., 2023); and influence functions trace which training examples affect model predictions (Koh and Liang, 2017). These methods reveal structure after training, but they do not by themselves make communicability a training pressure. The utility model of Chapter 8 suggests how to add that pressure. It measures communicability by the downstream effect of a transmitted representation on another agent’s planning. This could serve as an auxiliary objective in multi-agent training, human–AI teaching, or settings where one model’s internals must be legible to another system. Interpretability would then be shaped during training, not only recovered afterward.

Implications for education

Curriculum design as a measurable optimization. Curriculum sequencing in education has so far been governed by expert intuition and domain-specific empirical comparisons. Meanwhile, adaptive tutoring tradition has augmented this practice by sequencing items according to difficulty estimates or recent error rates adaptively (Anderson et al., 1995; Koedinger et al., 2013; Pavlik and Anderson, 2008; VanLehn, 2011). The dominant performance signal focuses on what the learner can currently do, but leaves the question of “which orderings build representations that

support further learning” untouched. **Chapter 4** shows that information-theoretic properties of an ordering predict the reusability of the library it induces, and **Chapter 7** then evaluates whether a candidate library entry would expand the learner’s future capacity, supplying a target that the existing curriculum-design tradition has had to approximate without naming. The contribution to education is to convert curriculum sequencing from a craft leveraging expert intuition into an empirical problem that can be solved using constrained optimization. Here, the construction chapters supply the inference machinery and the curation chapter supplies the objective.

Assessment as inference about what was built. Educational assessment usually begins from observable performance, e.g., test scores, item responses, error patterns, or task products (pellegrino2001knowing; Messick, 1995). Item response theory and Bayesian knowledge tracing make this performance-based tradition more precise, but they still infer ability or knowledge from what the learner currently produces (Corbett and Anderson, 1994b; Lord, 2012b; Pardos and Heffernan, 2011). Validity theory and evidence-centered design have long emphasized the gap between observed performance and the underlying competencies those observations are meant to indicate (Messick, 1995; Mislevy et al., 2003; Pellegrino et al., 1999). The lifecycle account makes that gap the target of inference. Response histories are treated not only as evidence of mastery, but as traces of a structured representational state: which components have been built, which prerequisites organize them, and which learner-specific traits affect their use. **Chapter 5** provides this inference step, recovering prerequisite-structured knowledge states from naturalistic data; **Chapter 7** then evaluates whether the recovered library can support further learning. This is close in spirit to Vygotsky’s zone of proximal development: assessment should estimate not only independent performance, but what the learner is positioned to learn next (Vygotsky, 1978). The contribution is therefore not to reject performance-based assessment, but to treat performance as evidence for a deeper object: the representational library that makes current and future performance possible.

Pedagogy as listener modeling. Education research has studied explanation both as a property of the message and as a response to the learner’s prior state (Chi et al., 1989; Lombrozo, 2006, 2009; Rittle-Johnson et al., 2017). The self-explanation effect shows that learners benefit by actively generating links between new material and prior knowledge (Chi et al., 1989); the curse of knowledge shows that experts often misjudge what novices can infer from the same message (Nathan and Petrosino, 2003). Both phenomena point to that a theory of explanation must model the listener’s representational state, because the same utterance can produce different learning outcomes in different learners. The utility model of **Chapter 8** provides such a listener-state calculus. It evaluates an explanation by its downstream effect when installed in a model of the listener’s planner, making the listener’s prior library an explicit input to the teacher’s optimization problem. The contribution to education is therefore to make “the best explanation” a conditional question: best for which learner, with which current library, facing which task? Pedagogy becomes listener modeling plus message selection, with listener-type inference closing the loop between what is said and what the learner can use.

9.2 Where current paradigms fall short

The individual chapters each acknowledge their own specific limitations. [Chapter 4](#) uses simple melodic sequences with a modest behavioral sample, so the path-dependence claim is established for a designed grammar rather than for naturalistic learning. [Chapter 5](#) works from observational educational logs without causal manipulation, leaving the inferred latent knowledge states identified only up to the assumptions of the generative model. [Chapter 6](#) compares motivational signals inside two agent architectures and a narrow set of grid environments, so the empowerment-as-curiosity claim is shown to outperform alternatives in this regime rather than across the full space of embodied learners. [Chapter 7](#) relies on tractable proxies (goal-space expansion, cross-environment utility) and a planned but uncollected human study, so the curation criterion is computationally operational but not yet behaviorally validated. [Chapter 8](#) tests the utility model in short online tasks, leaving open whether the same effects survive in extended instructional settings where learners interact over weeks rather than minutes.

This section focuses on the limitations that are visible at the thesis level, where the individual caveats compound into structural concerns about the framework as a whole.

9.2.1 The gap between the claim and the evidence

This thesis makes the broad claim that intelligence is the resource-rational lifecycle management of a generative library. The evidence, however, supports a more modest statement: that library-based models capture important aspects of human learning, that representational empowerment provides a useful curation criterion, and that explanation can be formalized as library alignment. Here, the gap between the grand claim and the specific evidence is acknowledged, along with the future work still needed to bridge it.

Domains and scales. The empirical chapters span melodic sequences, mathematics education, grid worlds, and navigation tasks. These are all structured, discrete, and procedural domains where the library metaphor fits quite naturally. The thesis does not provide evidence for library-based accounts of perception (Clark, 2013; Rao and Ballard, 1999), motor control (Todorov and Jordan, 2002; Wolpert et al., 2011), emotional regulation (Barrett, 2017), or social reasoning (Baker et al., 2017; Saxe and Kanwisher, 2013)—domains where the dominant theoretical objects (e.g., predictive hierarchies, control policies, constructed affect, or mentalizing inferences) sit awkwardly with the “reusable compositional building blocks” framing. Whether the library lifecycle extends to these domains, or whether different theoretical objects are needed, is still an open question.

The lifecycle as a coupled system. The strongest theoretical claim—that construction, curation, and communication form a self-modifying loop—is not holistically tested in any single experiment. Each chapter studies one stage of the lifecycle in isolation, or at most two stages in interaction, as in [Chapter 6](#)’s motivation-representation feedback loop. A stronger test would require agents who construct a library from experience, curate it for future use, communicate parts of it to others, and revise it across cycles. Such a design would test whether the stages interact, rather than merely work in isolation: whether constructed libraries

become easier to curate, curated libraries become easier to communicate, and communicated structure improves later construction. It would also allow direct comparisons against alternatives with fixed representations, construction without curation, curation without communication, or communication that transmits answers rather than reusable structure.

9.2.2 Formalization choices

Programs as the privileged representational format. The thesis treats programs—executable, compositional, and inspectable procedures—as the canonical format for library entries⁵. This is a principled choice: programs satisfy the four criteria of [Section 2.4](#): compositionality, inspectability, executability, and communicability. However, it is not the only possible choice. Neural modules (Andreas et al., 2016; Rosenbaum et al., 2019), causal graphs (Pearl, 2009; Spirtes et al., 2000), and skill policies (Bacon et al., 2017; Sutton et al., 1999) are alternative formats that satisfy some, but not all criteria. The thesis does not systematically compare program-based libraries against these alternatives, and it is possible that hybrid formats—programs augmented with learned components (Ellis et al., 2021; Mao et al., 2019; Valkov et al., 2018), or neural modules with programmatic interfaces (Gaunt et al., 2017; Johnson et al., 2017)—would better capture the full range of representational dynamics.

5: “Program” is used here functionally rather than syntactically: the relevant property is that an entry can be executed, recombined, inspected, and communicated, not that it must be written in a human programming language.

Representational empowerment as a proxy. Representational empowerment is defined as the channel capacity between modification operations and resulting library states ([Equation \(7.5\)](#)). This is a clean formal criterion, but its practical computation requires either enumeration of library states (tractable only for small libraries) or Monte Carlo approximation (which introduces estimation error). The proxy scores used in the empowerment experiments—goal-space expansion and cross-environment utility—are interpretable but imprecise. It thus remains an open question whether more accurate estimators of representational empowerment would produce qualitatively different curation decisions (Abel et al., 2026).

Resource rationality without an explicit resource model. The thesis invokes resource rationality throughout: agents operate under bounded computation, memory, and communication bandwidth. Yet the framework does not yet specify the resource being spent, its cost, or the architecture in which the cost is incurred. Without such a model, capacity constraints can justify why agents cannot represent everything, but they cannot by themselves show that a particular library-management strategy is optimal rather than simply selected by the modeler. In this thesis, constraints such as $K_{\max}$ on library size, intervention budgets in empowerment experiments, and channel-capacity limits in communication models are treated as hyperparameters⁶. A fuller account would connect these quantities to established resource models: bounded optimality and metareasoning over computational cost (Russell and Subramanian, 1994), resource-rational time-accuracy trade-offs (Griffiths et al., 2015; Lieder and Griffiths, 2020), rate-distortion and information-bottleneck accounts of capacity-limited cognition (Sims, 2016; Zenon et al., 2019), or architectures that directly price memory and control (Howes et al., 2009; Kool et al., 2010). The next step is to derive library size, modification budgets, and communication bandwidth from such costs, so that construction, curation, and communication policies emerge as resource-rational consequences rather than imposed constraints.

6: For instance, a fixed $K_{\max}$ says that the library cannot grow without bound, but it does not explain why the bound should have that value, what cognitive resource it reflects, or what trade-off is incurred by increasing it.

9.2.3 Empirical scope

The human experiments are suggestive, but still not conclusive. The human behavioral data in the thesis comes from controlled settings or existing large-scale educational logs. However, none of these settings fully tests the developmental predictions of the library framework: that children’s representational reorganization follows the trajectory predicted by representational empowerment, that expert-novice differences reflect differences in library organization rather than merely differences in library content, and that teaching effectiveness depends on the match between the teacher’s curated library and the learner’s current representational state. These predictions are specific enough to be testable, but may require longitudinal studies, cross-cutting developmental participant samples, and expertise research that this thesis does not provide.

The library is never directly observed. Every empirical claim in this thesis is licensed by fitting a library-structured generative model to behavior and reading off latent quantities—entries, masteries, utilities, or listener types—from the posterior. Behavior alone underdetermines this latent structure: a sufficiently flexible non-library account, such as a wide neural predictor, flat exemplar memory, or end-to-end policy, could in principle reproduce path dependence, transfer asymmetries, and explanation preferences without containing reusable library entries (Hayes, 1989; McCloskey, 1991)⁷. A stronger empirical program would seek signatures that non-compositional accounts struggle to match: combinatorial generalization to held-out compositions (Lake and Baroni, 2018), systematic trial-by-trial hypothesis biases (Rubino et al., submitted), neural evidence for reusable subroutine activation (Dehaene et al., 2022), or interventions that selectively disrupt one inferred library entry and predict downstream failures. The current evidence therefore supports the library framework, but does not yet provide the discriminating tests needed to rule out flexible non-compositional alternatives.

7: The issue is not that non-library models cannot fit the data, but that good fit alone does not distinguish whether the model has recovered reusable structure or merely approximated the input-output mapping.

9.3 The future directions

Section 9.2 listed what the thesis did not establish. Each limitation points to a concrete future direction, and this section lays out those directions as a research agenda.

9.3.1 Three axes of extension

Each axis names a direction along which the framework needs to extend beyond what the current empirical chapters have demonstrated.

Scale. Chapter 4 and Chapter 7 demonstrated lifecycle operations on designed sequential data or grid worlds where library entries can be cleanly enumerated and the environment is hand-built around them. Naturalistic libraries—the kind humans carry across tasks—have entries proposed in natural language, grounded in open-world environments, and shaped through interaction with other agents (most often humans). The remaining work required for scaling largely consists of engineering Curator architecture for more complex settings. This could include using LLM-augmented Curators in richer environments (e.g., Minecraft-class open worlds, code, or web-browsing), applying variational lower bounds on representational empowerment to adapt it for continuous and

high-dimensional libraries, and developing tractable approximations of listener-relative utility under open-ended task spaces⁸. Human-AI collaboration is a natural test case for this scaled setting. There, the library is not shaped by a synthetic curriculum alone, but by an interacting human partner who proposes goals, corrects abstractions, and evaluates whether the system's representations are useful. Scaling is therefore also the problem of making library construction, curation, and communication work under open-ended social feedback.

Duration. The thesis studies the lifecycle within sessions or short cycles, treating the learner as approximately stationary across them. But library reorganization also unfolds over developmental and cultural timescales, where the relevant mechanism is often not an isolated learner but a dyad or community. For example, a child's library may be scaffolded by caregiver interaction over many years (Gopnik, 2023), while a scientific community may reshape its representational library through shared notation, instruments, and explanatory norms. Extending the framework along this axis requires longitudinal library inference. Relevant data would pair caregiver interventions with learner responses, compare experts and novices in settings where expertise was socially transmitted, or trace historical cases where a notational or scientific library changed through collective use. Chapter 5 offers a small prototype: educational logs treat curriculum as an external structuring force on learner libraries, but at a much shorter timescale and with a thinner model of social scaffolding than the duration axis ultimately requires.

Multiplicity. Chapter 8 treats explanation as largely one-directional: a knowledgeable speaker helps a listener with a partial library. The multiplicity axis extends this to settings where representational libraries are jointly constructed rather than simply transmitted. This includes collaborative problem solving, where agents with non-overlapping libraries build a shared vocabulary through interaction, and community-level learning, where scientific, technical, or cultural vocabularies emerge from repeated coordination across many agents (Clark and Brennan, 1991; Hutchins, 1995; Tomasello et al., 1999). This axis also brings into view questions that the thesis leaves mostly untouched. Shared libraries are not only shaped by usefulness, but also by social position: whose abstractions are adopted, whose are ignored, which vocabularies coexist, and which become standardized as the community's default. In this sense, "representational politics" refers to the social process by which some library entries become canonical while others are marginalized. Extending the framework to multiplicity therefore requires models of negotiation, authority, expertise, and institutional uptake, not only models of speaker-listener alignment.

Next I outline the future directions. The first direction is an integrative pilot that tests the coupling claim directly. The remaining three sit at intersections of the axes and instantiate the framework on problems specific fields already care about.

9.3.2 End-to-end test of the coupling claim

Framing. A direct test of the lifecycle would ask whether construction, curation, and communication interact, rather than merely improve performance when added one after another. This thesis does not yet provide such a test: each empirical chapter studies one stage, or at most two in interaction. The proposed experiment therefore targets the framework itself. If the three stages do not produce interaction effects, the lifecycle

8: Here "continuous" means that library entries are no longer discrete symbolic objects that can be enumerated exactly; they may be embeddings, policies, code fragments, natural-language skills, or latent factors whose boundaries must be inferred.

remains a useful organization of mechanisms, but not yet a distinct claim about coupled representational dynamics.

Setup. A suitable testbed would be Crafter (Hafner, 2021), whose 22 achievements form a prerequisite structure over reusable skills. Each achievement could be represented as a typed library entry $e = \langle \text{pre}, \text{post}, \pi \rangle$, with PDDL-style preconditions and postconditions over inventory states and a learned option policy π . This shared entry format matters because it lets construction, curation, and communication act on the same library $\mathcal{L}$, rather than on separate proxies. The design would be a $2 \times 2 \times 2$ factorial: a speaker agent receives construction, curation, and communication at high or low settings, then teaches a recipient agent with an independent learning history. Transfer is measured by the recipient’s performance on held-out achievements.

Technical instantiation. The three factors could be implemented as follows. *Construction* contrasts a hierarchical-adaptor-grammar-style two-level schedule (per-phase local libraries consolidated into a global library between phases, achievement targets ordered by partial-information-decomposition synergy on the achievement co-occurrence statistics, following Chapter 4) against a uniform random target schedule. *Curation* contrasts MCMC proposals over Ω accepted under an InfoNCE⁹ lower bound on the channel capacity $I(\Omega; \mathcal{L}')$ estimated from K rollouts per candidate (the variational estimator anticipated in Section 9.3.1, scale axis) against pass-through retention with no merge or delete. *Communication* contrasts greedy maximization of recipient marginal solve rate per description-length unit—using the utility of Chapter 8 instantiated with the recipient’s planner—against uniform sampling of operators up to the same description-length budget. The recipient is a hierarchical agent with its own option library from an independent construction phase. For each operator the teacher demonstrates, the recipient computes a Bayesian posterior over whether to adopt it conditioned on the teaching trajectory (Chapter 5), and runs a verification rollout that checks whether the operator’s postcondition predicate fires in the recipient’s own environment; operators that fail verification are rejected even when the posterior favors adoption.

Predictions. A construction⁺-curation[−] library should grow roughly linearly in achievements covered and exhibit a *negative* slope of recipient solve rate against library size at fixed inference compute, because operator-match search dominates the inference budget—a signature additive accounts cannot produce. A curation⁺-communication[−] teaching trajectory should under-sample the low-frequency operators that bridge prerequisite chains, so recipient failures concentrate on held-out achievements whose minimum-cost solution path passes through operators with high betweenness centrality in the DAG. Construction[−]-communication⁺ should produce a teacher who selects high-utility operators from a low-empowerment library, with the recipient’s posterior concentrating on entries whose verification step subsequently fails. If the three-way interaction is null and these structural dissociations fail to appear, the coupling story is wrong: the lifecycle reframing collapses to a useful taxonomy without a load-bearing scientific claim, and the framework retreats to the more modest statement of Section 9.2.1.

What it advances. This experiment would test what the thesis currently assumes but does not demonstrate: that the lifecycle stages are mutually enabling. The existing chapters show that construction, curation, and communication can each be formalized and can each matter. The pilot would ask whether their combination produces transfer patterns that

9: InfoNCE would be used here only as a tractable estimator of mutual information, not as a new theoretical commitment. The target remains representation empowerment; the bound is the engineering approximation.

cannot be explained by any single stage alone. A null three-way interaction would weaken the central lifecycle claim and reduce the framework to a taxonomy of useful mechanisms; a positive interaction would turn the lifecycle from an organizing frame into a directly tested account of representational change.

9.3.3 Naturalistic curation and human-AI collaboration

Framing. The Curator of Chapter 7 was demonstrated on a designed grid world with enumerable libraries and a synthetic task distribution; Section 9.2.2 flagged that the empowerment criterion has not been tested on libraries with naturalistic structure or in environments shaped by interaction with humans. The open question is whether the Curator architecture extends to LLM-backed agents in open-world environments and, beyond that, whether the same machinery supports human-AI co-curation when the human is in one of the agent slots.

Setup. The testbed would be an open-world game in the Crafter lineage but naturalistic in scale: Minecraft via MineDojo (Fan et al., 2022), in which prior work has demonstrated LLM-driven skill libraries (Wang et al., 2023) but without a representational-empowerment-driven Curator. A library entry retains the typed form $\langle \text{pre}, \text{post}, \pi \rangle$ of Direction 1, with π a code-synthesized policy produced by an LLM from a natural-language goal description, and pre- and post-condition predicates LLM-extracted over inventory and world state. The base architecture is held fixed; what varies is the curation condition (empowerment-driven Curator vs. pass-through retention of all LLM-proposed skills) and the agent composition (AI-only vs. a human-AI dyad sharing one Curator).

Technical instantiation. The modification operator set can be $\Omega = \{\text{ADD}, \text{MERGE}, \text{REFACTOR}, \text{DELETE}\}$: ADD has the LLM propose a new code-synthesized skill from a goal description, MERGE performs LLM-mediated semantic composition of two skills, REFACTOR re-synthesizes the policy code with predicate re-extraction, and DELETE unmounts an entry. Representational empowerment can be estimated by an InfoNCE lower bound on $I(\Omega; \mathcal{L}')$ from rollouts in MineDojo, instantiating the variational estimator the scale axis of Section 9.3.1 promised at naturalistic scale. In the human-AI dyad condition, both the human and the AI may propose entries; the Curator decides admission on representational empowerment alone, with no privileged authority for either party—the social-product framing of the multiplicity axis applied at the dyadic scale.

Predictions. Three signatures can distinguish the framework’s predictions from baselines. First, an empowerment-curated AI library should solve a held-out task suite at higher rates than an LLM-only library of comparable token cost, replicating the Direction 1 finding at naturalistic scale and ruling out a coverage-only explanation. Second, the human-AI dyadic library should outperform either party’s solo library on tasks that require operators neither party would have proposed alone, with the gain concentrated on tasks requiring operators that bridge the two parties’ initial expertise. Third, an audit by independent human raters should rate empowerment-curated entries as more reusable than entries from a library-of-everything baseline, with reusability ratings tracking $I(\Omega; \mathcal{L}')$ rather than entry count or coverage breadth. The falsifier is sharp on each axis: a pass-through baseline that retains every LLM-proposed skill matches the Curator on transfer, OR human ratings track library size rather than empowerment, OR the dyad’s library reduces to a permissive union of solo libraries rather than producing operators that bridge them.

What it advances. This direction can discharge the formalization limitation in Section 9.2.2 by demonstrating representational empowerment on libraries that are continuous in entry content and proposed in natural language, and adds a use case the thesis names but does not test: human-AI co-curation as the deployment-relevant instance of the framework. It also makes the multiplicity framing concrete at the dyadic scale, providing an empirical handle on the “library as social product” claim before the full collaborative setting of Direction 4.

9.3.4 Library inference for developmental trajectories

Framing. Developmental theories describe representational reorganization qualitatively—children move through stages, U-shaped trajectories appear and resolve, abstractions emerge from instances—but do not predict the timing or content of those transitions from first principles, in part because the dyadic mechanism (caregiver intervention paired with learner response) is rarely modeled jointly with the learner’s representational state. Section 9.2.3 flagged that the thesis’s human experiments are short and online, leaving the developmental predictions of the framework untested on the timescales that matter.

Setup. The testbed is two longitudinal corpora in domains where the developmental literature has stable theoretical scaffolding: English past-tense acquisition drawn from CHILDES (MacWhinney, 2000; Marcus et al., 1992), whose canonical U-shape on irregular verbs is one of the most-studied trajectories in cognitive development and which pairs the child’s productions with the caregiver’s verbal input, and number-conservation longitudinal data following Siegler (2004), in which the caregiver-equivalent is the experimenter’s task structure rather than verbal input. A library entry is a typed rule $\langle \text{trigger}, \text{output}, m \rangle$: a context predicate over the input, a probabilistic output procedure, and a mastery score, executed within the same probabilistic-program formalism used by Chapter 5. The library is held responsible for the child’s behavior at each time window, conditioned on the input window the caregiver provided; what varies is the inferred library’s structure, and what is predicted is the timing of structural transitions.

Technical instantiation. The inference model is a hierarchical Bayesian extension of PSIKT in which the library itself is non-stationary: a Beta-Bernoulli process prior over rule presence allows entries to be born, refined, or pruned across age windows, with per-window mastery sampled conditional on rule presence. Per-entry representational empowerment is computed as the leave-one-out drop in $I(\Omega; \mathcal{L}')$ when the entry is deleted, giving a per-rule sensitivity score rather than a library-level scalar. The U-shape prediction is operational: the trough timing for an irregular verb is predicted as the age window in which the empowerment of the candidate library containing the general ADD-ED rule first exceeds that of the lexical-only library, even though task accuracy on irregulars temporarily declines—a coupling between empowerment-gradient and accuracy-dip that is computed per child from their own response history. The Karmiloff-Smith (Karmiloff-Smith, 1994) prediction is operational the same way: redescription is predicted to target rules whose leave-one-out empowerment is low and whose successor rules have higher empowerment, with the redescription-phase ordering ($E1 \rightarrow E2 \rightarrow E3$) recovered from the cross-entry empowerment ranking at the redescription window.

Predictions. Quantitative success has three signatures. First, a per-child match between predicted and observed U-trough windows on irreg-

ular verbs, with the alignment measured by the correlation between predicted and observed onset and offset ages across the cohort; the prediction is that this correlation exceeds what age and vocabulary-size baselines achieve. Second, in the conservation data, a within-child dissociation between rules that are redescribed (low leave-one-out empowerment, high-empowerment successor available) and rules that are merely strengthened, with the framework predicting which rules fall into which category before the redescription is observed. Third, a structural prediction the dominant exemplar-memory and connectionist accounts cannot match: the recovery from the U should be faster on verbs whose neighbors in phonological space were also overregularized, because shared empowerment gains transfer; this is a non-monotonic effect of within-cohort co-occurrence that requires a library-structured explanation. The falsifier is a flexible non-library account—a wide neural predictor or a flat exemplar memory—that matches all three signatures within noise; the thesis-level limitation in [Section 9.2.3](#) is precisely this concern, and this direction either discharges it on developmental data or concedes it.

What it advances. This direction extends the lifecycle from session-scale to year-scale, validating the framework on the timescale at which library reorganization is theoretically most consequential. It also gives developmental theories that are currently qualitative—representational redescription, U-shaped learning, stage-like transitions—a quantitative criterion (per-entry empowerment) that predicts *which* entries reorganize and *when*, rather than describing the reorganization after the fact. Success would convert library inference from a method demonstrated on educational logs into a tool the developmental field can deploy.

9.3.5 Collaborative library construction

Framing. The communication chapters of this thesis treat explanation as uni-directional, with a knowledgeable speaker and a partial listener, but the most consequential cases of library transmission—between collaborators, between teachers and learners across many cycles, between communities whose vocabulary emerges from interaction—are bidirectional (Wu et al., 2024). [Section 9.3.1](#) flagged this as the multiplicity axis; the open question is whether the framework’s listener-relative utility extends to a setting where each agent is simultaneously listener and speaker, with no fixed authority over what the shared library should contain. This direction sits at the multiplicity $\times$ all-fields intersection and is the long-term bet—the most ambitious and the least immediately tractable.

Setup. The testbed is collaborative Crafter, a two-agent extension of the Direction 1 environment in which neither agent’s initial library covers the full prerequisite DAG and a held-out joint task suite is solvable only by compositions whose constituent operators are split across the two libraries. Each agent starts with a typed library $\mathcal{L}_A, \mathcal{L}_B$ of $\langle \text{pre}, \text{post}, \pi \rangle$ entries (Direction 1’s format), and a shared library $\mathcal{L}_S$ that is initially empty and grows through interaction. The interaction protocol is alternating dyadic exchange: on each turn, an agent may execute, demonstrate, propose an entry for $\mathcal{L}_S$, or request that the partner demonstrate; what varies across conditions is the asymmetry between the initial libraries and the utility model that drives proposal selection.

Technical instantiation. Each agent maintains a posterior over the partner’s library and updates it from observed demonstrations using the same

10: The recursion captures mutual modeling: each agent chooses what to communicate based on what it believes the partner believes, and this nesting is truncated to keep inference finite.

PSIKT-style library inference of Direction 1, now run in both directions simultaneously. The bidirectional utility for agent A proposing entry e is the expected gain in B 's posterior solve rate on the joint task suite under B 's current model of A , which itself depends on B 's model of A 's utility—formally a fixed-point computation over an RSA-style (Frank and Goodman, 2012) recursion truncated at fixed depth, with each agent's policy a best response to its model of the partner's policy¹⁰. Curation operates on the shared library: entries are admitted to $\mathcal{L}_S$ when their joint representational empowerment $I(\Omega; \mathcal{L}'_S)$, evaluated on the task distribution both agents face, exceeds a threshold; MERGE acts when both agents have proposed near-duplicates, and naming conflicts are resolved by retaining the entry with higher leave-one-out empowerment. The dependent measures are joint solve rate on held-out tasks, the composition of the converged $\mathcal{L}_S$, and a dominance index measuring how much of $\mathcal{L}_S$ traces back to one agent versus the other.

Predictions. Three structural predictions distinguish the bidirectional model from a one-directional baseline (one agent fixed as teacher). First, when the two initial libraries are non-overlapping but comparable in size, the converged $\mathcal{L}_S$ should have higher collective representational empowerment than either initial library, with the gain exceeding what additive merging of the two libraries would produce—a coupling effect on the multiplicity axis paralleling the coupling claim of Direction 1. Second, when one library has higher empowerment than the other regardless of size, the converged $\mathcal{L}_S$ should be dominated by the higher-empowerment library; the framework predicts that representational dominance is set by empowerment, not by size or initiative, and the prediction is testable by manipulating the two independently in matched dyads. Third, when the depth of the RSA-style recursion is artificially capped at zero (each agent treats the partner as a fixed encoder), the joint solve rate should fall below the bidirectional condition by a margin that grows with library asymmetry—isolating the bidirectional inference as the load-bearing component rather than the mere fact of two-way interaction. The falsifier is a result in which equal-empowerment dyads still produce a single-dominant outcome regardless of recursion depth: the bidirectional utility model is then wrong, and the framework collapses to one-directional teaching with role assignment.

What it advances. This direction extends the framework along the multiplicity axis to settings the thesis identified but did not address: representational politics, the conditions under which competing libraries coexist, and the role of authority in determining shared canon. It also opens the framework's communication machinery to applications in human–AI collaboration, team science, and the design of shared scientific knowledge bases, where the one-directional teacher–learner abstraction has long been recognized as the wrong default. Success would convert the lifecycle from a single-agent process into a model of how cognitive communities construct their shared representational infrastructure.

9.3.6 Coevolution and the unit of analysis

The four directions above are future projects for extending the framework. This final conjecture is different: it concerns the future trajectory of the phenomenon itself. In domains where LLM use becomes pervasive, the individual human library may no longer be the right unit of analysis for representational dynamics. The measurable and intervenable object

may instead be the joint human-AI library, co-constructed and co-curated through interaction.

Distributed-cognition and extended-mind accounts (Clark and Chalmers, 1998; Hutchins, 1995) have long held that the cognitive unit appropriate to many tasks is not the individual but the individual-plus-environment; what LLMs change is the time constant. The coupling can occur at the timescale of seconds, in both directions, and across many symbolic domains—writing, coding, planning, reasoning, and education. In such settings, treating the human library as fixed and internal may miss the very dynamics that determine what the person can think, write, or learn next.

The conjecture is that the construct-curate-communicate decomposition will still apply, but to dyadic collaboration rather than individual cognition. Construction occurs as human and model jointly introduce new abstractions; curation occurs as they retain, revise, or discard them across interaction; communication occurs as each side adapts its representation to what the other can use. The lifecycle therefore moves from a within-person process to a coevolutionary process over a coupled human-AI library.

The conjecture has two components, each independently falsifiable. First, individual cognition may remain separable: people may use LLMs heavily while their internal libraries remain recoverable by traditional individual-level methods. In that case, the framework is correctly scoped to the individual. Second, the unit of analysis may shift to the dyad, but the library lifecycle may not be the right formal apparatus for describing it. The joint dynamics might require a different object than libraries, or different operations than construction, curation, and communication. Either outcome would be informative: the first would bound the framework; the second would preserve the coevolutionary insight while forcing a new decomposition.

9.4 Closing reflection

This thesis began from the historic wager of cognitive science: that thinking can be explained as an organized form of information processing. That wager produced two enduring questions: the first asks what can be computed, and the second asks what counts as a rule, representation, or procedure in the first place. The claim of this thesis is that these questions meet in the problem of representational dynamics. Computation must be directed not only at the world, but also at the agent's own computational substrate: the representations it constructs, evaluates, and communicates.

The generative library is our answer to the question of what theoretical object lives at that junction. A library is computational because its entries can be composed, executed, and tested. It is pragmatic because those entries derive their value from use: from what they make easier to infer, learn, explain, or do. The lifecycle of a library—construction, curation, and communication—describes how an agent manages this dual demand: building representations precise enough to compute with, yet flexible enough to revise when future tasks require it.

The thesis has not resolved the representational dynamics gap. Doing so would require a general theory of how arbitrary representational formats can be constructed, evaluated, and transmitted under arbitrary resource constraints. What the thesis has shown is more modest: the gap is real,

it is structured, and it becomes empirically tractable once the library is taken as the object of analysis.

The framework yields specific predictions: construction should be path-dependent; curation should depend on library-internal rather than merely environment-facing objectives; and communication should be listener-dependent because explanations act on another agent's library. The empirical chapters provide initial support for these predictions across multiple domains, methods, and scales.

Turing showed that computation is universal: a single machine can simulate any computable function, given the right program. But he also showed, through his analysis of the universal machine, that programs are themselves data—objects that can be stored, copied, modified, and composed. This insight—that the substrate of computation is itself computable—is the seed of everything this thesis has explored. A library is a collection of programs-as-data: computational tools that the agent can inspect, reorganize, and share. The lifecycle of a library is the process by which a finite mind manages this self-referential capacity: building the tools it will think with, evaluating whether they are worth keeping, and teaching them to others who will build further.

Intelligence, on this view, is not the execution of a fixed program. It is the ongoing, resource-bounded, socially embedded management of the programs by which future thought becomes possible.

Part VI.

Appendix

Additional Material for Chapter 4

A.

The appendix for Chapter 4 is organized as follows:

- ▶ **Section A.1** describes the five baseline models (RLE, Chunking, PCFG, AG, HAG) in full detail, specifying the operator sets, library mechanisms, and how each differs from HAG in representational assumptions.
- ▶ **Section A.2** reports melodic sequence statistics for the training and held-out sets, and summarizes behavioral performance across all three experimental phases (error rates and reaction times).
- ▶ **Section A.3** documents the full parameter grids used in the simulation study and the plausible observer analysis, covering the memory-rate parameter β , search and backtracking budgets N_s and N_b , Pitman-Yor reuse parameters, and library size limits.
- ▶ **Section A.4** provides supplemental simulation results, including generalization RD curves on held-out sequences, RD performance as a function of training set size, and the effect of the memory-rate parameter β on compositional reconstruction length.
- ▶ **Section A.5** provides supplemental behavioral analyses, including detailed definitions and detection procedures for all four error categories, the full mixed-effects regression for choice prediction, and additional reaction-time analyses for the chunking observer.
- ▶ **Section A.6** reports the model fitting procedure, the full parameter grids, individual-level BIC and log-likelihood scores, Phase 3 continuation predictions, parameter recovery analyses, and the relationship between fitted parameters and individual behavioral outcomes.

A.1 Method details	187
A.2 Data details	193
A.3 Paradigm and spans of three observers	195
A.4 Supplemental simulation results	196
A.5 Supplemental behavioral analyses	199
A.6 Model fitting details . . .	203

A.1 Method details

A.1.1 Alternative models

We consider five models that make progressively richer assumptions about structure learning. All models use the same typed CL representation and the same resource-constrained objective; they differ in which primitives are available and whether a library is maintained.

- ▶ **RLE (Run-Length Encoding):** Programs are built from single operator $\text{rep}(n, c)$ which encodes repeating note n for c steps. After each sequence, the global library $\mathcal{L}_{\text{glob}}$ is updated by adding any $\text{rep}(n, c)$ subprograms selected in the program inferred for that sequence.
- ▶ **Chunking:** Programs are built from a single operator $\text{chunk}(n)$, which concatenates stored chunks (Wu et al., 2023). The model assumes that reusable structure takes the form of contiguous memorized subsequences, rather than compositional transformations, and is therefore a chunking account. Similar to RLE, the learning operates over a single global library $\mathcal{L}_{\text{glob}}$, which can add contiguous subsequences as new chunks whenever doing so decreases the total description length.

1: This is the “rich get richer” property of the Pitman-Yor process: each time a subprogram is reused, its count n_{ik} increases, raising its predictive probability for the next subprogram slot. Early commitments therefore compound over the sequence, making the model path-dependent within each melody as well as across melodies.

2: The nesting is strict: $\text{RLE} \subset \text{Chunking} \subset \text{AG} \subset \text{HAG}$. Each step adds one capacity: RLE adds repetition; Chunking adds verbatim memorization; AG adds compositional operators and cross-sequence reuse; HAG adds a local within-sequence library on top. Each model is recovered from HAG by appropriate limiting parameter values.

- PCFG (Probabilistic Context-Free Grammars): Programs are built from pre-defined operators: up/down (transposition by a step count), range (generate a stepwise sequence from a start note and step size), rep (repeat a note or sequence), rev (reverse order of a sequence), and chunk (verbatim memorization). The grammar and its rule probabilities are fixed across sequences, with operators and combinators assigned equal prior probability over well-typed compositions. Each sequence is therefore parsed independently under the same prior. Unlike the library-based models, PCFG has no mechanism for updating its representations across sequences.
- AG (Adaptor Grammar): Programs are built from the same operators as with PCFGs. This model augments the PCFG with a capacity for building a shared library of programs. AG places a nonparametric prior over reusable programs via a Pitman-Yor process (PYP; Johnson et al., 2006; Teh, 2006):

$$G \sim \text{PYP}(d_{\text{glob}}, \alpha_{\text{glob}}, H), \quad (\text{A.1})$$

where the base distribution H is given by the PCFG. After each melody, the posterior over G is updated: programs that recur across melodies become more probable—and therefore cheaper to reuse—in subsequent inductions.¹ The concentration parameter α_{glob} controls the rate of innovation, while the discount parameter d_{glob} controls the diversity of reuse.

- HAG (Hierarchical Adaptor Grammar): Programs are from the same operators as the PCFG and AG. HAG replaces the single global adaptor in AGs with a *hierarchical* Pitman-Yor process (HPYP) that induces two nested libraries: a sequence-specific *local* library $\mathcal{L}_{\text{loc}}$ and a shared *global* library $\mathcal{L}_{\text{glob}}$. The global library stores reusable programs that have been useful across sequences, whereas the local library stores reusable programs that are especially useful within the current sequence (see Equation (A.11)). Concretely, the PCFG provides the base structural prior over well-typed combinatory-logic expressions, while the HPYP determines how inferred programs are reused and updated across the two libraries (Teh, 2006).

The five models form a nested hierarchy. RLE and Chunking are special cases of AG with restricted primitive sets. When α_{glob} is very large (minimizing reuse), AG converges to PCFG. When α_{loc} is very large (suppressing local reuse), HAG reduces to AG. This nesting enables principled model comparison, with each model adding one degree of representational capacity.²

A.1.2 Bayesian program and library induction

Formally, Bayesian program induction (Correa et al., 2023; Goodman et al., 2008; Lake et al., 2015; Piantadosi et al., 2012; Rule et al., 2018) offers a probabilistic realization of the *Language of Thought* hypothesis (Fodor, 1975). In our setting, inference is conditioned on hierarchical libraries that bias hypotheses toward previously learned structure. Given an observation $x \in \mathcal{X}$, a candidate program $\pi \in \mathcal{P}$, a global library $\mathcal{L}_{\text{glob}}$, and a local library $\mathcal{L}_{\text{loc}}$, the posterior can be defined as:

$$p(\pi \mid x, \mathcal{L}_{\text{glob}}, \mathcal{L}_{\text{loc}}) \propto p(x \mid \pi) p(\pi \mid \mathcal{L}_{\text{glob}}, \mathcal{L}_{\text{loc}}). \quad (\text{A.2})$$

The likelihood $p(x \mid \pi)$ measures how well the program reconstructs the observation, whereas the library-conditioned prior $p(\pi \mid \mathcal{L}_{\text{glob}}, \mathcal{L}_{\text{loc}})$

favors programs that reuse previously learned structure.

We represent programs in typed combinatory logic (CL) (Liang et al., 2010; Schönfinkel, 1924), following Section 4.3.1. Base types distinguish semantically different values (notes t_n , counts t_c , time indices t_m); function types specify valid input–output relations, e.g. $\text{up} : t_n \times t_c \rightarrow t_n$.

Two resource constraints: memory and computation. Bayesian inference over a compositional hypothesis space of candidate programs is, in principle, unbounded and therefore intractable. Because our goal is to model human learners as resource-rational agents rather than unconstrained ideal observers, we impose two resource constraints on program inference (Equation (A.6)). The first is a *memory* constraint, R_{mem} (Equation (4.1)), which limits the complexity of retained programs through their description length. This constraint is implemented as a preference, governed by β , for shorter and therefore more compressible programs. The second is a *computational* constraint, R_{comp} (Equation (4.1)), which limits the amount of processing the model can perform when proposing candidate programs.

We formalize the *memory* constraint with rate–distortion theory (RDT) adapted to program induction (Bates and Jacobs, 2020; Nagy et al., 2020; Sims, 2016; Zhou et al., 2024b). Let $g : \mathcal{P} \rightarrow \mathcal{X}$ denote the interpreter that maps a candidate program to its reconstruction. For a target x and a candidate program π , execution yields a reconstruction $\hat{x}_\pi = g(\pi)$, and reconstruction quality is measured by a task distortion $d(x, \hat{x}_\pi)$. Critically, the relevant description length is the complexity of the *representation that is retained*. In our setting, the retained object is the reconstruction $\hat{x}$. To connect description length to our probabilistic model $p(\pi)$ over programs, we assume programs are encoded with a (near) Shannon-optimal prefix code induced by this prior. This assigns each program π a codelength $\ell(\pi) := \lceil -\log p(\pi) \rceil$ (Grünwald, 2007; Grünwald, Vitányi, et al., 2008), which is within an additive constant of $-\log p(\pi)$ (at most 1 bit, plus any fixed coding overhead).³ We then define the description length of a reconstruction $\hat{x}$ as the length of the shortest code among all programs whose execution yields that reconstruction:

$$L(\hat{x}) := \min_{\pi: g(\pi)=\hat{x}} \ell(\pi) \approx \min_{\pi: g(\pi)=\hat{x}} (-\log p(\pi)). \quad (\text{A.3})$$

The corresponding RD objective is then

$$D(R_{\text{mem}}) = \inf_{x \in \mathcal{X}} d(x, \hat{x}) \quad \text{s.t.} \quad L(\hat{x}) \leq R_{\text{mem}}, \quad (\text{A.4})$$

or equivalently, an optimization over programs while measuring rate at the level of the retained reconstruction. In practice, we optimize over candidate programs directly and use $-\log p(\pi)$ for the selected program as an approximation to this optimal reconstruction-level description length.

We next model *computation* as a stochastic, resource-bounded search process A_{N_s, N_b} over the program space $\mathcal{P}$. The computational constraint R_{comp} is operationalized by two limits: a search budget N_s , which caps the number of candidate programs evaluated, and a backtracking budget N_b , which caps the number of previously selected programs that can be revised in light of new observations. Let $q_{N_s, N_b}(\pi | x)$ denote the resulting distribution over candidate programs, which captures both which programs are reachable and with which probability they are found.

3: A Shannon-optimal code assigns codeword length $\lceil -\log p(\pi) \rceil$ to each program π . This is optimal in the sense that no uniquely decodable code can have shorter expected length (Cover, 1999). Using $-\log p(\pi)$ as a proxy therefore measures description length in natural units and connects the program prior directly to an information-theoretic rate.

The computationally bounded objective is then the expected distortion under this resource-limited search:

$$D_{\text{comp}}(R_{\text{mem}}; N_s, N_b) = \mathbb{E}_{\pi \sim q_{N_s, N_b}(\cdot | x)} [d(x, \hat{x}_\pi)] \quad (\text{A.5})$$

s.t. $L(\hat{x}_\pi) \leq R_{\text{mem}}.$

Since A_{N_s, N_b} is stochastic (i.e., sampling-based), q_{N_s, N_b} explicitly captures discoverability under limited computation.

The corresponding Lagrangian form of Equation (A.5) is:

$$\mathcal{J}_{N_s, N_b}(x) = \mathbb{E}_{\pi \sim q_{N_s, N_b}(\cdot | x)} [d(x, \hat{x}_\pi) + \beta L(\hat{x}_\pi)], \quad (\text{A.6})$$

where β controls the trade-off between accuracy and complexity, whereas the search budget N_s and backtracking budget N_b control exploration and revision during inference.

Inference over hierarchical libraries. The previous section defined program induction for a single melodic pattern under memory and computational constraints. We now generalize the formulation to the present setting, in which each data point is an entire melodic sequence rather than a single pattern. For each sequence $X^{(i)}$, the model infers a sequence-level program $\Pi^{(i)}$ built from reusable subprograms $\pi_j^{(i)}$. Each subprogram generates a local reconstruction and is evaluated under the same rate-distortion (RD) trade-off introduced above. Sequence-level inference then operates over the ordered concatenation of these subprograms, determining both which subprograms are reused and how they are assembled to reconstruct the full sequence.

Formally, each observation is a sequence $X^{(i)} = [x_1^{(i)}, \dots, x_T^{(i)}] \in \mathcal{X}$, explained by a sequence-level program

$$\Pi^{(i)} = \pi_1^{(i)} \oplus \pi_2^{(i)} \oplus \dots \oplus \pi_{M^{(i)}}^{(i)}, \quad (\text{A.7})$$

where each $\pi_j^{(i)} \in \mathcal{P}$ is a subprogram, $\oplus$ denotes concatenation, and each subprogram generates a contiguous subsequence under the interpreter g , such that $\hat{X}_j^{(i)} = g(\pi_j^{(i)})$. The full reconstruction is therefore

$$\hat{X}^{(i)} = g(\Pi^{(i)}) = \hat{X}_1^{(i)} \oplus \hat{X}_2^{(i)} \oplus \dots \oplus \hat{X}_{M^{(i)}}^{(i)}. \quad (\text{A.8})$$

The posterior over sequence-level programs $\Pi^{(i)}$ is

$$p(\Pi^{(i)} | X^{(i)}) \propto p(X^{(i)} | \Pi^{(i)}) p(\Pi^{(i)} | \mathcal{L}_{\text{glob}}, \mathcal{L}_{\text{loc}}), \quad (\text{A.9})$$

whereas, in practice, the inference is carried out locally over candidate subprograms that explain successive subsequences. We next specify the three components of this inference procedure: i) a hierarchical prior over reusable subprograms, ii) a sequence likelihood derived from the subprogram objective, and iii) an approximate online inference algorithm with bounded backtracking.

Hierarchical prior over reusable subprograms. The basic unit of reuse in the model is the subprogram π . We therefore place a prior over subprogram expansions that allows a candidate subprogram to be generated either by composing operators from the underlying typed grammar or by reusing previously inferred subprograms stored in the local and global libraries.

As the base structural prior, we use a uniform distribution over well-typed CL trees (Liang et al., 2010).⁴ For a subprogram $\pi = [r, x, y]$ with type t , combinator r and subtrees x, y , the prior factorizes as

$$\log p(\pi | t) = \log p(x | t) + \log p(y | t) + \log p(r), \quad (\text{A.10})$$

with operators and combinators equiprobable, conditioned on their type.

To model reuse, we place a hierarchical Pitman-Yor process (HPYP) prior over subprogram choices (Teh, 2006). Let H denote the base distribution over typed subprogram expansions defined by the typed grammar above. At the global level, we define a distribution over reusable subprograms, $G_0 \sim \text{PYP}(d_{\text{glob}}, \alpha_{\text{glob}}, H)$, which captures domain-general reuse tendencies across sequences. For each sequence $X^{(i)}$, we then define a sequence-specific distribution over subprograms, $G_i | G_0 \sim \text{PYP}(d_{\text{loc}}, \alpha_{\text{loc}}, G_0)$ which biases inference within that sequence toward reusing its own previously inferred subprograms while still permitting reuse of subprograms from the global distribution G_0 .

Under this hierarchy, we now define the predictive probability of a subprogram in a sequence. For each sequence $X^{(i)}$, $\pi_{<j}^{(i)}$ denotes the subprograms inferred before position j , which defines the current local library $\mathcal{L}_{\text{loc}}$: its distinct elements are ϕ_k , and their counts determine the sequence-specific predictive distribution implied by G_i . In the same spirit, $\pi^{(<i)}$ denotes the subprograms inferred before sequence $X^{(i)}$ and constitute the global library $\mathcal{L}_{\text{glob}}$, and G_0 denotes the corresponding global predictive distribution. Thus, the predictive probability $p(\pi | \pi_{<j}^{(i)})$ of the next subprogram is

$$p(\pi | \pi_{<j}^{(i)}) = \underbrace{\sum_{k=1}^{K_i} \frac{n_{ik} - d_{\text{loc}}}{\alpha_{\text{loc}} + n_{i\cdot}} \delta_{\phi_k}}_{\text{reuse from local library}} + \underbrace{\frac{\alpha_{\text{loc}} + d_{\text{loc}} K_i}{\alpha_{\text{loc}} + n_{i\cdot}} G_0}_{\text{back off to global distribution}}, \quad (\text{A.11})$$

where n_{ik} is the number of times subprogram ϕ_k has been used in sequence i , K_i is the number of distinct subprograms used in that sequence, and $n_{i\cdot} = \sum_{k=1}^{K_i} n_{ik}$ is the total number of subprogram instances assigned so far in sequence i .

The predictive distribution is also governed by four free parameters in HPYP. The concentration parameters α_{loc} and α_{glob} control the rate of innovation at each library. Larger values increase the probability of introducing a new subprogram rather than reusing an existing one, whereas smaller values favor reuse. When $\alpha_{\text{loc}} > \alpha_{\text{glob}}$, sequence-specific inference is relatively more likely to introduce new local subprograms than to rely on globally shared ones. The discount parameters d_{loc} and d_{glob} control the diversity of reuse: $d \approx 0$ yields “rich-get-richer” dynamics allowing a few subprograms to dominate, whereas $d > 0$ flattens the distribution and encourages a heavier-tailed mix of common and rare subprograms.⁵

Tradeoff between distortion and rate as likelihood. For a candidate subprogram $\pi_j^{(i)}$ assigned to subsequence $X_j^{(i)}$, we use the Lagrangian form of the RD objective (Equation (A.6)) to define its score:

$$s(X_j^{(i)}, \pi_j^{(i)}) = -d(X_j^{(i)}, \hat{X}_{j,\pi}^{(i)}) - \beta L(\hat{X}_{j,\pi}^{(i)}), \quad (\text{A.12})$$

4: A well-typed tree is one in which every combinator application is type-consistent: the output type of each subtree matches the input type expected by its parent. Ill-typed trees (e.g., applying a transposition operator to a duration argument) are assigned zero prior probability. Type constraints dramatically prune the search space without requiring rejection sampling.

5: Concretely: if a learner has encountered the ascending-fifth subprogram π_1 three times and a chromatic-fill subprogram π_2 once in a sequence, then with $\alpha_{\text{loc}} = 1$ and $d_{\text{loc}} = 0.5$, the probability of reusing π_1 at the next position scales as $(3 - 0.5)/(1 + 4) = 0.5$ and of reusing π_2 as $(1 - 0.5)/(1 + 4) = 0.1$; the remaining 0.4 backs off to G_0 . Increasing d_{loc} would reduce both counts’ advantage, spreading probability more evenly across the library.

6: This “soft” objective is important for modeling human memory errors. A hard criterion would assign zero probability to any program that does not exactly reproduce the target, making recall errors unexplainable. The soft RD score instead assigns nonzero probability to imperfect reconstructions, with magnitude determined by the distortion; this directly predicts the regularization errors documented in the experiment.

7: Online processing is not only a computational convenience but also a cognitive one: it reflects the fact that listeners and performers process sequences incrementally, committing to interpretations before the sequence ends (levy2008expectation).

where $\hat{X}_{j,\pi}^{(i)} = g(\pi_j^{(i)})$ is the reconstruction generated by the subprogram, $d(\cdot, \cdot)$ is the Hamming distance between the observed and reconstructed note sequences, and $L(\hat{X}_{j,\pi}^{(i)})$ is the description length of the retained reconstruction. This score favors subprograms that achieve accurate reconstruction with shorter descriptions. Unlike program-induction approaches that impose a hard consistency criterion (Correa et al., 2023; Ellis et al., 2021; Zhao et al., 2023), this formulation allows partially matching hypotheses to remain under consideration, with support decreasing smoothly as distortion increases.⁶

Because new observations are incorporated sequentially, inference must proceed online;⁷ for long sequences, jointly optimizing over all subprograms is computationally intractable. We therefore evaluate candidate subprograms locally and then compose them into a sequence-level objective. Given a sequence-level program $\Pi^{(i)} = \pi_1^{(i)} \oplus \dots \oplus \pi_{M^{(i)}}^{(i)}$, we define its log-likelihood by summing subprogram scores:

$$\log p(X^{(i)} | \Pi^{(i)}) \propto \sum_{j=1}^{M^{(i)}} s(X_j^{(i)}, \pi_j^{(i)}). \quad (\text{A.13})$$

Approximate inference. The full inference problem is to approximate the joint posterior over sequence-level programs and hierarchical reuse distributions,

$$p(\{\Pi^{(i)}\}_{i=1}^N, \{G_i\}_{i=1}^N, G_0 | \{X^{(i)}\}_{i=1}^N). \quad (\text{A.14})$$

We approximate this joint posterior using a hierarchical, resource-bounded search that interleaves within-sequence program induction and across-sequence HPYP updates.

A.1.3 Inference algorithm

This section provides the full pseudocode for the resource-bounded inference procedure used to approximate the posterior over subprograms and hierarchical libraries. Because exact inference over long sequences is intractable under our typed program grammar, we adopt a streaming, piecewise search that maintains a stack of accepted subprograms and revises only a bounded history:

1. *Within-sequence search.* For each sequence i , inference proceeds over subsequences. At each step, we perform Gibbs sampling under a fixed search budget N_s . Candidate subprograms are proposed from the HPYP mixture, either by reusing from the local and global libraries, or by generating new candidates from the base structure prior (Equation (A.10)), which ensures type-consistent syntactic validity. Each candidate π is evaluated using the loss $\mathcal{J}_{N_s, N_b}(x)$ in Equation (A.6) such that subprograms with lower loss are more likely to be selected, although higher-loss candidates may still be chosen with non-zero probability.
2. *Depth-limited backtracking.* To allow for local revisions when seeing new data, the inference process maintains a stack of accepted subprograms and allows up to N_b backtracking steps. When no local revision improves compression score, the most recent decision is used. Setting $N_b = 0$ implements a purely greedy strategy,

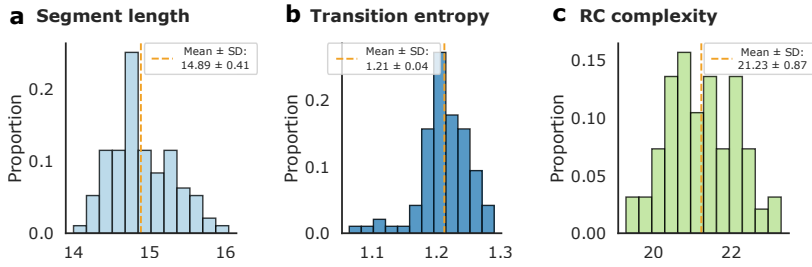


Figure A.1: Distributions of melody statistics across participants. Each panel shows the histogram across participants of the participant-wise mean for the indicated measure. Dashed vertical lines mark the across-participant mean; shaded bands indicate ± 1 s.d. Summary values: (a) $M = 14.89 \pm 0.41$ notes; (b) $H_1 = 1.21 \pm 0.04$ bits/note; (c) $RC = 21.23 \pm 0.87$ symbols (Dehaene et al., 2022).

while increasing N_b progressively approximates optimal dynamic-programming solutions.

3. *Library (HPYP) updates.* After inference selects subprogram assignments $\pi_j^{(i)}$ for sequence $X^{(i)}$, we update the hierarchical reuse prior. In the local library, each selected subprogram instance $\pi_j^{(i)}$ is matched to a distinct reusable subprogram type ϕ_k and increments its count n_{ik} . At the global level, the corresponding cross-sequence counts are updated in G_0 once a complete program $\Pi^{(i)}$ is selected. These count updates modify the predictive distributions G_i and G_0 , increasing the probability of reusing subprograms that have proved useful in previous inference.
4. *Alternation and stopping.* The procedure alternates between within-sequence search and library updates across sequences, resulting in an approximate posterior over sequence-level programs and hierarchical libraries.

Note that procedure in the within-sequence search is *streaming*. It processes notes incrementally and makes local decisions, rather than requiring access to the full sequence before inference begins. Backtracking provides a bounded form of global optimization: it allows the learner to revise earlier commitments in light of later evidence, but only within a finite horizon N_b .⁸

A.2 Data details

Melodic sequence statistics. As shown in Figure A.1, the mean segment length per participant was 14.89 ± 0.41 notes. We deliberately chose this range to make participants have to use compression. The marginal note distribution in the training set was approximately uniform across the six notes, with proportions [.170, .156, .166, .167, .171, .171] (maximum deviation from uniform $< .011$), ensuring that no individual note dominated the material. To test whether simple run-length structure (repetition complexity) and local unpredictability (entropy) capture the same aspect of melodic structure, we correlated repetition complexity (RC (Dehaene et al., 2022; Planton et al., 2021)) with first-order transition entropy (H_1) at the subsequence level. Both linear (Pearson’s r) and rank-based (Spearman’s ρ) associations were near zero and non-significant (two-sided): Pearson’s $r = .018, p = .374$; Spearman’s $\rho = .037, p = .067$. These results indicate that the two measures are only weakly related, if at all, and therefore capture partially different dimensions of melodic complexity.

Behavioral data. Participants’ performance changed systematically across the three experimental phases (Figure A.2). Accuracy was quantified

8: The connection to dynamic programming (DP) is direct. Full DP over a sequence of length T maintains optimal solutions for all prefixes simultaneously and recovers the globally optimal segmentation in $O(T^2)$ time. Depth-limited backtracking approximates this by maintaining only the last N_b decisions on a stack and revising them greedily; it recovers full DP in the limit $N_b \rightarrow T$. This bounded-revision scheme trades global optimality for computational tractability, analogous to bounded lookahead in game-tree search.

Figure A.2: Overall behavioral performance across the three experimental phases. (a) Distributions of per-participant error rates in each phase. Vertical dashed lines indicate phase-wise means. (b) Distributions of per-participant reaction times (log-transformed). Participants respond progressively faster across phases, with the fastest responses during prediction. Vertical dashed lines indicate phase-wise means.

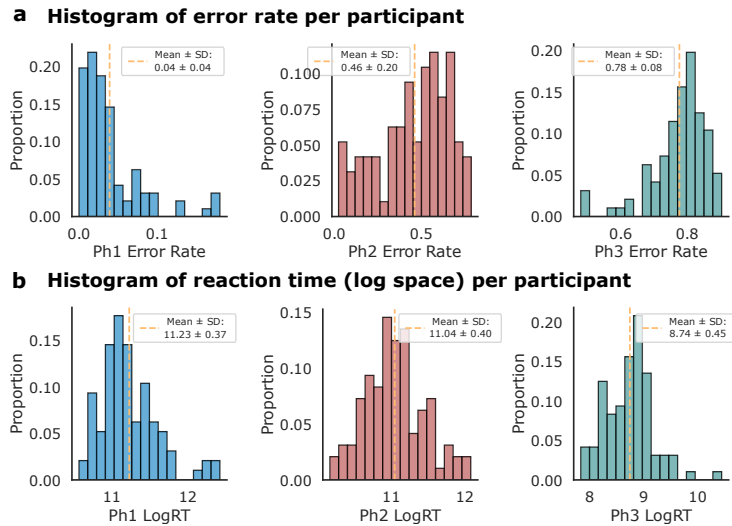
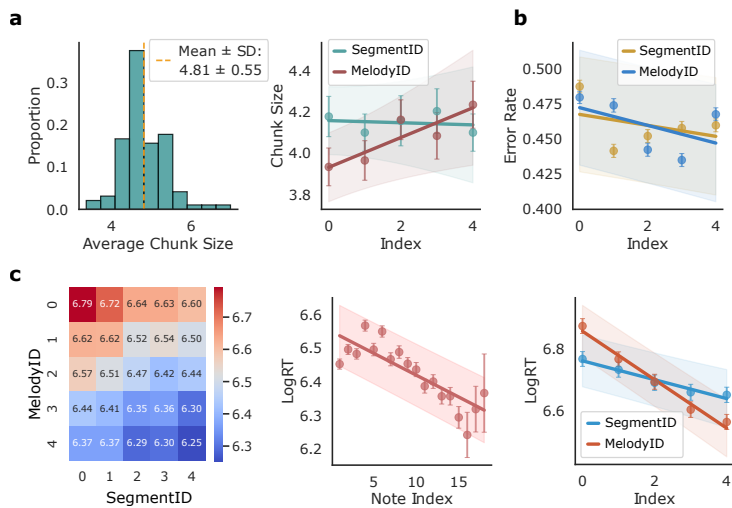


Figure A.3: Behavioral learning dynamics across segments and sequences. (a) Left: Distribution of average chunk sizes inferred from participants' inter-key timing during Phase 2, with dashed line indicating mean $\pm$ s.d. Right: Estimated chunk size as a function of segment index for SegmentID and SequenceID conditions (points denote means; error bars indicate s.e.m.). (b) Error rate as a function of exposure index for SegmentID and SequenceID conditions, with linear trend estimates shown. (c) Left: Heat map of mean log-transformed reaction times (LogRT) across segments and sequences. Middle: Linear mixed-effects regression of LogRT against note index within segments during Phase 2. Right: LogRT as a function of exposure index for SegmentID and SequenceID conditions.



as each participant's average error rate, and response speed as log-transformed reaction time (LogRT). Error rates were lowest during Phase 1 (learning), where participants simply reproduced shown notes. Errors increased substantially in Phase 2 (recall), and peaked in Phase 3 (continuation). Mean error rates were: $.04 \pm .04$, $.46 \pm .20$, and $.78 \pm .08$ in three phases respectively. Reaction times showed the opposite trend. Participants responded more quickly as the task progressed from learning to recall to prediction: 11.23 ± 0.37 , 11.04 ± 0.40 , and 8.74 ± 0.45 in three phases respectively. Figure A.3 characterizes behavioral signatures of learning across repeated exposure to structured melodic sequences. Reaction times and error rates decrease systematically across trials, consistent with increasing fluency and improved memory for repeated structure. Linear mixed-effects analyses reveal progressively faster responses both within and across sequences, indicating learning dynamics over constituent units. In parallel, chunk sizes inferred from inter-key timing increase with practice, suggesting that learners gradually integrate smaller elements into larger, more coherent internal representations. Differences between SegmentID and SequenceID conditions further indicate that representational organization is shaped by the hierarchical structure of the learning environment.

A.3 Paradigm and spans of three observers

This section documents the parameter settings used across the three modeling stages: (i) a simulation study that establishes qualitative behavior across a broad resource grid; (ii) a plausible observer analysis that generates theory-driven behavioral predictors without participant-level fitting; and (iii) participant-level model fitting that estimates individual resource parameters. Each stage uses a different parameter regime chosen for a distinct inferential purpose, described in turn.

Simulation parameters. We evaluated all models across a broad grid of parameter settings intended to (i) span qualitatively distinct resource regimes (weak to strong constraints), (ii) include the resource levels that best fit human behavior while also stress-testing the models outside that regime, and (iii) ensure conclusions do not depend on a narrow tuning of computational or memory assumptions. Because these parameters govern resource limitations during learning, sweeping across them allows us to test whether qualitative model comparisons (e.g., HAG's advantage) are robust across plausible cognitive capacities.

The following parameter grids were used in the simulation study (Section 4.4):

- ▶ Memory-rate parameter β : Controls the trade-off between simplicity and memory capacity. Larger values favor shorter subprograms, reflecting stronger constraints on memory and a preference for parsimony. β was varied across $\{0.1, 0.5, 1.0, 5.0, 10.0\}$
- ▶ Search budget N_s : Controls the number of subprogram proposals evaluated at each time step. Higher values mean greater computational capacity, as more candidates are considered before selecting one. We varied the mean of this budget λ_s across $\{1, 5, 10, 20\}$; realized values N_s were sampled per run from a Poisson distribution with the corresponding mean.⁹
- ▶ Backtracking budget N_b : Controls the extent of backtracking within each sequence. Larger values allow the model to revisit and revise previously chosen subprograms more frequently, again reflecting greater computational capacity. We varied the mean of this budget λ_b across $\{0, 1, 3, 5, 10\}$; realized values N_b were sampled per run from a Poisson distribution with the corresponding mean.
- ▶ reuse parameters in the Pitman–Yor process $\alpha_{loc}, d_{loc}, \alpha_{glob}, d_{glob}$: Control the tendency to reuse existing subprograms in either local or global libraries. Higher values of $\alpha_{loc}, \alpha_{glob}$ correspond to weaker reuse tendencies, indicating increased computational capacity. We varied α_{loc} and α_{glob} across $\{1.0, 5.0, 10.0, 15.0\}$, while holding d_{loc} and d_{glob} constant at 0.2 to avoid confounding effects.

Plausible observer parameters To generate trial-by-trial predictors, we used resource-bounded *plausible observers*.¹⁰ We use *plausible* rather than *ideal* because these models have limited memory and computation, and therefore lack exhaustive search or unlimited representational resources. These observers were not fit separately to participants. Instead, we used them to derive theory-driven predictions across resource settings, asking whether the framework's qualitative predictions remain stable across memory and computational constraints rather than relying on a narrowly tuned parameterization. Parameter grids were selected to span the range of memory and computational resources recovered in the participant-level fits (Figure A.15), while also including somewhat higher resource budgets to test whether qualitative predictions remain robust under slightly less stringent constraints.

9: A Poisson distribution is a natural choice for count-valued resources: it has a single parameter (the mean budget λ), is defined on non-negative integers, and captures variability in cognitive effort across trials. Fixed deterministic budgets would assume identical effort on every sequence; Poisson sampling introduces trial-level variability while keeping the mean as the fitted quantity.

10: An ideal observer is optimal given the task statistics but unconstrained by cognitive resources (Geisler and Diehl, 2003). A plausible observer instead imposes bounded computation and memory, yielding cognitively realistic predictions. Its parameters interpolate between near-ideal behavior with large budgets and highly constrained behavior with small budgets.

The following parameter grids were used in the plausible observer analysis:

- ▶ Memory-rate parameter β : Controls the simplicity–accuracy trade-off. We varied β across $\{0.1, 0.5, 1.0, 5.0, 10.0\}$.
- ▶ Search budget N_s : Controls the number of candidate subprograms evaluated per step. Grid values denote Poisson means $\lambda_s \in \{1, 5, 10, 20\}$; realized budgets were sampled per sequence.
- ▶ Backtracking budget N_b : Controls the depth of revision. Grid values denote Poisson means $\lambda_b \in \{0, 1, 3, 5, 10\}$; realized budgets were sampled per sequence.
- ▶ Reuse parameters $\alpha_{\text{glob}} = \alpha_{\text{loc}}$: For library-based models, global and local innovation rates were varied jointly across $\{1.0, 5.0, 10.0, 15.0\}$.

Model fitting parameters. We estimated five free parameters for the full HAG model, with other models having a subset as nested variants. The parameters were defined as follows:

1. Memory budget (β): This parameter ($\beta \geq 0$) governs the tradeoff between minimizing distortion and complexity in the objective function $D + \beta R$ (Equation (4.1)). A higher β indicates a stronger preference for concise (shorter) programs. Values were optimized via grid search in log-space: $\log(\beta) \in \{0, 0.5, 1, 1.5, 2, 2.5, 3, 3.5\}$.
2. Computational budget (N_s, N_b): These parameters constrain the effort in proposing program candidates. To model cognitive variability, we assumed the computational effort for proposing (N_s) and revising (N_b) programs follows a Poisson process. For each participant, segment, and sequence, budgets were sampled as $N_s \sim \text{Pois}(\lambda_s)$ and $N_b \sim \text{Pois}(\lambda_b) + 1$. The rate parameters were defined along discrete sets $\lambda_s \in \{1, 3, 5, 7, 9\}$ and $\lambda_b \in \{0, 1, 2, 3, 4\}$, where higher rates reflect greater computational resources.
3. Reuse parameters ($\alpha_{\text{glob}}, \alpha_{\text{loc}}$) control the balance between innovation (generating new programs) and reuse. α_{glob} governs the global library, while α_{loc} governs the local library. Higher values favor the generation of new programs over the reuse of existing cached operators. These are optimized via a grid search in log-space: $\log \alpha_{\text{loc}} \in \{-0.5, 0, 0.5, 1, 1.5, 2, 2.5\}$.

The other four models use subsets of these parameters. PCFG includes only resource parameters ($\beta, \lambda_s, \lambda_b$). Chunking, RLE, and AG add a global library governed by α_{glob} ; as α_{glob} increases and reuse is minimized, AG converges to PCFG. HAG further adds a local library governed by α_{loc} ; as α_{loc} increases and local reuse is suppressed, HAG reduces to AG¹¹. To ensure robust inference, each model was fitted using 50 independent random seeds. For each fitted model, we drew 50 posterior samples of the induced programs to construct a categorical distribution over note space. Model log-likelihoods were computed relative to this distribution.

11: This nesting puts all five models on a level playing field: each more complex model contains the simpler ones as special cases. Thus BIC or Bayes-factor comparisons penalize HAG for its additional α_{loc} parameter, so its advantage reflects explanatory gain from local library reuse rather than parameter count alone.

12: HAG vs. AG: $z(99) = 1183, p < .001, r = -.531, \text{CI} = [-.702, -.337]$

13: $z(99) = 1387, p < .001, r = -.451, \text{CI} = [-.649, -.241]$

A.4 Supplemental simulation results

Generalization performance on unseen sequences. In Figure 4.4, HAG achieved the most favorable RD trade-off with AG typically second-best. A paired-samples Wilcoxon test confirms that HAG attains significantly lower distortion and rate than AG¹². AG in turn outperformed RLE¹³. In Figure A.4, we report generalization performance under the parameter setting $N_s = 10, N_b = 1, \alpha_{\text{glob}} = \alpha_{\text{loc}} = 1.0$, and $d_{\text{glob}} = d_{\text{loc}} = 0.2$. After training the library on 50 sequences, we evaluated compression

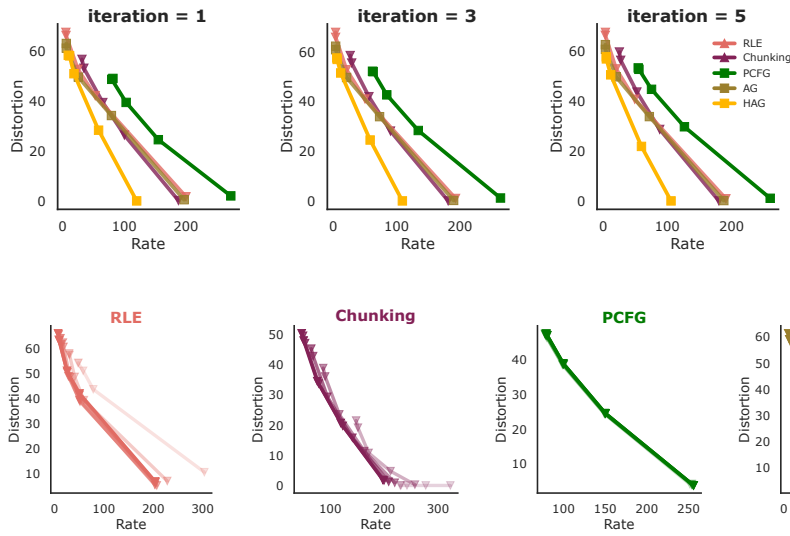


Figure A.4: Generalization rate-distortion (RD) performance. One-shot, three-shot, and five-shot generalization RD curves obtained by compressing unseen sequences using the trained library without additional optimization iterations. All results use $N_s = 10$, $N_b = 1$, $\alpha_{loc} = \alpha_{glob} = 1.0$, and $d_{loc} = d_{glob} = 0.2$. The library was trained on 50 sequences, and the curves report compression performance on held-out sequences.

Figure A.5: Rate-distortion (RD) trajectories during training across models. RD curves over the course of training for each model. Within each panel, color intensity (increasing opacity) corresponds to increasing data amount with lighter curves indicating limited training data and darker curves indicating more training data (training data percentage = 2%, 10%, 40%, 80%, 100%, $N_s = 10$, $N_b = 1$). Across models, except for PCFG which does not have a library, increased training data shifts RD curves toward lower distortion at comparable rates.

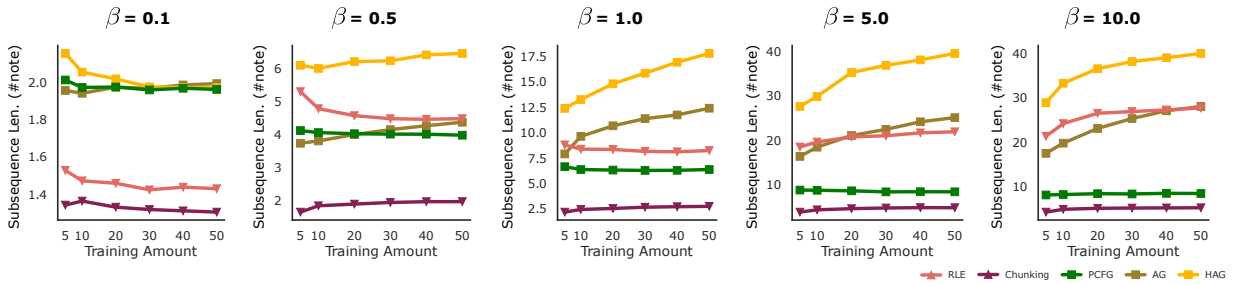


Figure A.6: Reconstructed subsequence length under varying memory constraints (β). Average subsequence length per selected subprogram (in notes) as a function of training data size, shown separately for different values of the memory-rate parameter β . As β increases, AG and especially HAG produce progressively longer reconstructed subsequences. With additional training data, this trend becomes stronger for AG and HAG, whereas Chunking, PCFG, and RLE change comparatively little.

performance on a separate set of previously unseen sequences. During evaluation, we reused the trained library but varied the number of optimization iterations. Increasing the number of iterations improves absolute compression performance but does not alter the ordering of models. HAG consistently outperforms AG across all iteration counts¹⁴. However, varying β does affect relative performance: when the memory constraint is made sufficiently strong (e.g., $\beta = 10.0$), the objective becomes dominated by rate, reducing the advantage of maintaining hierarchical libraries since all models are pushed toward maximally compact representations regardless of structure, and HAG's advantage over AG diminishes¹⁵.

RD performance as training data increases. Figure A.5 shows how library-based models are built on increasing training sequences to improve compression efficiency. While PCFG shows no change in its RD curve, all models equipped with library learning, Chunking, RLE, AG and HAG, gradually shift their RD frontiers downward, indicating improved distortion for a given rate. To show how the training process affects compositional reconstruction, we evaluated the subsequence length per subprogram under varying values of the memory-rate param-

14: HAG vs. AG: $z(99) = 81, p < .001, r = -.968, CI = [-.991, -.925]$; AG vs. RLE: $z(99) = 4008, p = .999, r = .587, CI = [.408, .740]$

15: When $\beta = 10.0$, HAG vs. AG: $z(99) = 396, p = .010, r = -.38, CI = [-.66, -.08]$.

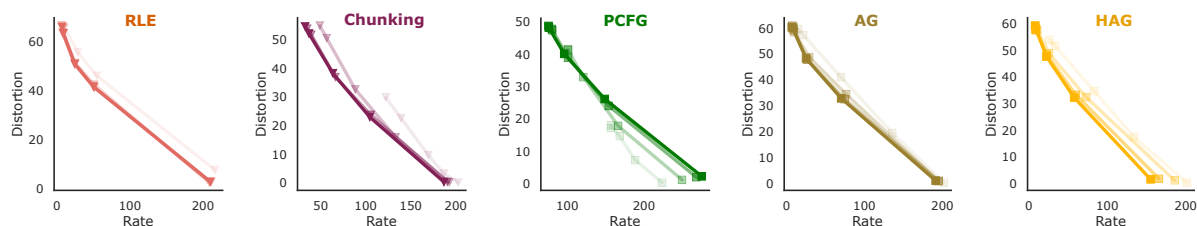
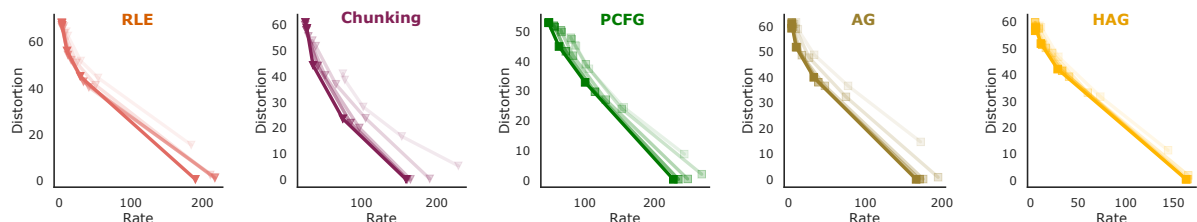
a Search budget**b Backtracking budget**

Figure A.7: Rate-distortion (RD) curves across models under varying computational resources. (a) search budgets N_s and **(b)** backtracking budgets N_b . Within each panel, color intensity (increasing opacity) corresponds to increasing computation with lighter curves indicating lower budgets and darker curves indicating higher budgets ($N_s = 1, 5, 10, 20$, $N_b = 0, 1, 3, 5, 10$).

16: HAG vs. AG: $z(99) = 1268$, $p < .001$, $r = .98$, $CI = [.96, .99]$

17: HAG vs. Chunking: $z(99) = 1275$, $p < .001$, $r = 1.0$, $CI = [1.0, 1.0]$

18: HAG vs. AG: $z(99) = 296$, $p = .99$, $r = -.53$, $CI = [-.79, -.25]$; HAG vs. PCFG: $z(99) = 651$, $p = .45$, $r = .02$, $CI = [-.29, .35]$

eter β (Figure 4.4a). We fixed all other parameters to $N_s = 10$, $N_b = 1$, $\alpha_{\text{loc}} = \alpha_{\text{glob}} = 1.0$, $d_{\text{glob}} = d_{\text{loc}} = 0.2$, and imposed no limit on the library sizes. HAG subprograms consistently produced the longest subsequences followed by AG¹⁶, while Chunking yielded substantially shorter subsequences¹⁷. In Figure A.6, we disentangle the effect of β by plotting the same measure separately for each value. As β increases, the model's capacity to store or reuse longer, more complex subprograms decreases, resulting in shorter reconstructed subsequences even as training data accumulate. Conversely, lower β values enable longer, more detailed reconstructions, consistent with looser memory constraints. In this regime (e.g., when $\beta = 0.1$), HAG does not exceed AG¹⁸, suggesting that extremely tight memory constraints suppress hierarchical abstraction: when all models are forced toward maximally short subprograms, the structural advantage of local library reuse disappears and HAG reduces to AG in practice.

Effects of memory and training data on compositional reconstruction.

Figure A.7a shows how increasing the search budget (N_s) influences rate-distortion trade-offs across all model classes. For all models except from PCFG, larger search budgets systematically improve performance, yielding lower distortion at comparable rates, indicating that broader search enables more effective discovery of high-quality subprograms. In PCFG, the apparent advantage of the smaller-search curve reflects a *simplicity bias* induced by limited search. When N_s is small, inference evaluates only a few candidate derivations and therefore tends to accept “default” parses that are low-complexity (fewer tree expansions). As N_s increases, the model is more likely to propose deeper-nested subprograms or more specific combinations of productions, even though they did not prove to be useful in the past. These unnecessarily complex subprograms can increase rate and also the distortion. This different directions of performance changes between with or without global libraries suggest that accumulated global library constrains the proposal distribution, enabling efficient exploration even under limited search. Together, these results indicate that the impact of search budget depends on the extent to which model structure supports reuse and compositional generalization. Figure A.7b shows how backtracking budget (N_b) influences rate-distortion

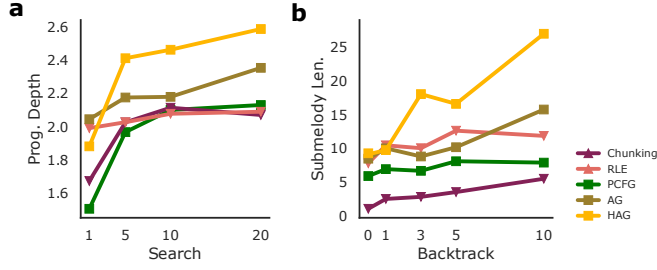


Figure A.8: Effects of search budget and backtracking budget on subprogram structure. (a) Average subprogram depth as a function of search budget ($N_s = 1, 5, 10, 20$). (b) Average subsequence length as a function of backtracking budget ($N_b = 0, 1, 3, 5, 10$).

trade-offs across all model classes. Increasing backtracking N_b improves rate–distortion performance for all models, indicating that the ability to revise earlier decisions enables more effective subprogram refinement. This effect is especially pronounced for models without local libraries RLE, Chunking, PCFG, and AG. In contrast, HAG shows more modest improvements.

Effects of search and backtracking budgets. Figure A.8 characterizes how search budget and backtracking budget shape the structural properties of inferred subprograms. Larger search budgets N_s are associated with increased subprogram depth, indicating that broader exploration helps discover more deeply compositional solutions. In contrast, increasing backtracking budget N_b mainly affects encoding length, as models iteratively revise and elaborate candidate subprograms. Together, these results highlight complementary roles of search and backtracking: search expands the space of compositions considered, while backtracking refines solutions within that space.

A.5 Supplemental behavioral analyses

A.5.1 Error pattern details

We analyzed recall errors by assigning each error to one of four categories (Figure 4.6). For each sequence and participant, we scanned all contiguous windows $[a : b]$ and evaluated candidate error categories whenever at least one note differed, i.e., $d(X_{a:b}, \tilde{X}_{a:b}) \geq 1$.

Vertical shifts capture errors where participants recalled the correct relative interval structure, but transposed the true note up or down in pitch. We computed element-wise pitch differences $d_j = \tilde{x}_j - x_j$. If there was a constant pitch offset across the window $d_a = \dots = d_b = c \neq 0$, we classified the error as a vertical shift with magnitude $|c|$.

Temporal shifts correspond to when participants recalled the correct notes, but at the wrong temporal position. We tested whether the recalled notes matched the ground truth after a uniform index shift in time. Specifically, if there exists a lag s such that $a' = a + s \geq 0$, $b' = b + s \leq T$, and $\tilde{X}_{a':b'} = X_{a:b}$, then the window was classified as a temporal shift with lag s .

Repetition errors capture when participants produced notes that matched earlier patterns rather than the current target. We operationalized two forms of repetition errors within a recalled window $\tilde{X}_{a:b}$: (i) *Run repetition* corresponds to repeating (once or multiple times) the immediately preceding recalled note. Let $p = \tilde{X}_{a-1}$ (with $a > 0$) and let $\mathbf{1}$ denote an all-ones vector of length $W = b - a + 1$. If $\tilde{X}_{a:b} = p\mathbf{1}$, we labeled this as a run repetition and report the run length W ; (ii)

Chunk repetition corresponds to repeating (once or multiple times) an earlier sequence of multiple notes. Let j index the current segment. For each earlier segment $j' < j$ within the same melody, we checked whether the recalled window copied a contiguous earlier ground-truth sequence, i.e., there exist indices $u:v$ within $X_{j'}$ such that $\tilde{X}_{a:b} = X_{u:v}$ while $X_{a:b} \neq X_{u:v}$. If so, we labeled the window as a chunk repetition with segment lag $\Delta j = j - j'$.

Simpler programs capture errors in which an incorrect participant response was better explained by a more compressible program than the ground-truth pattern. Using a precomputed library of candidate programs (across multiple depths), we scored each window W by

$$v(X) = \log \left(1 - \frac{d(X, g(\pi))}{W} + \varepsilon \right) + \frac{\log p(\pi)}{W}, \quad (\text{A.15})$$

where $d(X, g(\pi))$ is the normalized Hamming distortion, and $\log p(\pi)$ is the program prior code length (both computed from the fixed candidate set, not fit to participants). For each window, we computed the best achievable score for the participant response (v_{resp}) and for the ground truth (v_{gt}). If $v_{\text{resp}} - v_{\text{gt}} > 0$, we labeled the error simpler program, with advantage $v_{\text{resp}} - v_{\text{gt}}$.

A.5.2 Predicting choices

To characterize how different models predicted participants' recall response in Phase 2, we fit a hierarchical mixed-effects multinomial regression model (Figure 4.7). The outcome variable was the selected note on each response, encoded as a categorical variable with six possible pitches. Predictors comprised two descriptive control models for non-structured errors (vertical shift and temporal shift), and five plausible-observer models for structured errors. Specifically, repetition-like errors were captured by RLE and Chunking, whereas errors favoring a more compressible latent program were captured by PCFG, AG, and HAG. For each observation, predictor values were aligned over a common ordering of the six candidate notes, resulting in a tensor X of shape $(N \times C \times M)$, where N is the number of observations, $C = 6$ the number of possible notes, and $M = 7$ the number of models.

Each predictor set m was assigned a population-level weight $\omega_m \sim \mathcal{N}(0, \sigma_\omega^2)$, capturing its average contribution to note choice, together with participant-specific random slopes $u_{p,m} \sim \mathcal{N}(0, \sigma_{u,m}^2)$ allowing for their contributions to vary across participants. We also included participant-specific note intercepts $v_{p,c}$, constrained to sum to zero across notes within each participant, to capture baseline note preferences.

The linear predictor combined fixed and random effects as follows:

$$\eta_{n,c} = \sum_{m=1}^M X_{n,c,m} (\omega_m + u_{p[n],m}) + v_{p[n],c}, \quad (\text{A.16})$$

where $p[n]$ denotes the participant associated with observation n . Choice probabilities were then given by a softmax function:

$$p_{n,c} = \frac{\exp(\eta_{n,c})}{\sum_{c'} \exp(\eta_{n,c'})}, \quad (\text{A.17})$$

and participant responses were modeled as categorical draws from this distribution. Inference was performed using automatic differentiation variational inference (ADVI (Kucukelbir et al., 2017)) to approximate the posterior distribution, followed by posterior sampling from the variational approximation¹⁹. This approach was chosen for computational tractability given the large number of observations and hierarchical parameters. Results reported in Figure 4.7 reflect posterior summaries of fixed and random effects.

A.5.3 Reaction time analyses

To infer whether participants “chunk” notes together during recall, we detected sudden positive “jumps” in consecutive LogRT within each recalled segment with $\Delta r_j = \log r_{j+1} - \log r_j$ for $j \in [1, \dots, L - 1]$. This corresponds to a slowdown in reaction, following a window of relatively faster responses. A position j is marked as a chunk boundary if $\Delta r_j > \mu_{\Delta r} + \sigma_{\Delta r}$ where $\mu_{\Delta r}$ and $\sigma_{\Delta r}$ are the mean and standard deviation of $\{\Delta r_j\}$ within the same segment. Since these boundaries partition the segment into subsequences, we define chunk size as the length of these consecutive notes (subsequence length). If no jump is detected, the whole segment is treated as one chunk²⁰.

We first characterize baseline temporal dynamics in sequence processing (i.e., faster RTs over trials) by fitting a linear mixed-effects model predicting log-transformed RTs from the serial note position within a segment $\text{LogRT} \sim \text{NoteID} + (1 \mid \text{ParticipantID})$. Here, NoteID represents the ordinal position of each note within its segment, and random intercepts account for individual differences in baseline response speed. We used restricted maximum likelihood (REML) estimation to fit the model (Figure A.3).

To test whether human processing time reflects program-based compression (Figure 4.7), we fitted a mixed-effects model incorporating hierarchical program boundary predictions from our set of plausible observers alongside baseline comparisons: $\text{LogRT} \sim \text{SequenceID} + \text{SegmentID} + \text{NoteID} + \text{RLE} + \text{Chunking} + \text{PCFG} + \text{AG} + \text{HAG} + (1 \mid \text{ParticipantID})$ where SequenceID and SegmentID capture hierarchical positional effects, while boundary predictors represent normalized segmentation strengths. Positive regressors thus indicate predictive power in explaining jumps in RTs.

A.5.4 Distortion analyses

To test whether recall errors reflect only surface-level sequence statistics or instead track the representational costs implied by program compression, we analyzed segment-level error rates as a function of both stimulus properties and plausible-observer program complexity (Figure 4.7). For each participant, we summarized each training segment and computed objective descriptors of its local statistics, then asked whether model-inferred program rate explained variance in errors beyond these descriptors. The sequence descriptors below serve as stimulus-level covariates (i.e., nuisance variables) capturing generic difficulty, such as longer sequences or entropy. Including them allows us to isolate the unique relationship between errors and the model-derived program rate (description length) for each model.

19: ADVI was preferred over MCMC because the model has a large number of hierarchical random effects (~50 participants $\times$ 7 models), making full HMC sampling computationally prohibitive. Posterior predictive checks confirmed that the variational approximation closely matched the marginal response distributions.

20: RT jumps are a standard behavioral marker of hierarchical chunking in sequence production (Sakai et al., 2003; Verwey and Dronkert, 1996): slower onset of a new chunk reflects the cognitive cost of initiating a new motor or memory program, whereas notes within a chunk are executed more fluently.

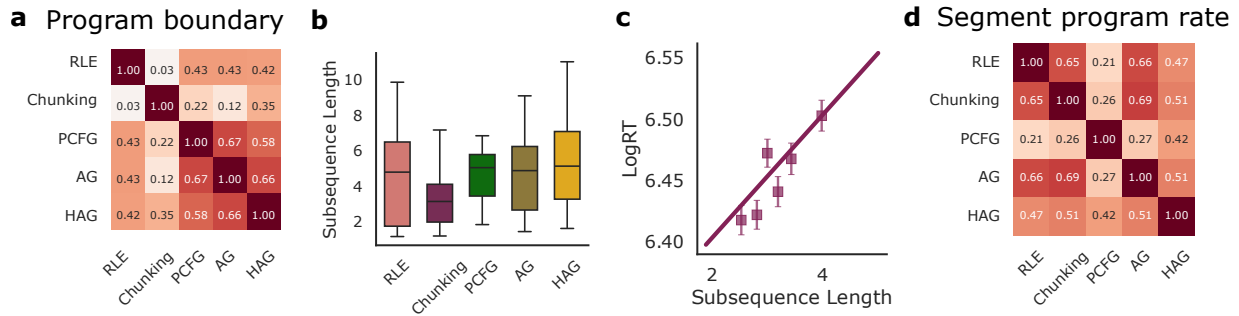


Figure A.9: Correlation structure and representational scale of plausible-observer-based predictors. (a) Mean within-participant correlations among model-predicted boundary signals. (b) Distribution of subsequence lengths inferred by each model. Chunking produced shorter subsequences overall than other models. (c) Relationship between subsequence length and log RT for the Chunking model. Longer Chunking-defined subsequences were associated with slower responses, consistent with a mixed-effects regression showing that subsequence length positively predicted log RT ($\gamma = .028$, $z = 3.01$, $p = .003$, $CI = [.010, .047]$). (d) Mean within-participant correlations among model-based segment-complexity.

21: First-order transition entropy H_1 and repetition complexity RC are only weakly correlated in our stimulus set (Pearson $r = .018$, $p = .374$; see [Section A.2](#)), so including both as covariates does not introduce collinearity—they capture partially orthogonal aspects of melodic difficulty.

For each segment $X^{(i,j)}$, we computed the following quantities: (i) Length T measures number of notes in $X^{(i,j)}$; (ii) Entropy of first-order transition distribution $H_1(X) = -\sum_{n \in \mathcal{A}} p(n) \sum_{m \in \mathcal{A}} p(m|n) \log_2 p(m|n)$ (bits per note) measures the local unpredictability of the next note given the current one, by treating each sequence as a first-order Markov chain over the note alphabet $\mathcal{A}$ in $X_j^{(i)21}$.

Together with the complexity (i.e., rate) outputs of our plausible observers, the full mixed-effects model predicted participants' error rates using: $\text{ErrorRate} \sim \text{SequenceID} + \text{SegmentID} + T + H_1 + \text{RLE} + \text{Chunking} + \text{PCFG} + \text{AG} + \text{HAG} + (1 | \text{ParticipantID})$. Intuitively, a positive relationship indicates that higher human error rates correspond to higher complexity in a given plausible observer, while accounting for lower-level factors (i.e., length and entropy).

A.5.5 Correlation analysis in regression

In [Figure 4.7](#), we showed different explanations in RT jumps from each model. AG and PCFG did not explain RT independently of HAG. Their boundary predictions were moderately correlated with HAG and with each other²², indicating shared variance among the hierarchical predictors. Accordingly, in the full regression, HAG captured the variance most closely associated with hierarchical transition costs, leaving AG negative and PCFG non-significant. We therefore interpret the AG and PCFG coefficients as reflecting collinearity, rather than evidence that these models predict the opposite behavioral effect.

Chunking showed a qualitatively different relationship to RT. Whereas HAG and RLE predicted increased RT at model-defined boundaries, Chunking was associated with faster responses. This pattern is unlikely to reflect suppression alone, because Chunking was only modestly correlated with the hierarchical predictors²³. To clarify this dissociation, we quantified the size of model-defined subsequences and found that Chunking only captured small-unit reuse, producing shorter subsequences overall ([Figure A.9](#)). Subsequence length was itself positively associated with LogRT in a mixed-effects regression²⁴, such that larger chunks were associated with slower responses. This difference in subsequence length may explain the opposite regression direction in two ways. At a cognitive level, shorter chunk-defined units may support more fluent local execution and therefore lower RT. At a statistical level,

22: AG-HAG: $r = .66$; PCFG-HAG: $r = .58$; AG-PCFG: $r = .67$; [Figure A.9](#)

23: Chunking-HAG: $r = .35$

24: $\gamma = 0.028$, $z = 3.01$, $p = .003$, $CI = [.010, .047]$

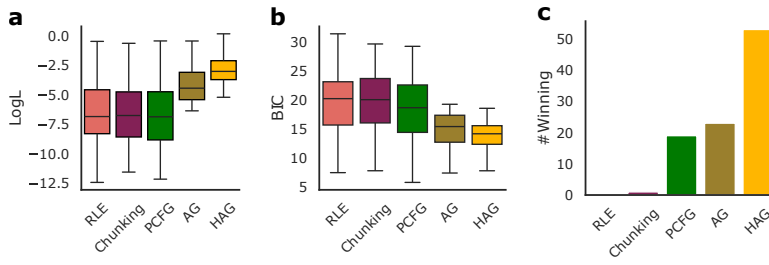


Figure A.10: Model-fit comparison across candidate models. Boxplots summarize the distribution across participants of (a) maximized log-likelihood (LogL; higher indicates better fit). (b) Bayesian Information Criterion (BIC; lower indicates better fit after penalizing model complexity). (c) Bars show the number of participants (among 96 participants in total) for which each model achieved the lowest BIC (“wins”). Boxes indicate the median and interquartile range; whiskers extend to $1.5 \times IQR$; points denote outliers.

because Chunking predicts boundaries more densely, its regressor is dominated by boundaries occurring within short subsequences, which are behaviorally faster on average. Together, these factors can account for why Chunking differs in sign from the hierarchical predictors.

A.6 Model fitting details

A.6.1 Fitting procedure

We estimated five free parameters for the full HAG model, with other models having a subset as nested variants. The parameters are: memory budget β (governing the rate–distortion trade-off), computational budgets N_s and N_b (governing search and backtracking, modelled as Poisson-distributed to capture cognitive variability), and reuse preferences α_{glob} and α_{loc} (governing innovation vs. reuse in the global and local libraries). Full parameter grids are reported in [Section A.3](#).

For each fitted model, we drew 50 posterior samples of the induced programs to construct a categorical distribution over note space. Model log-likelihoods were computed relative to this distribution. We refer to this analysis as a *plausible observer* rather than an ideal observer because the model is explicitly resource-bounded: it does not assume exhaustive search or unconstrained memory.

A.6.2 Supplementary results

Model fitting scores. Across participants, HAG shows consistently better fit than the alternative accounts: it achieves higher log-likelihoods²⁵ while also yielding lower BIC values²⁶, indicating that the improvement is not explained solely by increased flexibility. The winner-amount comparison based on BIC converges with the continuous metrics: HAG is the most frequent top-ranked model, winning for a substantial share of participants (53/96), with AG winning in a smaller subset (21/96)²⁷. The remaining baselines rarely achieve the lowest BIC.

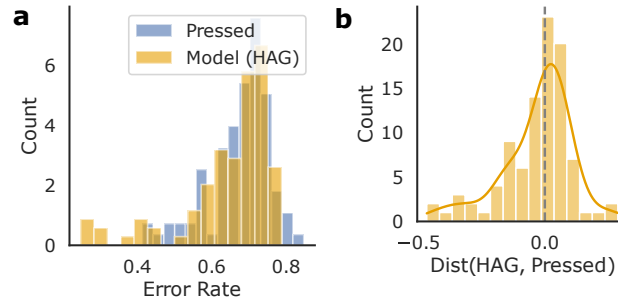
Phase 3 prediction. In Phase 3, participants generated free continuations of the sequences, allowing us to test whether the libraries inferred from earlier phases generalize to out-of-sample composition. For each participant, we fit HAG to their behavior up to the onset of Phase 3, yielding an individualized library that can be used as a generative model to sample candidate continuations. Rather than asking whether either participants or the model reproduced the held-out melody exactly, we asked whether HAG generated continuations that resembled participants’ continuations.

25: HAG: $-2.774 \pm .121$, AG: $-4.178 \pm .158$, PCFG: $-6.518 \pm .298$, Chunking: $-6.488 \pm .264$, RLE: $-6.442 \pm .276$

26: HAG: $13.596 \pm .243$, AG: $14.779 \pm .316$, PCFG: $17.865 \pm .597$, Chunking: $19.415 \pm .528$, RLE: $19.322 \pm .553$

27: BIC penalizes each additional free parameter by $\frac{1}{2} \log N$, where N is the number of observations. HAG has one more parameter than AG (α_{loc}); its BIC advantage therefore reflects a likelihood gain that exceeds the penalty, not merely greater flexibility.

Figure A.11: Distributional match between human Phase-3 composition errors and HAG-generated continuations. (a) Histogram of Phase-3 error rates, measured as note-wise mismatch from the held-out ground-truth continuation, for participants' pressed continuations (blue) and continuations sampled from each participant's fitted HAG library (orange). (b) Histogram of the per-participant discrepancy between model and human error rates, defined as $\text{Dist}(\text{HAG}, \text{Pressed}) = \text{ErrorRate}_{\text{HAG}} - \text{ErrorRate}_{\text{Pressed}}$; the dashed line marks zero, indicating a perfect match. The distribution is centered near zero, showing that participant-specific HAG libraries reproduce population-level variability in composition accuracy.



As shown in Figure A.11, the error-rate distributions of HAG-generated and participant-produced continuations closely overlapped, and the per-participant discrepancy between $\text{ErrorRate}_{\text{HAG}}$ and $\text{ErrorRate}_{\text{Pressed}}$ was concentrated near zero. Thus, participant-specific libraries capture both the overall distribution and the between-participant variability of continuation behavior.

Learner simulation using fitted parameters. We evaluated each fitted model by simulating behavior from participant-specific posterior samples and comparing predictions with Phase 2 recall and Phase 3 continuation. For each participant and model, simulations used the best-fitting trial-by-trial parameter estimates, plus inferred local and global libraries where applicable. To capture stochastic-inference uncertainty, we retained 50 posterior samples from independent random seeds; each sample specified a complete fitted model, including parameters and representational structure.

For Phase 2, we simulated recall for each learned subsequence and extracted simulated error rate, defined as average Hamming distance from the target, and simulated chunk size, defined from the lengths of subprograms in the inferred representation. We then related these simulated quantities to participants' observed error rates and RT-derived chunk sizes. Figure 4.8 summarizes model performance by correlating observed behavior with predictions from fitted parameters, without additional simulation. Figure A.12 additionally reports correlations and mixed-effects regressions predicting participant measures from simulated measures across candidate models, with participant as a random effect and marginal regressions plotted. For Phase 3, we used the fitted models and libraries as generative models of continuation. For each participant, we generated synthetic continuations from the 50 posterior samples, estimated a predictive categorical distribution over responses at each note, and computed the likelihood of the participant's actual continuation under this distribution.

Beyond relating fitted parameters to aggregate behavioral measures, we asked whether the fitted models also captured graded variation in behavior across segments. For each participant, we simulated the fitted model on the same set of sequences encountered in the experiment and, for each segment, derived a predicted error rate and a predicted chunk-size measure. We compared these model-derived quantities with the corresponding observed measures: actual error rate and chunk size inferred from RT. By fitting mixed-effects regressions in which the empir-

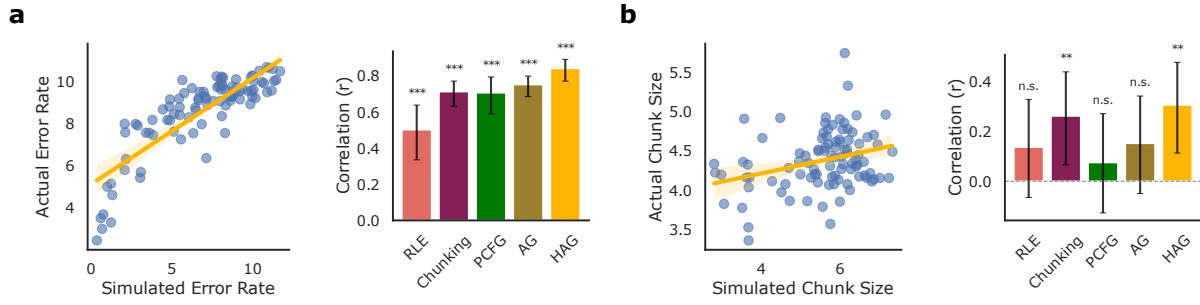


Figure A.12: Simulation-based prediction of error rate and chunk size. (a) Error rate. Left, partial effect of the HAG predictor in a mixed-effects regression relating actual error rates to simulated error rates from all candidate models, with participant as a random effect ($\gamma = .760$, $z = 4.47$, $p < .001$, $CI = [.427, 1.092]$). Right, mean Pearson correlation between simulated and actual error rates across participants for each model. (b) Chunk size. Left, partial effect of the HAG predictor in a mixed-effects regression relating RT-inferred actual chunk size to simulated chunk size from all candidate models ($\gamma = .151$, $z = 2.101$, $p < .05$, $CI = [.010, .292]$). Right, mean Pearson correlation between simulated and actual chunk sizes across participants for each model. Each point denotes one participant; lines show the fitted HAG relationship after controlling for competing models, and error bars denote 95%

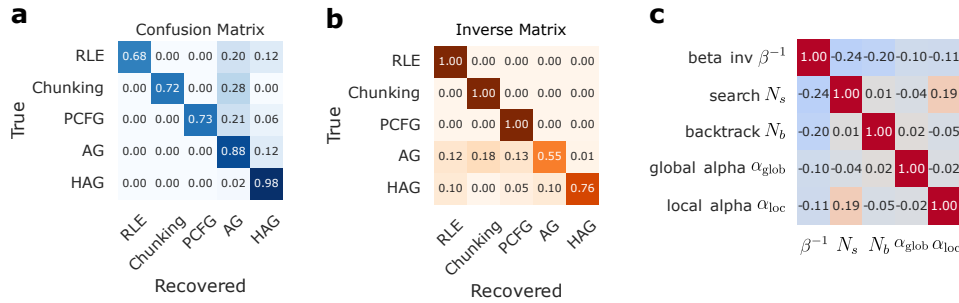


Figure A.13: Model recovery and parameter interdependence. (a) Model-recovery confusion matrix. Rows indicate the true data-generating model and columns indicate the recovered model. Cell values report the proportion of synthetic datasets assigned to each recovered model (row-normalized). (b) Inverse matrix. Rows indicate the true model and cell values are normalized by recovered model (column-normalized), showing $P(\text{true model} | \text{recovered model})$ to quantify how diagnostic each recovered label is of the generating process. (c) Across-participant parameter correlations. Heat map shows pairwise Pearson correlations between fitted parameter estimates across participants (inverse beta, search, backtracking, global alpha, local alpha); diagonal entries are 1 by definition.

ical measure was predicted jointly by the simulated predictions from all candidate models, the regression line shown for HAG in Figure A.12 represents its partial effect after controlling for competing model predictions. To complement this analysis, the figure also reports participant-level correlations between actual and simulated values for each model.

HAG explained significant unique variance in observed error rate, indicating that the fitted model captured graded differences in behavioral difficulty across segments. The same pattern held for chunk size inferred from RT, indicating that HAG also tracked variation in the granularity of participants' internal representations.

Model recovery. To assess identifiability under realistic operating conditions, we conducted a model-recovery analysis in which synthetic datasets were generated from each candidate model and then refit with the full model set using the same likelihood and fitting pipeline as in the main analyses. We first fit each model to participants' data and then defined an empirical parameter range for recovery using the distribution of best-fitting parameter values observed across participants. For each generating model, we sampled parameters from this empirically grounded range, simulated datasets matched to the experimental design, and applied the same model-comparison criterion to assign a recovered label. This approach targets practical identifiability in the regime actually supported by the data, i.e., where fitted parameters tend to lie, rather than identifiability under arbitrary parameter settings that participants

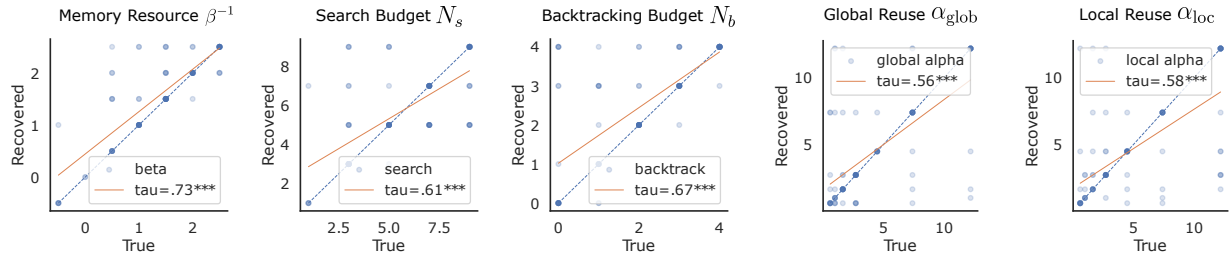


Figure A.14: Parameter recovery for HAG. Each panel plots recovered estimates against the ground-truth generating values for one parameter: inverse memory resource constraint β^{-1} , search budget N_s , backtracking budget N_b , global reuse α_{glob} , and local reuse α_{loc} . Points correspond to participants; dashed line indicates perfect recovery (identity line). Reported Kendall's τ summarizes rank-order correspondence between true and recovered parameters (all $^{***}p < .001$).

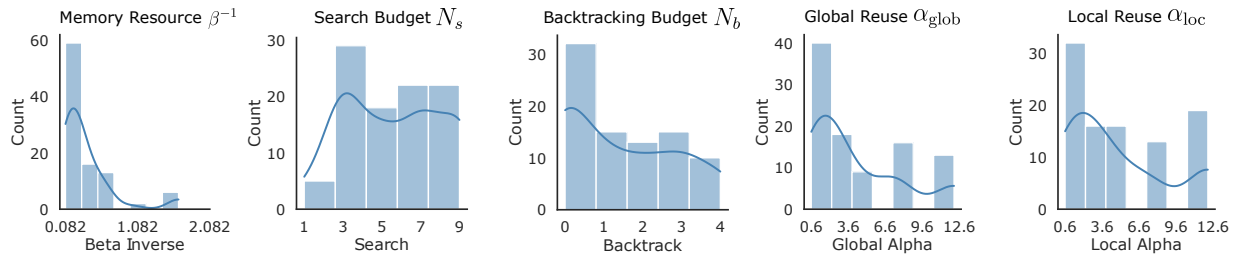


Figure A.15: Empirical distribution of fitted HAG parameters. Histograms show the distribution of participant-level maximum-likelihood estimates for the five HAG parameters. Curves indicate kernel density estimates to visualize central tendency and dispersion. The distributions reveal substantial between-participant variability.

never exhibit.

Model-recovery analyses indicate that each candidate model is generally identifiable: recovery is strongest for HAG (.98 on the diagonal) and AG (.88), with moderate diagonal recovery for PCFG (.73), Chunking (.72), and RLE (.68). Misclassifications primarily reflect a bias toward selecting AG (e.g., a subset of PCFG/Chunking/RLE fitting are recovered as AG), whereas confusion between HAG and other models is rare (only .02 of HAG datasets recovered as AG). The inverse confusion matrix further shows that when PCFG, Chunking, or RLE is recovered, it is highly diagnostic of the true generating model, while recovered AG and HAG labels are somewhat less exclusive, reflecting overlap in the patterns they can produce. Finally, fitted parameters exhibit generally weak cross-participant correlations (largest magnitudes $\sim .20 - .25$), suggesting limited trade-offs among parameters and supporting practical identifiability of the parameterization at the group level.

Parameter recovery and distribution. To verify that our fitted parameters are interpretable and not artifacts of overfitting, we conducted a parameter recovery analysis. We treated each participant's fitted HAG as a data-generating model, simulated synthetic choices using their estimated parameters, and then applied the same fitting pipeline to recover parameters from the simulated data. As shown in Figure A.14, recovered values closely track the generating values and preserve participants' rank ordering (Kendall's τ shown in each panel), supporting the identifiability of the parameters used in subsequent analyses.

We next summarize the fitted parameter estimates across participants to characterize variability in the strategies inferred by HAG. Figure A.15 plots the marginal distributions of maximum-likelihood parameters, showing heterogeneity in memory constraints (β^{-1}), the extent of search and backtracking during inference, and the strength of library reuse at global versus local scales (α_{glob} , α_{loc}). This variability motivates our

individual-differences analyses linking fitted parameters to learning history and composition performance.

Additional Material for Chapter 5

B.

The appendix for Chapter 5 is organized as follows:

- ▶ **Section B.1** provides a detailed derivation of our objective function ELBO in two scenarios: inference involving complete learning histories and inference for in a continual learning setting (**Section 5.3.3**).
- ▶ **Section B.2** provides in-depth descriptions of baseline models, and the details and the selection criterion of the three datasets we use for experiments.
- ▶ **Section B.3** describes the PSI-KT architecture in full detail, including its hyperparameters.
- ▶ **Section B.4** provides additional prediction results. We show the average accuracy, average f1-score, and their standard error of **Figure 5.5** in the prediction experiment given entire learning histories. We also show the average accuracy score and their standard error of **Figure 5.6** in the prediction experiment for continual learning setup.
- ▶ **Section B.5** describes the details of the experiment setup and how we derive the metrics for specificity, consistency, and disentanglement. We also provide comprehensive results on the operational interpretability of baseline models.
- ▶ **Section B.6** elaborates on the graph assessment framework, including details about the alignment metrics and a discussion of causal support, and extends the main text evaluations to all datasets.

B.1 ELBO of the hierarchical SSM	209
B.2 Baseline models and datasets	212
B.3 PSI-KT model architecture	216
B.4 Prediction and generalization experiments details	218
B.5 Learner-specific representations analysis	221
B.6 Graph inference analysis	227
B.7 Ablation study	233

B.1 ELBO of the hierarchical SSM

In this section, we derive the single-learner ELBO, **Equation (5.12)**-**Equation (5.14)** in the main chapter. For clarity, we omit the superindex ℓ in these derivations. Note that the parameters ϕ and θ are global, i.e., they are optimized based on the entire interaction data across learners.

In variational inference (VI), we approximate an intractable posterior distribution $p_\theta(z|y)$ with $q_\phi(z|y)$ from a tractable distribution family. We learn ϕ and θ together by maximizing the evidence lower bound (ELBO) of the marginal likelihood (Attias, 1999; Blei et al., 2017), given by $\log p_\theta(y) \geq \text{ELBO}(\theta, \phi) = \mathbb{E}_{q_\phi(z|y)} [-\log q_\phi(z|y) + \log p_\theta(y, z)]$. The two terms in the ELBO represent the entropy of the variational posterior distribution, $H[q_\phi(z|y)] = \mathbb{E}_{q_\phi(z|y)} [-\log q_\phi(z|y)]$, and the log-likelihood of the joint distribution of observations and latent states $\mathbb{E}_{q_\phi(z|y)} \log p_\theta(y, z)$.

We now formulate the ELBO for our hierarchical SSM (**Figure 5.2**) with two layers of latent states. We assume that fixed learning histories $\mathcal{H}_{1:N}$ until time t_N are available and we use capital N to represent the fixed time point. We approximate the posterior $p_\theta(\mathbf{z}_{1:N}, \mathbf{s}_{1:N} | \mathbf{y}_{1:N})$ using the mean-field factorization, $q_\phi(\mathbf{z}_{1:N}, \mathbf{s}_{1:N} | \mathbf{y}_{1:N}) = q_\phi(\mathbf{z}_{1:N}) q_\phi(\mathbf{s}_{1:N})$:

In the generative model of PSI-KT, the observation y_n at time t_n depends on the particular knowledge state z_n^k associated with the interacted KC $k = x_n$. All knowledge states z_n rely on previous states z_{n-1} and cognitive traits s_n , which themselves are influenced by s_{n-1} . Thus, we

$$\begin{aligned}
\text{ELBO}(\theta, \phi) &= \mathbb{H}[q_\phi(\mathbf{z}_{1:N}, s_{1:N} | \mathbf{y}_{1:N})] + \mathbb{E}_{q_\phi(\mathbf{z}_{1:N}, s_{1:N} | \mathbf{y}_{1:N})} \log p_\theta(\mathbf{y}_{1:N}, \mathbf{z}_{1:N}, s_{1:N}) \\
&= \mathbb{E}_{q_\phi(\mathbf{z}_{1:N}, s_{1:N} | \mathbf{y}_{1:N})} [-\log q_\phi(\mathbf{z}_{1:N}, s_{1:N} | \mathbf{y}_{1:N}) + \log p_\theta(\mathbf{y}_{1:N}, \mathbf{z}_{1:N}, s_{1:N})] \\
&= \mathbb{E}_{q_\phi(\mathbf{z}_{1:N}) q_\phi(s_{1:N})} [-\log q_\phi(\mathbf{z}_{1:N}) - \log q_\phi(s_{1:N}) + \log p_\theta(\mathbf{y}_{1:N}, \mathbf{z}_{1:N}, s_{1:N})]. \tag{B.1}
\end{aligned}$$

can factorize the joint distribution $p_\theta(\mathbf{y}_{1:N}, \mathbf{z}_{1:N}, s_{1:N})$ in Equation (B.1) over all latent states and observations:

$$\begin{aligned}
p_\theta(\mathbf{y}_{1:N}, \mathbf{z}_{1:N}, s_{1:N}) &= p_\theta(s_{1:N}) p_\theta(\mathbf{z}_{1:N} | s_{1:N}) \prod_{n=1}^N p_\theta(y_n | z_n^{x_n}) \\
&= p_\theta(s_1) p_\theta(\mathbf{z}_1) \prod_{n=2}^N p_\theta(s_n | s_{n-1}) p_\theta(\mathbf{z}_n | \mathbf{z}_{n-1}, s_n) \prod_{n=1}^N p_\theta(y_n | z_n^{x_n}). \tag{B.2}
\end{aligned}$$

Here, $p_\theta(s_1)$ and $p_\theta(\mathbf{z}_1)$ are the Gaussian initial priors for the latent states.

By incorporating the factorized joint distribution (Equation (B.2)), the ELBO for PSI-KT can be derived as follows:

$$\begin{aligned}
\text{ELBO}(\theta, \phi) &= \mathbb{E}_{q_\phi(\mathbf{z}_{1:N}) q_\phi(s_{1:N})} [-\log q_\phi(\mathbf{z}_{1:N}) - \log q_\phi(s_{1:N}) + \log p_\theta(\mathbf{y}_{1:N}, \mathbf{z}_{1:N}, s_{1:N})] \\
&= \mathbb{E}_{q_\phi(\mathbf{z}_{1:N}) q_\phi(s_{1:N})} \left[-\log q_\phi(\mathbf{z}_{1:N}) - \log q_\phi(s_{1:N}) + \log p_\theta(s_1) + \log p_\theta(\mathbf{z}_1) \right. \\
&\quad \left. + \sum_{n=2}^N \log p_\theta(s_n | s_{n-1}) + \sum_{n=2}^N \log p_\theta(\mathbf{z}_n | \mathbf{z}_{n-1}, s_n) \right. \\
&\quad \left. + \sum_{n=1}^N \log p_\theta(y_n | z_n^{x_n}) \right] \\
&= \mathbb{E}_{q_\phi(s_{1:N})} [-\log q_\phi(s_{1:N}) + \log p_\theta(s_1) + \sum_{n=2}^N \log p_\theta(s_n | s_{n-1})] \\
&\quad + \mathbb{E}_{q_\phi(\mathbf{z}_{1:N})} [-\log q_\phi(\mathbf{z}_{1:N}) + \log p_\theta(\mathbf{z}_1) + \sum_{n=1}^N \log p_\theta(y_n | z_n^{x_n})] \\
&\quad + \mathbb{E}_{q_\phi(\mathbf{z}_{1:N}) q_\phi(s_{1:N})} \left[\sum_{n=2}^N \log p_\theta(\mathbf{z}_n | \mathbf{z}_{n-1}, s_n) \right]. \tag{B.3}
\end{aligned}$$

B.1.1 Extension to continual learning

We now extend the ELBO to the continual learning setting, where we observe learning performances $y_{1:n}$ sequentially. Here we use the lower case n to indicate the running time index. We seek the posterior distribution $p_\theta(\mathbf{z}_n, s_n | y_{1:n})$ at each interaction time t_n given all observations so far. Usually, one would approximate the posterior with the variational posterior distribution $q_{\phi_n}(\mathbf{z}_n, s_n | y_{1:n}) = q_{\phi_n}(\mathbf{z}_n) q_{\phi_n}(s_n)$ using the mean-field factorization (Equation (B.1)). In that case, the inference process consists of maximizing the $\text{ELBO}(\theta, \phi_n)$ only over ϕ_n :

$$\text{ELBO}(\theta, \phi_n) = \mathbb{E}_{q_{\phi_n}(\mathbf{z}_n)q_{\phi_n}(s_n)}[-\log q_{\phi_n}(\mathbf{z}_n) - \log q_{\phi_n}(s_n) + \log p_{\theta}(y_{1:n}, \mathbf{z}_n, s_n)]. \quad (\text{B.4})$$

However, it is challenging to calculate the joint distribution $p_{\theta}(y_{1:n}, \mathbf{z}_n, s_n)$ in our setup, since it requires marginalizing the full joint distribution $p_{\theta}(y_{1:n}, \mathbf{z}_{1:n}, s_{1:n})$ over all $\mathbf{z}_{n'}$ and $s_{n'}$ with $n' < n$.

Henceforth, we aim to reconfigure the objective function $\text{ELBO}(\theta, \phi_n)$, which involves the variational posterior distribution $q_{\phi_n}(\mathbf{z}_n, s_n | y_{1:n})$ at the time point t_n , to establish a linkage with the posterior $q_{\phi_{n-1}}(\mathbf{z}_{n-1}, s_{n-1} | y_{1:n-1})$ observed at the preceding time point t_{n-1} . By doing so, we can recursively optimize the variational parameters ϕ_n whenever new observations y_n are received (Nguyen et al., 2017), wherein the initialization draws from the parameters ϕ_{n-1} obtained at time t_{n-1} .

First, we show that the marginal joint distribution $p_{\theta}(y_{1:n}, \mathbf{z}_n, s_n)$ is proportional to the prior distribution $p_{\theta}(\mathbf{z}_n, s_n | y_{1:n-1})$ on $s_n, \mathbf{z}_n$ at t_{n-1} :

$$p_{\theta}(y_{1:n}, \mathbf{z}_n, s_n) \propto p_{\theta}(\mathbf{z}_n, s_n | y_{1:n-1}) p_{\theta}(y_n | \mathbf{z}_n). \quad (\text{B.5})$$

Second, we show how the prior distribution $p_{\theta}(\mathbf{z}_n, s_n | y_{1:n-1})$ can be formulated using the posterior $q_{\phi_{n-1}}(\mathbf{z}_{n-1}, s_{n-1} | y_{1:n-1})$ at the previous time t_{n-1} , which evolves for a single time step:

$$\begin{aligned} p_{\theta}(\mathbf{z}_n, s_n | y_{1:n-1}) &= \int p_{\theta}(\mathbf{z}_n, \mathbf{z}_{n-1}, s_n, s_{n-1} | y_{1:n-1}) ds_{n-1} d\mathbf{z}_{n-1} \\ &= \int p_{\theta}(\mathbf{z}_{n-1}, s_{n-1} | y_{1:n-1}) p_{\theta}(s_n | s_{n-1}) p_{\theta}(\mathbf{z}_n | s_n, \mathbf{z}_{n-1}) ds_{n-1} d\mathbf{z}_{n-1} \\ &= \int q_{\phi_{n-1}}(\mathbf{z}_{n-1}, s_{n-1} | y_{1:n-1}) p_{\theta}(s_n | s_{n-1}) p_{\theta}(\mathbf{z}_n | s_n, \mathbf{z}_{n-1}) ds_{n-1} d\mathbf{z}_{n-1} \\ &= \int q_{\phi_{n-1}}(\mathbf{z}_{n-1}) q_{\phi_{n-1}}(s_{n-1}) p_{\theta}(s_n | s_{n-1}) p_{\theta}(\mathbf{z}_n | s_n, \mathbf{z}_{n-1}) ds_{n-1} d\mathbf{z}_{n-1} \\ &= \underbrace{\mathbb{E}_{q_{\phi_{n-1}}(s_{n-1})}[p_{\theta}(s_n | s_{n-1})]}_{:= \tilde{p}_{\phi_{n-1}}(s_n)} \underbrace{\mathbb{E}_{q_{\phi_{n-1}}(\mathbf{z}_{n-1})}[p_{\theta}(\mathbf{z}_n | s_n, \mathbf{z}_{n-1})]}_{:= \tilde{p}_{\phi_{n-1}}(\mathbf{z}_n | s_n)}. \end{aligned} \quad (\text{B.6})$$

Substituting $p_{\theta}(s_n, \mathbf{z}_n | y_{1:n-1}) = \tilde{p}_{\phi_{n-1}}(s_n) \tilde{p}_{\phi_{n-1}}(\mathbf{z}_n | s_n)$ using our variational approximation in Equation (B.5) in the ELBO, we finally arrive at the objective function for variational continuous learning: This provides a derivation of Equation (5.14) as presented in the main text. Here our focus lies in the optimization of the parameters ϕ_n , while holding constant the parameters ϕ_{n-1} acquired from the preceding time step.

$$\begin{aligned}
\text{ELBO}_{\text{VCL}}(\theta, \phi_n) &= \mathbb{E}_{q_{\phi_n}(s_n) q_{\phi_n}(z_n)} \left[-\log q_{\phi_n}(s_n) - \log q_{\phi_n}(z_n) + \log \tilde{p}_{\phi_{n-1}}(s_n) \right. \\
&\quad \left. + \log \tilde{p}_{\phi_{n-1}}(z_n | s_n) + \log p_{\theta}(y_n | z_n^{x_n}) \right] \\
&= \mathbb{E}_{q_{\phi_n}(s_n)} \left[-\log q_{\phi_n}(s_n) + \log \tilde{p}_{\phi_{n-1}}(s_n) \right] \\
&\quad + \mathbb{E}_{q_{\phi_n}(z_n)} \left[-\log q_{\phi_n}(z_n) + \log p_{\theta}(y_n | z_n^{x_n}) \right] \\
&\quad + \mathbb{E}_{q_{\phi_n}(z_n) q_{\phi_n}(s_n)} \left[\log \tilde{p}_{\phi_{n-1}}(z_n | s_n) \right].
\end{aligned} \tag{B.7}$$

B.2 Baseline models and datasets

B.2.1 Baselines

KT models aim to predict the performance $\hat{y}_n^\ell$ of the presented KC x_n^ℓ at time t_n^ℓ for each learner ℓ , which amounts to learning the mapping $\hat{y}_n^\ell = f_{\theta}(\mathcal{H}_{n', < n}^\ell)$ (Section 5.2.1). Because baseline models lack learner-specific parameters, we here describe the prediction process for a single learner, and omit the superindex ℓ for clarity. Extending to multiple learners is straightforward since all parameters θ are global. We use $\tau_n := t_n - t_{n-1}$ to represent the time interval between consecutive interactions of a learner ℓ , and the KC-specific interval $\tau_n^k := t_n^k - t_{n-1}^k$ for consecutive interactions with the same KC k . The number of practice repetitions for each KC k up to time t_n is denoted as c_n^k . The dimension of embeddings D equals 16 in our experiments.

Table B.1: Models. # Emb/KC is the number of learnable embeddings per KC. *Forgetting* is the functional form of memory decay, with exponential (exp) decay the most common.

Feature	HLR/PPE	DKT	DKTF	HKT	AKT	GKT	QIKT	PSI-KT
# Emb/KC	×	2	2	6	6	3	1	1
Forgetting	exp	×	exp	Hawkes	×	×	×	OU
KC Sim	×	✓	✓	✓	✓	✓	×	✓
KC Pre	×	×	×	×	×	✓	×	✓

We compare with eight baseline models (Section 5.4):

1: HLR can be viewed as a classic spaced-repetition model: recall decays with time since the last practice, while repeated successful practice increases the half-life.

- HLR (Settles and Meeder, 2016)¹ uses the cumulative counts of correct, incorrect, and total interactions of KC k until time t_n , collectively denoted $c_n^k = [c_n^{k,1} c_n^{k,0} c_n^k]^\top \in \mathbb{R}^3$, as well as the last interval τ_n^k . When a learner interacts with KC x_n at time t_n , HLR predicts the probability of a correct performance as:

$$\hat{y}_n := 2^{-\tau_n^k / h_n^k}, \quad \text{with memory half-life } h_n^k := 2^{\theta^\top c_n^k} \text{ and } k = x_n. \tag{B.8}$$

The learnable weights $\theta \in \mathbb{R}^3$ modulate the influences of correct, incorrect, and total interaction counts. The training process of HLR does not differentiate features from different KCs or learners, thus HLR cannot model the relational structure of KCs or any learner-specific characteristics.

2: Unlike HLR, PPE explicitly separates a learning effect from a forgetting effect.

- PPE (Walsh et al., 2018)² is similar to HLR in predicting performance as a function of interaction histories. It defines the activation m_n^k of KC k at time t_n , $m_n^k := (c_n^k)^\beta (T_n^k)^{-\alpha}$ with separate forgetting rate α and learning rate β . The forgetting term T_n^k is a function of the interaction history, which it summarizes as a weighted average of

times τ_n^k elapsed between the exposures to a given KC prior to t_n :

$$T_n^k := \sum_{i=1}^{n-1} w_i^k \tau_i^k, \quad \text{with } w_i^k = (\tau_i^k)^{-\eta} \sum_{j=1}^{n-1} (\tau_j^k)^\eta. \quad (\text{B.9})$$

The forgetting rate α is a function of a *stability term* κ and a cumulative average of interval durations between KC exposures modulated by the slope λ :

$$\alpha_n^k = \kappa + \lambda \left(\frac{1}{n-1} \sum_{j=1}^{n-1} \frac{1}{\ln(\tau_j^k + e)} \right). \quad (\text{B.10})$$

Finally, PPE treats performance $\hat{y}_n$ as a logistic function of m_n^k with $k = x_n$. The learnable parameters are $\theta = \{\beta, \eta, \kappa, \lambda\}$.

- DKT (Piech et al., 2015)³ infers two separate embeddings $u^k = \{u^{k,0}, u^{k,1}\}$ for each KC k , depending on performance. Here, $u^{k,0}, u^{k,1} \in \mathbb{R}^D$ represent incorrect interactions and correct interactions on KC k , respectively, and are shared across all learners, with D being the dimensionality of the embeddings. DKT trains an LSTM (Hochreiter and Schmidhuber, 1997) over $\mathcal{H}_{n' < n}$ to encode the combined information of KC indices and performance. For each learner ℓ and all time points $t_{n' < n}$, DKT takes performance embeddings $u^{x_{n'},0}$ of interacted KC $x_{n'}$ as the input when the performance $y_{n'}$ is incorrect, or $u^{x_{n'},1}$ for correct performance, i.e., DKT takes inputs $\mathbf{u}_{n'} = u^{x_{n'}, y_{n'}}$ for all $t_{n'}$ with $n' < n$. DKT then predicts the subsequent performance on all KCs $\hat{\mathbf{y}}_n = [y_n^1, \dots, y_n^K]^\top \in \mathbb{R}^K$, and chooses only the interacted one, i.e., the x_n -th dimension:

$$\begin{aligned} \mathbf{h}_n &= \text{LSTM}(\mathbf{u}_{n' < n}; \mathbf{W}_h, \mathbf{b}_h) \\ \hat{\mathbf{y}}_n &= \sigma(\mathbf{W}_{\hat{y}h} \mathbf{h}_n + \mathbf{b}_{\hat{y}}) \\ \hat{y}_n &= \hat{\mathbf{y}}_n[x_n]. \end{aligned} \quad (\text{B.11})$$

Thus, θ for DKT consists of the neural network parameters $\mathbf{W}_h, \mathbf{b}_h, \mathbf{W}_{\hat{y}h}, \mathbf{b}_{\hat{y}}$.

- DKTF (Nagatani et al., 2019)⁴ uses the same LSTM architecture and the same combined KC-performance embeddings, $u^k = \{u^{k,0}, u^{k,1}\}$ that we described above for DKT. DKTF uses additional 3-dimensional features $\mathbf{t}_n := [\tau_n, \tau_n^k, c_n^k]$ representing the KC-unspecific and KC-specific intervals defined above and the cumulative interaction counts c_n^k for KC k until time t_n . Then, for inputs of every time point $t_{n'}$, DKTF concatenates the time information $\mathbf{t}_{n'}$ with KC performance inputs $\mathbf{u}_{n'}$ for the interacted KC $x_{n'}$. DKTF predicts future performances following the same architecture based on concatenated input $[\mathbf{t}_{n' < n}; \mathbf{u}_{n' < n}]$.
- HKT (Wang et al., 2021)⁵ is the most similar model to our PSI-KT. It uses a Hawkes process to model the structural influence on the state of KC k due to every other KCs state in the past interactions $i \in x_{n' < n}$ until time t_n :

$$\begin{aligned} m_n^k &= \lambda_k + \sum_{i \in x_{n' < n}} a_n^{i,k} \kappa(t_n^k - t_{n'}^i) \\ \kappa(t_n^k - t_{n'}^i) &= \exp\left(-(1 + \beta_n^{i,k} \log(t_n^k - t_{n'}^i))\right). \end{aligned} \quad (\text{B.12})$$

Here m_n^k includes a base level λ_k and all previous learned KCs' influences $a_n^{i,k}$ weighted by the a temporal exponential decay $\kappa(t_n^k -$

3: DKT is purely sequential: it summarizes the interaction history in order, but does not explicitly encode elapsed time or KC relations.

4: DKTF extends DKT by appending handcrafted temporal features to the sequence input.

5: HKT explicitly models transfer across KCs: past work on one KC can influence another KC through a decaying cross-KC effect.

6: We use *assignment* to denote the observed problem or item presented to the learner, whereas a *KC* denotes the underlying skill or concept. Multiple assignments may map to the same KC.

7: AKT learns dependencies implicitly through self-attention rather than through an explicit graph or Hawkes kernel.

8: GKT assumes knowledge components form a graph and updates neighboring KC states jointly.

$t_{n'}^i$). The base level λ_k reflects aspects of KC k but also of the specific assignments that were interacted with at time point t_n , given that distinct assignments can provide practice for a single KC⁶. To model cross-KC influences $a_n^{i,k}$, HKT infers embeddings $\{u_a^{k,0}, u_a^{k,1}, u_a^k\} \in \mathbb{R}^D$ for each KC k . Here $u_a^{k,0}, u_a^{k,1}$ are defined similarly to the DKT embeddings, whereas u_a^k only depends on KC identity k . When interacting with KC k at time t_n , the influence on its state due to having interacted with KC i at time $t_{n'}$ with performance $y_{n'}$ is estimated as $a_n^{i,k} = (u_a^{i,y_{n'}})^\top u_a^k$. For the coefficient $\beta_n^{i,k}$, HKT estimates three additional KC-specific embeddings $\{u_\beta^{k,0}, u_\beta^{k,1}, u_\beta^k\}$, and follows similar calculations as for a above. HKT also predicts the performance $\hat{y}_n$ as a logistic function of m_n^k with $k = x_n$.

► AKT (Ghosh et al., 2020)⁷ is a transformer-based model (Vaswani et al., 2017b) that learns the structure of KCs implicitly from the self-attention weights. Unlike LSTM models, which only capture temporal information, AKT captures both temporal and structural relations. Specifically, AKT first initializes three embeddings $\{u_a^{k,0}, u_a^{k,1}, u_a^k\}$ for each KC k and a scalar μ^q for each specific assignment q , representing its difficulty level. For each KC k , these embeddings are defined as in HKT to separately reflect KC-specific correct/incorrect interactions and KC identity. However, AKT combines these representations with three additional embeddings $\{u_b^{k,0}, u_b^{k,1}, u_b^k\}$, in order to account for difficulty levels. When a learner interacts at time $t_{n'}$ with an assignment q related to KC k , the KC identity embedding becomes $u^k = u_a^k + \mu^q u_b^k$; after assessing the performance at time $t_{n'}$, the interaction embeddings are similarly updated as $u^{k,y_{n'}} = u_b^{k,y_{n'}} + \mu^q u_b^{k,y_{n'}}$. Consequently, a learner's entire interaction history $\mathcal{H}_{n'<n}$ is represented as a sequence of these combined KC-interaction-difficulty embeddings. AKT processes these sequential embeddings as input, using KC embeddings as queries and keys, and interaction embeddings as values within its attention mechanism. To predict performance $\hat{y}_n$ given the KC and assignment, AKT uses the KC embeddings at time t_n to compare with previous queries and keys in the learning history, and then extract the value. Details about the transformer architecture can be found in Ghosh et al. (2020).

► GKT (Nakagawa et al., 2019)⁸ applies a graph neural network to leverage the graph-structured nature of knowledge. Like AKT, GKT initializes three embeddings $\{u^{k,0}, u^{k,1}, u^k\}$ for each KC k , but instead of only using the embeddings to determine the KC relations, GKT learns an additional undirected KC graph, represented by its adjacency matrix A . Here $a_{ij} = 1$ represents KC i and KC j are related, i.e., there is information transmission among KC i and KC j every time the model gets updated. To use the KC relations, GKT first aggregates the hidden states h_n^k and embeddings for the KC reviewed at time t_n , k and its neighboring KCs i :

$$(h_n^k)' = \begin{cases} [h_n^k, u^{x_n, y_n}] & (i = k) \\ [h_n^k, u^i] & (i \neq k \text{ with } a_{ik} = 1) \end{cases} \quad (\text{B.13})$$

After aggregating the information from the neighboring KCs, GKT updates the hidden states based on the aggregated features and

	Assist12		Assist17		Junyi15	
	All	> 50	All	> 50	All	> 50
# Interactions	6,123,270	2,431,788	942,816	942,489	25,925,992	23,907,121
# Learners	46,674	12,443	1,709	1,697	247,606	77,655
# Assignments	179,999	51,866	3,162	3,162	5,174	6,174
# KCs	265	263	102	102	722	721
KC Examples	Rounding		substitution		matrix_basic_distance	
	Unit Rate		fraction-division		circles_and_arcs	
	Perimeter of a Polygon		prime-number		arithmetic_means	

Table B.2: Overview of Educational Datasets Assist12, Assist17, and Junyi15, including the number of interactions, learners, assignments, and KCs from overall log data (All) and the log data including learners with more than 50 interactions (> 50).

the graph structure:

$$h_{n+1}^k = \begin{cases} f_{\theta}(h_n^{k'}, h_n^k) & (i = k) \\ f_{\theta}((h_n^k)', (h_n^i)', h_n^k) & (i \neq k \text{ with } a_{ik} = 1). \end{cases} \quad (\text{B.14})$$

Finally, GKT uses an MLP layer to predict the probability of a correct answer at the next time step, $\hat{y}_{n+1}^k = f_{\theta}(h_{n+1}^k)$ where $k = x_{n+1}$.

- QIKT (Chen et al., 2023)⁹ focuses on assignments together with KCs, where multiple different assignments can test one KC. Inspired by item-response theory (IRT) (Lord, 2012a), QIKT defines three modules, each parameterized by a neural network, to infer interpretable features, namely assignment-specific knowledge acquisition α_n , assignment-specific problem-solving ability ζ_{n+1} , and assignment-agnostic but KC-specific knowledge mastery β_n . Apart from neural network parameters, QIKT learns three sets of assignment-specific embeddings $\{v^{q,0}, v^{q,1}, v^q\}$, which have the same purpose as the KC embeddings defined in AKT, namely for correct interactions, incorrect interactions, and assignment identity. Furthermore, another set of KC-specific embeddings u^k is learned for KC-specific features. QIKT uses LSTM and sum pooling to learn the three features based on each learner's history, $\alpha_n := f_{\theta}(V_{1:n}, U_{1:n})$, $\beta_n := f_{\theta}(U_{1:n})$, where $V_{1:n}$ and $U_{1:n}$ denote respectively the all the assignments and KC embeddings in the learning history. The problem-solving ability $\zeta_{n+1} := f_{\theta}(V_{1:n+1}, U_{1:n+1})$ is learned by including the assignment and KC information in the coming interaction. To predict performance, all three features are aggregated and input into the sigmoid function, $\hat{y}_{n+1} = \sigma(\alpha_n + \beta_n + \zeta_{n+1})$.

9: QIKT is item-aware: it distinguishes the assignment being attempted from the KC being assessed.

B.2.2 Datasets

Here we describe the datasets that we have used for evaluation (Assist12, Assist17, and Junyi15), articulate the reasons for their selection, and discuss some of the limitations derived from this choice.

Description of the selected datasets. Assist12 and Assist17 are subsets of the ASSISTments dataset released by Worcester Polytechnic Institute (Selent et al., 2016). ASSISTments is an online math learning platform used in U.S. grades 4–12, primarily by middle school students in Massachusetts and nearby areas. It supports classroom and homework assignments, with or without paper materials, and provides immediate correctness feedback after each student response.

Junyi Academy is a non-profit Chinese online education platform. Its Junyi15 release contains one year of math assignment logs from over 72,000 students, totaling 16 million attempts, plus two annotation sets on the structure of 837 curriculum KCs. Expert annotations from three teachers identify 553 prerequisite relations, while crowd-sourced annotations from 51 senior-high-or-higher graduates rate prerequisite and

similarity strength for 1,954 KC pairs on a 0–9 scale, with at least three ratings per pair.

B.3 PSI-KT model architecture

In this section, we introduce the detailed architecture of our PSI-KT model and its hyperparameters. The inference network consists of an embedding network f_{ϕ}^{Emb} , the cognitive traits encoder f_{ϕ}^s , and the knowledge states encoder f_{ϕ}^z . The weights of these interconnected networks collectively constitute the inference parameters ϕ .

Interaction embedding network. The network f_{ϕ}^{Emb} extracts features from the learning history tuples $\mathcal{H}_{1:N}^{\ell} = \{x_n, y_n, t_n\}_{1:N}^{\ell}$, combining information about interaction time, KC identity and performance. The *KC identity embedding* for KC x_n corresponds to the learned embedding u^{x_n} , which is part of the generative model that parameterizes the graph structure. The *performance embedding* is obtained by expanding the scalar value y_n into a vector $\mathbf{y}_n$ with the same dimensionality as the time and KC embeddings so that the performance features will be represented on an equal footing. We then concatenate the KC embedding u^{x_n} with the performance embedding $\mathbf{y}_n$. The *interval embedding* is a positional encoding (Vaswani et al., 2017b), $\text{PE}_n = (\sin \alpha(\tau_n); \cos \alpha(\tau_n))$. Using sinusoidal interval encodings allows similar time gaps to map to similar representations while preserving sensitivity across both short and long timescales. Thus, the joint embedding for a learning interaction is given by $v_n = f_{\phi, \text{Emb}}([u^{x_n}; \mathbf{y}_n]) + \text{PE}_n$, inspired by the transformer architecture (Vaswani et al., 2017b).

Latent state encoder. The network f_{ϕ}^z infers the parameters of the variational posterior distribution $q_{\phi}(z_{1:n})$. Since learning histories do not have a pre-determined length, we use an LSTM (Hochreiter and Schmidhuber, 1997) as the inference architecture. At each time point, we extract the hidden states in the LSTM, $h_{z_{1:n}} = \text{LSTM}(v_{1:n})$. Meanwhile, in the continual learning setting, information about the history is already encoded and available in the variational parameters for the last time step ϕ_{n-1} , so we use a multi-layer perceptron (MLP), $h_{z_n} = \text{MLP}(v_n)$. Finally, another MLP Kingma and Welling (similar to the encoder in 2014) takes the hidden states h_{z_n} at every time point as inputs and produces the mean $\mu_{z_n} \in \mathbb{R}^K$ and log-variance $\log \sigma_{z_n} \in \mathbb{R}^K$ for knowledge states z_n .

Latent trait encoder. The network f_{ϕ}^s infers the parameters of the variational posterior distribution $q_{\phi}(s_{1:n})$. The resulting approximate posterior distribution enables the sampling of learner-specific traits to facilitate personalized predictions. One immediately obvious approach is to use the same architecture of f_{ϕ}^z . However, the unimodal Gaussian prior over the latent variables cannot account for the diversity of cognitive trait combinations that we expect to find across learners in diverse cohorts. What we need is to allow for multimodality¹⁰ in the distribution of s over all learners.

There is work on factorizing the joint variational posterior as a combination of isotropic posteriors, using a mixture of M experts (MoE; Shi et al., 2019), i.e., $q_{\phi}(s_{1:n}^{\ell} | \mathcal{H}_{1:n}^{\ell}) = 1/M \sum_m q_{\phi_m}(s_{1:n}^{\ell} | \mathcal{H}_{1:n}^{\ell})$, assuming the different modalities are of comparable complexity. However, this may lead to over-parameterization. Instead, inspired by Dilokthanakul et al. (2016), we opt for a mixture of Gaussians as a prior distribution that

10: Learners with similar observed response sequences need not share the same underlying cognitive profile. A unimodal Gaussian prior tends to concentrate mass around a single average profile and therefore may blur such qualitatively distinct learner types in the latent space.

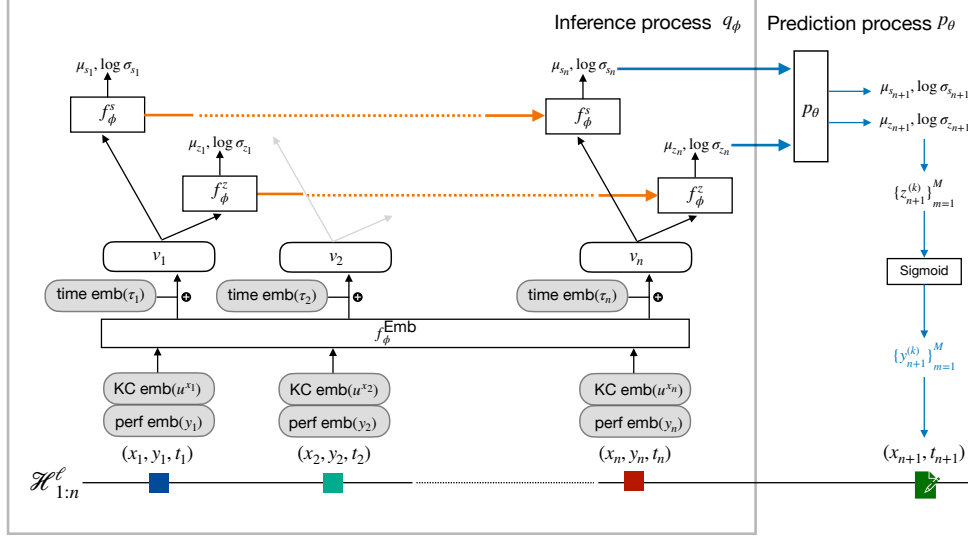


Figure B.1: Inference model of PSI-KT with an example of a single learner's history as the input. Note that all parameters ϕ, θ are shared across learners. Grey backgrounds mark inputs. Right rectangles are neural networks' layers and rounded rectangles designate features. Layer f_ϕ^{Emb} maps the input features (time τ , KC u , and performance y , described in the text below) into embedding vectors. Layers f_ϕ^s and f_ϕ^z output the parameters of the variational posterior distribution. These are all part of the inference networks and parameterized by ϕ (surrounded by the grey box). The orange arrow is only applicable for inference on entire learning histories. Blue arrows represent the prediction stage, where during prediction M samples are drawn from the predicted distribution based on $\mu_{z_{n+1}}$ and $\log \sigma_{z_{n+1}}$.

generalizes the unimodal Gaussian prior and provides multimodality¹¹. By assuming that the observed data arises from a mixture of Gaussians, determining the category of a data point becomes equivalent to identifying the mode of the latent distribution from which the data point originates. This approach allows us to partition our latent space into distinct categories. With these discrete variables, it is no longer possible to directly apply the reparameterization trick. To solve this inference challenge, we modify the standard VAE architecture by incorporating the Gumbel-Softmax trick (Jang et al., 2016)¹². We employ an LSTM network, taking history embeddings $v_{1:n}$ as inputs and generating one of C category labels through the Gumbel-Softmax technique, denoted as $w = \text{LSTM}(v_{1:n}) \in \text{Cat}(\pi)$. Here $\text{Cat}(\pi)$ represents the categorical distribution with probabilities $\pi \in \Delta^C$. Simultaneously, we capture hidden states at each time point as $h_{z_{1:n}}$. Subsequently, we utilize an MLP to process both the category label and hidden states as input, producing the mean $\mu_{s_n} \in \mathbb{R}^4$ and log-variance $\log \sigma_{s_n} \in \mathbb{R}^4$ of latent states s_n for each time point. Table B.3 presents an overview of the PSI-KT model architecture and hyperparameters used for all experiments.

11: We therefore favor a mixture prior over a mixture of encoders, as it introduces multimodality in the latent space without duplicating the full inference network.

12: Direct sampling of the mixture component would require gradient estimators for discrete variables. The Gumbel-Softmax provides a continuous relaxation of the categorical variable, enabling low-variance gradient-based optimization of the ELBO.

Table B.3: PSI-KT architecture and hyperparameters. $\text{FC}(a, b)$ represents a fully connected layer with input dimension a and output dimension b ; K represents the number of KCs, different across datasets; C represents the number of categories in the mixture of Gaussians for s (we use $C = 10$ in our experiments); the semicolon ; separates connected layers, while the slash / separates the layer architecture for inference on entire histories from the continual learning set-up, where different.

	f_{ϕ}^{Emb}	f_{ϕ}^z	f_{ϕ}^s
Hidden Layers	$\text{FC}(32, 16)$ $\text{LeakyReLU}(0.2)$ $\text{FC}(16, 16)$	$\text{LSTM}(16, 32) / \text{FC}(16, 32)$ $\text{FC}(32, 16)$ $\text{LeakyReLU}(0.2)$ $\text{FC}(16, 16)$ $\text{LeakyReLU}(0.2)$ $\text{FC}(16, K)$ $\text{FC}(16, K)$	$\text{FC}(16, 32)$ $\text{FC}(32, 16)$ $\text{LeakyReLU}(0.2)$ $\text{FC}(16, 64)$ $\text{LeakyReLU}(0.2)$ $\text{GumbelSoftmax}(\text{FC}(64, C))$ $\text{FC}(32 + C, 64)$ $\text{LeakyReLU}(0.2)$ $\text{FC}(64, 16)$ $\text{LeakyReLU}(0.2)$ $\text{FC}(16)$ $\text{FC}(64, 4)$
Inputs & Dim	KC Emb & 16 Perf Emb & 16	v_n & 16	v_n & 16
Outputs	v_n	$\mu_{z_n}, \log \sigma_{z_n}$	$\mu_{s_n}, \log \sigma_{s_n}$

B.4 Prediction and generalization experiments details

B.4.1 Within-learner prediction results and training hyperparameters

In our prediction experiments, we use supervised training: for each learner, the first 10 interactions are used for training and the next 10 for testing, with 20% of learners held out for validation. We train all models with Adam (Kingma and Ba, 2014), an initial learning rate of 0.005, gradient clipping at 10.0, linear learning-rate decay by half every 200 epochs, and batch size 32.

In Figure 5.5, we show average accuracy curves. Detailed accuracy, F1-score, and standard deviations over 5 random seeds are reported in Table B.4 and Table B.5. While baseline models traditionally predict a single future time point from all available history, we argue that short-term prediction alone is insufficient for capturing long-term learner performance trends, which are essential for personalized material recommendation. It is also often unrealistic to assume immediate access to ground-truth data for such predictions. Therefore, we predict 10 time points into the future, using the predicted performances as inputs for each step. In other words, instead of using ground-truth data, if the model can predict $\hat{y}_n$ based on all previous training data $y_{n' < n}$, we incorporate the predicted performance along with the historical data $[y_{n' < n}; \hat{y}_n]$ to predict $\hat{y}_{n+1}$.

In our evaluations, we focus on prediction and generalization for small learner cohorts (100–1,000), reflecting the limited data often available in educational settings. This setting is important for practical ITS deployment: it reduces experimental exposure, mitigates cold-start issues, and supports classroom-scale use. To enable adaptive guidance and long-term learner assessment, we always predict the next 10 interactions. For a fair comparison with deep learning baselines, we also evaluate on an expanded dataset of over 1,000 learners after filtering out those with fewer than 50 interactions, reserving 20% for validation. Average accuracy, learner counts, and model parameter counts are reported in Table B.6.

Table B.4: Accuracies for within-learner prediction across numbers of learners (mean $\pm$ SEM across random seeds).

Dataset	# Learners	HLR	PPE	DKT	DKTF	HKT	AKT	GKT	QIKT	PSI-KT
Assist12	100	.54 _{.03}	.65 _{.01}	.65 _{.03}	.60 _{.01}	.55 _{.01}	.67 _{.02}	.63 _{.03}	.63 _{.03}	.68 _{.02}
	200	.55 _{.02}	.63 _{.03}	.66 _{.02}	.62 _{.01}	.58 _{.01}	.67 _{.02}	.61 _{.02}	.66 _{.02}	.70 _{.02}
	300	.55 _{.01}	.66 _{.01}	.67 _{.01}	.62 _{.00}	.58 _{.01}	.69 _{.02}	.65 _{.02}	.65 _{.02}	.71 _{.01}
	400	.55 _{.01}	.65 _{.01}	.68 _{.01}	.63 _{.01}	.60 _{.02}	.67 _{.03}	.63 _{.02}	.66 _{.01}	.71 _{.01}
	500	.55 _{.01}	.64 _{.01}	.67 _{.01}	.63 _{.01}	.59 _{.03}	.67 _{.02}	.63 _{.02}	.65 _{.02}	.70 _{.01}
	1,000	.54 _{.00}	.65 _{.00}	.68 _{.01}	.63 _{.01}	.60 _{.01}	.70 _{.02}	.64 _{.01}	.64 _{.01}	.70 _{.01}
Assist17	100	.45 _{.01}	.53 _{.02}	.57 _{.02}	.53 _{.03}	.52 _{.03}	.56 _{.02}	.56 _{.04}	.58 _{.02}	.63 _{.02}
	200	.45 _{.01}	.53 _{.02}	.57 _{.02}	.54 _{.02}	.54 _{.01}	.55 _{.01}	.56 _{.02}	.60 _{.02}	.63 _{.01}
	300	.46 _{.01}	.53 _{.01}	.57 _{.02}	.55 _{.02}	.55 _{.02}	.56 _{.04}	.58 _{.02}	.61 _{.01}	.63 _{.01}
	400	.45 _{.01}	.53 _{.01}	.56 _{.01}	.57 _{.02}	.56 _{.02}	.56 _{.02}	.58 _{.02}	.61 _{.01}	.64 _{.00}
	500	.46 _{.01}	.53 _{.00}	.60 _{.01}	.58 _{.01}	.54 _{.01}	.56 _{.02}	.58 _{.01}	.61 _{.02}	.63 _{.01}
	1,000	.44 _{.01}	.55 _{.01}	.60 _{.01}	.57 _{.01}	.57 _{.01}	.61 _{.01}	.60 _{.01}	.63 _{.01}	.64 _{.00}
Junyi15	100	.55 _{.02}	.66 _{.03}	.79 _{.03}	.78 _{.01}	.63 _{.02}	.81 _{.02}	.78 _{.02}	.81 _{.02}	.83 _{.02}
	200	.57 _{.01}	.65 _{.03}	.79 _{.01}	.78 _{.02}	.68 _{.03}	.80 _{.01}	.80 _{.01}	.80 _{.01}	.84 _{.01}
	300	.56 _{.02}	.65 _{.03}	.81 _{.01}	.79 _{.01}	.70 _{.01}	.81 _{.01}	.78 _{.02}	.81 _{.01}	.85 _{.01}
	400	.61 _{.02}	.65 _{.02}	.81 _{.01}	.80 _{.02}	.69 _{.02}	.82 _{.02}	.75 _{.02}	.80 _{.01}	.85 _{.01}
	500	.61 _{.01}	.67 _{.02}	.82 _{.01}	.80 _{.02}	.70 _{.01}	.82 _{.02}	.78 _{.02}	.81 _{.01}	.85 _{.01}
	1,000	.59 _{.01}	.66 _{.02}	.81 _{.01}	.81 _{.00}	.69 _{.01}	.82 _{.01}	.79 _{.02}	.83 _{.01}	.85 _{.01}

Table B.5: F1 scores for within learner prediction across learner numbers (mean $\pm$ SEM across random seeds.)

Dataset	# Learners	HLR	PPE	DKT	DKTF	HKT	AKT	GKT	QIKT	PSI-KT
Assist12	100	.59 _{.02}	.77 _{.01}	.77 _{.03}	.72 _{.01}	.64 _{.01}	.79 _{.02}	.76 _{.01}	.73 _{.03}	.80 _{.01}
	200	.60 _{.02}	.74 _{.03}	.78 _{.02}	.73 _{.01}	.68 _{.01}	.76 _{.02}	.74 _{.02}	.77 _{.02}	.82 _{.01}
	300	.59 _{.02}	.77 _{.01}	.79 _{.01}	.74 _{.00}	.69 _{.01}	.73 _{.03}	.76 _{.02}	.77 _{.01}	.83 _{.01}
	400	.60 _{.02}	.77 _{.01}	.79 _{.01}	.74 _{.01}	.70 _{.03}	.73 _{.03}	.75 _{.01}	.76 _{.01}	.83 _{.01}
	500	.60 _{.01}	.76 _{.01}	.79 _{.01}	.74 _{.01}	.64 _{.10}	.74 _{.02}	.75 _{.02}	.76 _{.01}	.82 _{.01}
	1,000	.60 _{.01}	.76 _{.00}	.79 _{.00}	.74 _{.01}	.71 _{.01}	.73 _{.02}	.76 _{.01}	.76 _{.01}	.82 _{.00}
Assist17	100	.45 _{.01}	.44 _{.01}	.42 _{.02}	.40 _{.03}	.42 _{.03}	.40 _{.02}	.40 _{.02}	.41 _{.02}	.48 _{.03}
	200	.45 _{.01}	.44 _{.01}	.40 _{.03}	.42 _{.01}	.43 _{.01}	.44 _{.01}	.41 _{.02}	.43 _{.02}	.47 _{.04}
	300	.45 _{.01}	.45 _{.02}	.40 _{.02}	.41 _{.01}	.42 _{.03}	.45 _{.03}	.42 _{.01}	.44 _{.03}	.46 _{.03}
	400	.44 _{.01}	.44 _{.02}	.41 _{.01}	.42 _{.02}	.43 _{.02}	.45 _{.03}	.42 _{.02}	.45 _{.01}	.47 _{.03}
	500	.46 _{.01}	.45 _{.01}	.40 _{.01}	.42 _{.00}	.40 _{.10}	.45 _{.02}	.43 _{.02}	.45 _{.01}	.47 _{.02}
	1,000	.44 _{.01}	.44 _{.02}	.40 _{.01}	.43 _{.00}	.43 _{.03}	.46 _{.02}	.43 _{.02}	.47 _{.01}	.47 _{.04}
Junyi15	100	.53 _{.02}	.70 _{.03}	.88 _{.02}	.87 _{.01}	.75 _{.03}	.89 _{.01}	.87 _{.01}	.89 _{.01}	.92 _{.01}
	200	.54 _{.02}	.71 _{.02}	.88 _{.01}	.87 _{.01}	.80 _{.02}	.88 _{.01}	.88 _{.01}	.89 _{.01}	.91 _{.01}
	300	.53 _{.02}	.71 _{.02}	.89 _{.01}	.88 _{.01}	.80 _{.01}	.90 _{.01}	.87 _{.02}	.89 _{.01}	.92 _{.01}
	400	.52 _{.02}	.72 _{.03}	.89 _{.01}	.88 _{.01}	.80 _{.01}	.90 _{.01}	.87 _{.02}	.89 _{.01}	.92 _{.02}
	500	.53 _{.01}	.70 _{.02}	.89 _{.01}	.88 _{.01}	.74 _{.08}	.88 _{.01}	.86 _{.01}	.89 _{.01}	.92 _{.01}
	1,000	.52 _{.01}	.71 _{.02}	.90 _{.01}	.89 _{.00}	.80 _{.02}	.90 _{.01}	.85 _{.01}	.90 _{.01}	.93 _{.00}

While deep models benefit from large datasets, they remain challenging for real-time deployment due to their parameter size. Moreover, PSI-KT performs strongly on small cohorts, whereas baselines require data from at least 60k learners to reach comparable performance.

B.4.2 Between-learner fine-tuning hyperparameters

For between-learner generalization, we use the best validation-performing within-learner prediction model from [Section B.4.1](#), trained on 100 learners. We then evaluate it on 100 randomly selected learners excluded from training and validation. Without fine-tuning, the pre-trained model is applied directly to these unseen learners, using their first 10 interactions to predict the next 10; results are reported in [Table 5.2](#). In the fine-tuning experiment, we perform fine-tuning for each model using a batch size of 32. Additionally, we also set aside 20% of the learners as a validation set during this process to save the model that achieves the highest accuracy after fine-tuning. For baseline models, HLR, PPE, HKT, which are comprised entirely of learner-independent and KC-dependent parameters, we conduct fine-tuning for all of these parameters. In this scenario, we use the pre-trained models as the initial weight values for the fine-tuning process. Conversely, for models DKT, DKTF, AKT, we perform fine-tuning specifically on their KC embedding parameters and

Table B.6: Accuracy score in within-learner prediction with all learners in each dataset (mean $\pm$ SEM across random seeds).

Dataset	# Learners	HLR	PPE	DKT	DKTF	HKT	AKT	GKT	QIKT	PSI-KT
Assist17	1,358	.46 _{.00}	.55 _{.00}	.58 _{.01}	.55 _{.01}	.57 _{.01}	.60 _{.01}	.60 _{.01}	.61 _{.01}	.64 _{.00}
Assist12	9,954	.44 _{.02}	.47 _{.00}	.69 _{.00}	.66 _{.00}	.66 _{.00}	.68 _{.01}	.70 _{.00}	.68 _{.00}	.70 _{.01}
Junyi15	62,124	.65 _{.01}	.71 _{.01}	.85 _{.00}	.85 _{.00}	.84 _{.01}	.86 _{.01}	.86 _{.02}	.86 _{.00}	.85 _{.01}

Table B.7: Continual learning accuracy. We report accuracy in predicting 10 subsequent outcomes. # Data indicate the number of interactions from each learner for training.

Dataset	# Data	10	20	30	40	50	60	70	80	90
Assist12	HLR	.54 _{.03}	.57 _{.08}	.58 _{.08}	.59 _{.09}	.57 _{.10}	.56 _{.07}	.54 _{.07}	.55 _{.06}	.57 _{.08}
	PPE	.65 _{.01}	.55 _{.07}	.53 _{.07}	.52 _{.08}	.54 _{.06}	.57 _{.06}	.59 _{.06}	.61 _{.06}	.69 _{.04}
	DKT	.65 _{.03}	.66 _{.07}	.64 _{.06}	.68 _{.05}	.69 _{.04}	.66 _{.05}	.66 _{.05}	.68 _{.03}	.65 _{.01}
	DKTF	.60 _{.01}	.67 _{.04}	.65 _{.04}	.64 _{.04}	.66 _{.03}	.62 _{.06}	.61 _{.04}	.63 _{.02}	.63 _{.02}
	HKT	.55 _{.01}	.56 _{.05}	.62 _{.04}	.62 _{.05}	.63 _{.02}	.60 _{.02}	.61 _{.03}	.61 _{.02}	.62 _{.02}
	AKT	.67 _{.02}	.66 _{.04}	.62 _{.04}	.61 _{.04}	.61 _{.05}	.65 _{.02}	.62 _{.02}	.61 _{.02}	.63 _{.02}
	GKT	.65 _{.02}	.62 _{.02}	.62 _{.01}	.64 _{.05}	.65 _{.04}	.65 _{.03}	.66 _{.06}	.65 _{.05}	.65 _{.05}
	QIKT	.70 _{.02}	.63 _{.01}	.64 _{.02}	.63 _{.01}	.62 _{.03}	.62 _{.01}	.62 _{.02}	.62 _{.02}	.63 _{.01}
	PSI-KT	.68 _{.02}	.70 _{.03}	.68 _{.03}	.72 _{.03}	.75 _{.02}	.73 _{.03}	.74 _{.02}	.74 _{.02}	.74 _{.02}
Assist17	HLR	.45 _{.01}	.46 _{.07}	.45 _{.07}	.53 _{.06}	.55 _{.08}	.57 _{.06}	.55 _{.06}	.55 _{.04}	.54 _{.03}
	PPE	.53 _{.02}	.52 _{.06}	.52 _{.06}	.52 _{.07}	.52 _{.06}	.52 _{.05}	.51 _{.05}	.54 _{.04}	.56 _{.04}
	DKT	.57 _{.02}	.52 _{.05}	.52 _{.05}	.52 _{.06}	.59 _{.04}	.57 _{.05}	.60 _{.04}	.63 _{.02}	.59 _{.03}
	DKTF	.53 _{.03}	.58 _{.05}	.54 _{.05}	.58 _{.05}	.58 _{.04}	.55 _{.05}	.56 _{.05}	.56 _{.04}	.61 _{.02}
	HKT	.52 _{.03}	.57 _{.04}	.60 _{.03}	.60 _{.03}	.62 _{.02}	.61 _{.03}	.61 _{.02}	.60 _{.02}	.61 _{.02}
	AKT	.56 _{.02}	.53 _{.05}	.52 _{.06}	.54 _{.04}	.53 _{.04}	.53 _{.02}	.50 _{.03}	.51 _{.03}	.57 _{.02}
	GKT	.63 _{.02}	.59 _{.05}	.54 _{.04}	.60 _{.04}	.56 _{.03}	.54 _{.02}	.57 _{.02}	.58 _{.02}	.58 _{.03}
	QIKT	.65 _{.02}	.58 _{.03}	.59 _{.03}	.56 _{.05}	.58 _{.03}	.56 _{.02}	.58 _{.02}	.58 _{.01}	.56 _{.02}
	PSI-KT	.63 _{.02}	.62 _{.04}	.65 _{.04}	.60 _{.05}	.60 _{.05}	.62 _{.05}	.62 _{.04}	.62 _{.04}	.64 _{.03}
Junyi15	HLR	.55 _{.02}	.43 _{.06}	.42 _{.06}	.44 _{.05}	.60 _{.04}	.63 _{.04}	.63 _{.03}	.63 _{.04}	.64 _{.03}
	PPE	.66 _{.03}	.67 _{.06}	.64 _{.06}	.64 _{.05}	.62 _{.04}	.63 _{.05}	.60 _{.05}	.60 _{.03}	.61 _{.03}
	DKT	.79 _{.03}	.80 _{.04}	.78 _{.04}	.76 _{.05}	.77 _{.04}	.75 _{.04}	.84 _{.04}	.73 _{.02}	.74 _{.01}
	DKTF	.78 _{.01}	.74 _{.05}	.77 _{.05}	.74 _{.06}	.71 _{.05}	.71 _{.04}	.74 _{.03}	.71 _{.03}	.72 _{.02}
	HKT	.63 _{.02}	.63 _{.08}	.69 _{.07}	.67 _{.07}	.70 _{.04}	.73 _{.04}	.73 _{.03}	.79 _{.02}	.84 _{.02}
	AKT	.81 _{.02}	.79 _{.04}	.78 _{.05}	.79 _{.04}	.75 _{.04}	.75 _{.03}	.76 _{.03}	.74 _{.02}	.74 _{.03}
	GKT	.82 _{.01}	.80 _{.02}	.78 _{.03}	.78 _{.03}	.79 _{.04}	.79 _{.03}	.79 _{.03}	.79 _{.02}	.80 _{.02}
	QIKT	.84 _{.00}	.80 _{.02}	.80 _{.05}	.78 _{.03}	.78 _{.04}	.81 _{.03}	.80 _{.02}	.78 _{.01}	.85 _{.02}
	PSI-KT	.83 _{.02}	.81 _{.04}	.81 _{.04}	.80 _{.04}	.77 _{.06}	.81 _{.04}	.82 _{.04}	.83 _{.03}	.84 _{.03}

the last fully connected layer within the neural network, while keeping the remaining layers frozen during the fine-tuning process.

B.4.3 Continual-learning results

In this experiment, we randomly select 100 learners using five different random seeds, and collect their first 100 interactions. Initially, the models are trained using only the first 10 interactions, following the same setup as in the within-learner prediction experiment. Following the initial training, we continuously integrate new interaction data into the training process, introducing one interaction at a time for each learner. The model iteratively predicts the subsequent 10 performances. This simulates a common real-world scenario, where learners continually interact with existing or even new KCs.

In the objective function ELBO shown in Equation (5.14), all historical information up to time t_n has already been fully encoded into the variational parameters ϕ_n . Additionally, to allow for the possibility of learners encountering a new KC during their learning journey, we allow for optimization over the KC parameters in the generative model. As a result, when new interaction data $\mathcal{I} = (x_{n+1}, y_{n+1}, t_{n+1})^{1:L}$ becomes available at time t_{n+1} , we use the new data to update both the inference model parameters ϕ_{n+1} and the generative model parameters U and M which are related to KCs.

For the baseline models, which are designed to predict performances based on fixed learning histories, there is no need to update the model

parameters for each new data point from each learner individually. Instead, when new interaction data, denoted as $\mathcal{I} = (x_{n+1}, y_{n+1}, t_{n+1})^{1:L}$, becomes accessible at time t_{n+1} , we update all the model parameters using all the interaction data collected up to that point, referred to as $\mathcal{H}_{1:n+1}$. This update is performed through 10 gradient descent processes. It is important to note that we do not include an additional validation set to determine when to stop training each model separately. Instead, we aim for a fair comparison among all models, ensuring that they are trained on equal footing with the same limited data and resources available to them.

B.5 Learner-specific representations analysis

In this experiment, we examine temporal latent features (learner representations) derived from baseline models. When considering baseline models, it is noteworthy that only DKT, DKTF, AKT, QIKT, incorporate learner-specific temporal embedding vectors. While HKT utilizes temporal embeddings, all these embeddings originate from global parameters associated with KCs, rendering them non-learner-specific. Consequently, our comparative analysis focuses exclusively on PSI-KT compared with DKT, DKTF, AKT, QIKT. We initially present comprehensive results in Table B.8, complementing Table 5.3 from Section 5.4.4, wherein only the results from the best-performing baseline models are displayed. Subsequent sections will detail the experimental setups and metrics employed.

B.5.1 Experimental setup for specificity

In personalized learning, we assume each learner has a unique cognitive profile shaped by past experiences and educational contexts. Our first step is to connect learner representations s_n^ℓ with these inherent learner-specific cognitive traits, i.e., the *specificity* of learners given corresponding representations.

To quantify specificity, we employ mutual information, denoted as $\text{MI}(s; \ell) := H(s) - H(s | \ell)$ ¹³ among all learners, as a measure of the information shared between learner identities and learner representations. The detailed computation of the metric $\text{MI}(s; \ell)$ is outlined as follows:

$$\begin{aligned}
 \text{MI}(s; \ell) &= H(s) - H(s | \ell) \\
 &= - \int p(s) \log p(s) - \frac{1}{L} \sum_{\ell} \int p(s | \ell) \log p(s, | \ell) \\
 &= \frac{1}{2} (D(1 + \log 2\pi) + \log |\Sigma_s|) - \frac{1}{2L} \sum_{\ell} (D(1 + \log 2\pi) + \log |\Sigma_{s\ell}|) \\
 &= \frac{1}{2} (\log |\Sigma_s| - \frac{1}{L} \sum_{\ell} \log |\Sigma_{s\ell}|). \tag{B.15}
 \end{aligned}$$

Here Σ_s and $\Sigma_{s\ell}$ are the covariance matrices obtained from fitting learner representations from all L learners or, respectively, a single learner with a Gaussian distribution, and D is the dimensionality of learner representations. In experiments, we begin by randomly selecting 1,000 learners from each dataset and then extracting their first 50 interactions for training. To determine when to stop training effectively, we set aside

13: For a D -dimensional Gaussian $\mathcal{N}(\mu, \Sigma)$, the differential entropy is $\frac{1}{2}(D(1 + \log 2\pi) + \log |\Sigma|)$. The $D(1 + \log 2\pi)$ terms cancel in the difference, leaving only the log-determinant terms in Equation (B.15).

Table B.8: Specificity, consistency, and disentanglement of the learned representations across datasets. Arrows indicate the preferred direction of each metric. Higher specificity and disentanglement indicate that learner traits capture more learner-specific information while remaining distinct from learner identity; lower Consistency⁻¹ indicates greater stability of trait representations across interactions from the same learner.

Metric	Dataset	DKT	DKTF	AKT	QIKT	PSI-KT
Specificity $MI(s; \ell) \uparrow$	Assist12	8.83	6.62	6.45	2.47	<u>8.40</u>
	Assist17	8.08	7.50	10.05	2.95	<u>9.98</u>
	Junyi15	12.75	<u>13.50</u>	13.34	4.09	14.37
Consistency ⁻¹ $\mathbb{E}_{\ell_{\text{sub}}} MI(s^\ell; \ell_{\text{sub}}) \downarrow$	Assist12	14.13	12.24	20.15	<u>8.35</u>	7.48
	Assist17	14.95	13.11	24.47	<u>6.35</u>	6.35
	Junyi15	13.10	17.81	22.15	<u>7.66</u>	5.00
Disentanglement $D_{\text{KL}}(s \ell) \uparrow$	Assist12	-1.64	0.38	-8.17	<u>2.31</u>	7.42
	Assist17	-3.01	-0.44	-9.81	<u>0.56</u>	8.39
	Junyi15	-0.62	<u>4.96</u>	-6.65	1.57	11.49

14: Baseline models use 16-dimensional representations vs. PSI-KT’s 4. Higher dimensionality provides strictly more capacity to encode learner identity, so this comparison is already favorable to the baselines.

a validation set of 20% of learners, which amounts to 200 learners in our case. This setup mirrors our approach in the prediction experiment. The metric $MI(s; \ell)$ is calculated for the learners in the training set. Since our goal here is to evaluate the model’s capacity to distill representations s^ℓ that uniquely identify learners, there is no need for a test set. Note that the baseline models have higher-dimensional learner representations (16 dimensions in our experiments¹⁴), potentially allowing them to capture more information.

B.5.2 Experimental setup for consistency

We proceed with a supplementary *consistency* analysis to determine, among the shared information quantified in specificity, whether the learner representations capture intricate learner attributes or merely reflect transient dynamic fluctuations. In the experiment, we split the interaction data of each learner into five separate groups, i.e., subsets. Each subset contains 30 interactions. These specific sizes were chosen to ensure we have both enough learners for robust training and enough interactions in subsets to estimate covariance matrices for our metrics. We thus exclude learners who have engaged in fewer than 150 interactions.

To form subsets, we find out the average presentation time of each KC and assign the KCs to separate subsets, so that the overall average interaction time is as similar as possible across subsets. With this, we aim to wash out, to the extent possible with the limited amount of data, systematic biases in the partition induced by the dependence of learner representations on time.

The mutual information metric $\mathbb{E}_{\ell_{\text{sub}}} MI(s^\ell; \ell_{\text{sub}}) := \mathbb{E}_{\ell_{\text{sub}}} [H(s | \ell) - H(s | \ell_{\text{sub}})]$, employed in the consistency experiments, undergoes the a similar derivation process to Equation (B.15).

$$\begin{aligned}
 \mathbb{E}_{\ell_{\text{sub}}} MI(s^\ell; \ell_{\text{sub}}) &= \frac{1}{L} \sum_{\ell} (H(s | \ell) - \frac{1}{5} \sum_{\ell_{\text{sub}}} H(s | \ell_{\text{sub}})) \\
 &= \frac{1}{L} \sum_{\ell} \left(-\mathbb{E} [\log \mathcal{N}(\mu_{s^\ell}, \Sigma_{s^\ell}^2)] + \frac{1}{5} \sum_{\ell_{\text{sub}}} \mathbb{E} [\log \mathcal{N}(\mu_{s^{\ell_{\text{sub}}}}, \Sigma_{s^{\ell_{\text{sub}}}}^2)] \right) \\
 &= \frac{1}{L} \sum_{\ell} (\log |\Sigma_{s^\ell}| - \frac{1}{5} \sum_{\ell_{\text{sub}}} \log |\Sigma_{s^{\ell_{\text{sub}}}}|). \tag{B.16}
 \end{aligned}$$

We fit each sub-learner separately and quantify the divergence metric $\mathbb{E}_{\ell_{\text{sub}}} MI(s^\ell; \ell_{\text{sub}})$ between learners and their sub-learners. A lower value of $\mathbb{E}_{\ell_{\text{sub}}} MI(s^\ell; \ell_{\text{sub}})$ suggests a higher degree of consistency¹⁵, reflecting the difficulty in distinguishing between sub-learners and their corresponding overarching learners given learner representations. Overall, Table 5.3 shows that the learner representations of PSI-KT provide comparable learner specificity and superior consistency. The lower consistency

15: Lower $\mathbb{E}_{\ell_{\text{sub}}} MI(s^\ell; \ell_{\text{sub}})$ means the representation s cannot reliably distinguish *which subset* of a learner’s data it was inferred from—the representation is stable across different windows of the same learner’s history.

displayed by baseline models suggests that most of the representational capacity available in their higher-dimensional representations might be spent on capturing learner-unspecific characteristics of the training sample.

B.5.3 Experimental setup for disentanglement

With the insights gained from specificity, our analysis progresses to evaluating to what extent learner-specific representations, are disentangled. Disentanglement in machine learning has been characterized as the process of isolating and identifying distinct, independent, and informative generative factors of variation in the data (Bengio et al., 2013).

In our disentanglement experiments, we use the same setup for specificity, and compute the discrepancy $D_{\text{KL}}(s||\ell)$ based on 50 interactions of 1,000 learners. Our approach bears similarity to Kim and Mnih (2018), but we relax the unrealistic assumption of independent representations. In real-world scenarios, independence in cognitive attributes is not a priority. To assess how much information about learner identity is present in the covariance across the representation dimension, we use the divergence between full trait-vector entropy and diagonal learner-conditional trait-vector entropy.

The discrepancy $D_{\text{KL}}(s||\ell)$ is estimated by the full entropy of representations $H(s)$ and the diagonal elements of the covariance matrix in the conditional entropy $H(s|\ell)$

$$D_{\text{KL}}(s||\ell) := H(s)_{\text{full}} - H(s|\ell)_{\text{diag}} = \frac{1}{2} \left(\log |\Sigma_s| - \frac{1}{L} \sum_{\ell} \sum_{i=1}^D \log (\Sigma_s | \ell)_{ii} \right). \quad (\text{B.17})$$

Small non-diagonal elements of the covariance matrix in $H(s)$ suggest low cross-correlations. This can be interpreted as a form of disentanglement. As illustrated in the third row of Table 5.3, the representations from PSI-KT consistently exhibit a higher degree of disentanglement across all datasets.

B.5.4 Mixed-effect linear regressions in operational interpretability

Mixed-effects regression extends linear regression to handle data with hierarchical or clustered structures, such as repeated measurements from the same subjects (learners in our case). Taking one of our experiments as an example, we conduct regressions based on $y_n^\ell \sim \tilde{\mu}_n^{\ell,k} + (1|\text{learner})$. Here, y_n^ℓ represents the dependent variable and $\tilde{\mu}_n^{\ell,k}$ is a predictor variable at time t_n . Also, $(1|\text{learner})$ represents the random intercept associated with each learner. This random intercept accounts for variability between learners that cannot be explained by the fixed effect $\tilde{\mu}_n^{\ell,k}$. In other words, it accounts for the fact that different learners might have different biases in their responses, allowing us to capture a more robust estimate of the group-level effect.

For regression calculations, we use the models trained in the prediction experiments, as described in Section B.4. For consistent comparisons with specificity experiments, we opt for models trained on a group of 1,000 learners. This experiment goes beyond a simple sanity check (as

Table B.9: (regression coefficient, p -value) tuples for performance difference and initial performance across models and latents' dimensions. If there is no significant dimension in one model and dataset ($p > 0.05$), we show the dimension with the highest regression coefficient. Bold values indicate the dimension and the baseline model with the highest statistically significant linear relationship in one dataset, with which we show the regression results in Figure B.2 and Figure B.3.

Behavioural signature	Dataset	DKT	DKTF	AKT	PSI-KT
Performance difference	Assist12	(-.009, .643)	(.008, .736)	(-.009, .665)	(.300, <.001)
	Assist17	(-.008, .758)	(-.029, .304)	(-.001, .957)	(.556, <.001)
	Junyi15	(-.003, .853)	(.021, .771)	(.025, .064)	(.721, <.001)
Initial performance	Assist12	(.021, .078)	(.039, .008)	(.017, <.001)	(.544, <.001)
	Assist17	(.048, .004)	(.038, .030)	(.010, .030)	(3.705, <.001)
	Junyi15	(-.025, .034)	(.044, .021)	(.017, <.001)	(.921, <.001)

in Section B.5.1), so we use the testing data. This choice aligns with our objective of using operational interpretability to gain insights and inform future controlled experiments with unseen data. We use pre-trained models, specifically DKT, DKTF, AKT, and our PSI-KT model, selected based on their accuracy scores on the validation data.

To fairly compare with baseline models, we investigate whether any dimensions within the learner representations capture behaviors similar to our interpretable cognitive traits. Thus, we perform regression for each dimension within the learner representations of the baseline models. While Figure 5.7 in the main paper presents the regression results concerning the dimension featuring the most pronounced correlation among baseline models, we provide a complete list of dimensions that exhibit significant relationships with the behavioral data in Table B.9.

Performance decay and forgetting rate. To analyze the exponential decay of learner performances over time, we first show the relationship between performance decay Δy_n^ℓ and the raw time difference τ_n^ℓ , which is divided into 10 bins. We select bin centers to ensure an equal number of data points in each bin. This binning approach helps minimize the impact of outliers and ensures a balanced representation of data within each bin. Also, we show the relationship between decay Δy_n^ℓ and the time difference scaled by the corresponding forgetting rate α_n^ℓ at each time point, or each dimension of learner representations in the baseline models. We assume that if the forgetting rate α_n^ℓ is meaningful for each time interval and effectively controls the decay, then the standard error of behavior data Δy_n^ℓ within each bin should be smaller than the error of binning raw time differences. This indicates that the decay is better described as a function of $\alpha_n^\ell \tau_n^\ell$ than as a function of τ_n^ℓ alone. We also compute the standard error for each dimension of learner representations in the baseline models, and we show the dimension v_n^* with the lowest standard error in Figure B.2. Then, we perform mixed-effect regression over the exponential term $\exp(-\alpha_n^\ell \tau_n^\ell)$ (or $\exp(-v_n^* \tau_n^\ell)$ in baselines) to assess how well learner representations predict performance decay (as an exponential function). The results show that at least one dimension in the learner representations groups certain behavioral data and reduces the standard errors. However, none of these dimensions exhibit a statistically significant relationship with the behavioral data.

Initial performance and long-term mean. We conducted a mixed-effect regression analysis between the initial performance and the long-term mean, with the results presented in Figure B.3. These results indicate that, except DKT on the Assist12 dataset, at least one dimension in the baseline learner representations predicts initial performance. It is important to note that none of the dimensions exhibit a stronger effect compared to the identified trait $\tilde{\mu}_n^{\ell,k}$ in our PSI-KT. Additionally, we note

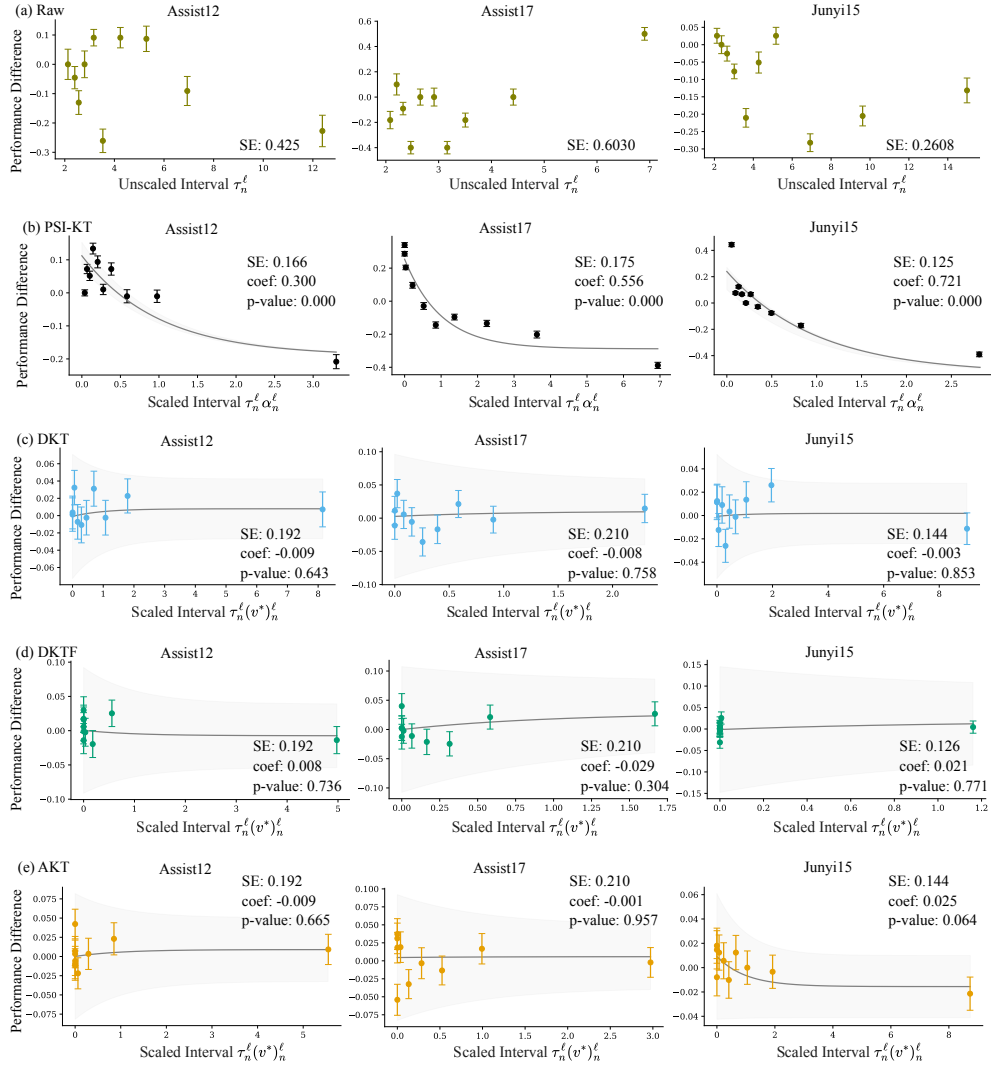


Figure B.2: The mixed-effect regressions of performance decay Δy_n^ℓ vs. scaled interval $\tau_n^\ell \alpha_n^\ell$ (scaled interval with the best dimension in baselines $\tau_n^\ell (v^*)^\ell$). The first row (a) shows the unscaled interval in the raw data. The aggregate standard error over 10 bins (SE), the regression coefficient (coef), and its p -value are reported in each panel.

that embedding dimensions in baselines are trained in a permutation-invariant manner, suggesting that these models can't route any particular generative factor of variation in the data (e.g. a behavioral signature) to a specific dimension.

Prerequisite transfer ability and learning variances. In our experiments, we sought to correlate two additional cognitive traits—transfer ability γ and learning volatility σ —with behavioral data. This task proved more complex than assessing the forgetting rate and long-term mean because assessing transfer ability requires reliable annotations of prerequisite relations and learning volatility can be connected to many unconstrained factors during the learning process.

Regarding transfer ability, our hypothesis posits that given the identified prerequisite KC i for KC j , a higher transfer ability γ_n^ℓ suggests an increased likelihood of correctly transitioning from one KC i to KC j . We calculate this transition probability $p(j^+ | i^+)_n^\ell$ by observing the frequency of correct responses to KC i followed by correct responses to KC j up to a certain time t_n . This implies that learners with greater transfer abilities are more likely to answer questions related to KC i correctly

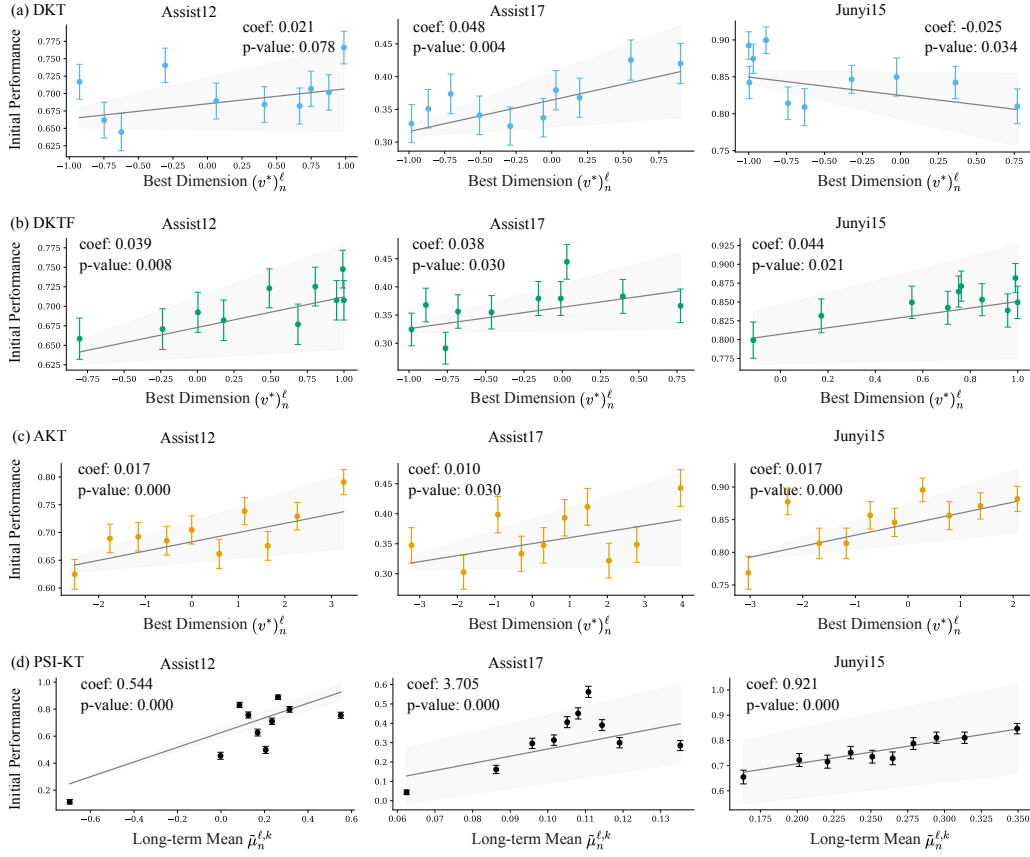


Figure B.3: The mixed-effect regressions of initial performance y_n^ℓ vs. long-term mean $\hat{\mu}_n^{\ell,k}$ (the best dimension $(v^*)^\ell_n$ in baselines). The aggregate standard error over 10 bins (SE), the regression coefficient (coef) and its p -value are reported in each panel.

after mastering KC i . However, this approach depends on accurately identifying prerequisite relationships between KCs. Therefore, we utilized the Junyi15 dataset, which includes expert-annotated and crowd-sourced prerequisite graphs, for our regression analysis. For learning volatility σ_n^ℓ , we connect the average squared mean $(\bar{\sigma}^\ell)^2$ for each learner with the variance in their performance $\text{Var}(y_{1:n}^\ell)$.

In Figure B.4, we present the results of our mixed-effect regression analyses. Each regression demonstrates a significant relationship. However, due to the sparsity of the expert-annotated graph, we do not have enough data to fit the regression model effectively. Thus we choose to use the crowd-sourcing graphs and consider the edge existence if the edge weight is above 0.5.

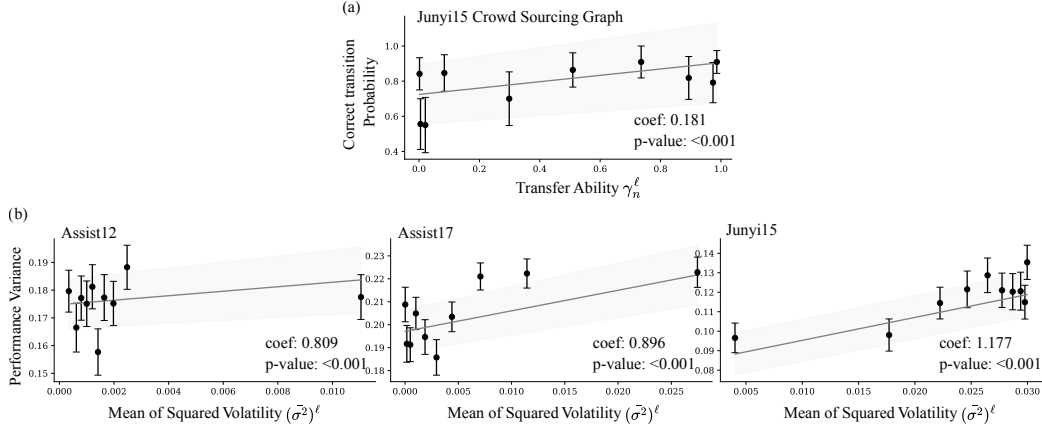


Figure B.4: The mixed-effect regressions of transfer ability γ with behavioral correct transition probability (a), and learning volatility σ with the variance in learning performances (b). We report the regression coefficient (coef) and its p -value in each panel, and each point illustrates the mean ($\pm$ SEM) of the corresponding decile.

B.6 Graph inference analysis

B.6.1 Details of the metrics for ground-truth graph comparison

Here we report comprehensive evaluations of the alignment of the inferred graphs with the human-annotated graphs in the Junyi15 dataset under different metrics.

As discussed in Section 5.4.4, the Junyi15 dataset provides two types of graph annotations - crowd-sourced similarity and prerequisite ratings (with 1,954 rated edges), as well as more sparse expert-annotated prerequisite relations (837 edges). We use the following metrics to compare graph representations learned by each model against these annotations:

1. **Mean Reciprocal Rank (MRR)** (Voorhees et al., 1999). We compare the inferred graph with the expert-annotated using the MRR, defined as $|K|^{-1} \sum_{i=1}^{|K|} (\text{rank}(i))^{-1}$, where K is the total number of KCs. We compute the rank of each expert-identified prerequisite relation $i \rightarrow k$ in the relevant sorted list of inferred probabilities $\{a^{jk}\}_{j=1}^K$ and take the harmonic average.
2. **Jaccard Similarity (JS)** is a classic measure of similarity between two sets, defined as the size of the intersection of set A and set B (i.e., the number of common elements) over the size of the union of set A and set B (i.e., the number of unique elements): $\text{JS}(A, B) = |A \cap B| / |A \cup B|$. Here we define the edge sets by thresholding weights at half the scale (0.5 for probability-scaled weights, 5 for the average of the 1 – 9 crowd-sourced rating).
3. **Negative Log-likelihood (nLL)** of edge weights given crowd-sourced annotations. The crowd-sourced annotations provide multiple 1 – 9 ratings per node pair. One set of annotations rates the strength of the directed prerequisite relations, whereas the other just rates the undirected similarity of the pair of nodes. We normalize the ratings from 0 to 1 and fit them with a Gaussian distribution. Then we compute the log-likelihood of the inferred edge probability under the Gaussian. The variance of the Gaussian accounts for inter-rater disagreements when comparing a model's inferred edge probability with the mean edge rating.

4. **Linear Regression Coefficient** between edge weights and the causal support (details are in [Section B.6.3](#)) from node i to node k on correctness of k if having correct interactions on i . We compute the causal support for transitions of every KC pair. However we remove the causal support of pairs of KCs that have only one transition in the dataset to avoid adding noise to our estimate.

B.6.2 Quantitative comparison results on the Junyi15 dataset

Note that the graphs of baselines are based on KC embeddings (as in [Section B.2](#)), and thus there is no edge directionality. For the baselines that have at least two embeddings for each KC, we can use to compute the directed edges, since one embedding for KC will end up in a symmetric structure adjacency matrix (DKT, DKTF, HKT, AKT). Thus, to conduct a fair comparison with the baseline models, we leniently compute edge weights based on every combination of KC embeddings. For example, in DKT, there are two embeddings $u^{k,0}, u^{k,1} \in \mathbb{R}^D$ representing incorrect interactions and correct interactions on KC k , respectively, and embeddings are shared across all learners¹⁶. We compute the edge weights a^{ik} based on two different combinations here, both $a^{ik} := u^{i,0\top} u^{k,1}$ and $a^{ik} := u^{i,1\top} u^{k,0}$, and report the graph with the best results. When extracting undirected graphs, we concatenate all KC embeddings to compute $a^{ik} := (u^{i,0} + u^{k,1})^\top (u^{i,0} + u^{k,1})$, in order to reflect all available information from all KC embeddings. In the case of baselines with a single embedding per KC, such as QIKT, or those using a parameterized undirected graph, like GKT, we allow their inferred graphs to be less accurate. This means that for these models, the presence of an edge between two KCs is deemed correct if there is a directed edge from either direction in annotated graphs, without the necessity for these edges to accurately indicate the directionality. We then compute the weights by min-max normalization. This normalization is necessary for computing the log-likelihood, where we also use a threshold of 0.5 to determine whether there is an edge when the comparison requires binary edges.

In [Table 5.5](#), we show the comparison of ground-truth prerequisite graphs and inferred graphs from our PSI-KT, and the best baseline models on the Junyi15 dataset under the different metrics. These results demonstrate that our inferred prerequisite graph outperforms others when compared with crowd-sourced and expert-annotated graphs under different metrics.

Here we show all of the comparison results, including a comparison of the similarity (undirected) graphs and the prerequisite (directed) graphs on four metrics in [Table B.10](#). We do not report MRR ranking scores for similarity graphs because the ground-truth similarity graph does not contain an expert-annotated version.

B.6.3 Causal support

Causal induction is the problem of inferring underlying causal structures from data. Here, we use a Bayesian framework (Griffiths and Tenenbaum, 2005, 2009) to infer a singular cause-and-effect relationship between all pairs of KCs, asking how performance on one node influences performance on another, and whether the strength of the causal relationship corresponds to our inferred prerequisite graph. In this context, we model

16: Similarity via dot product of embeddings is standard in word2vec (Mikolov et al., 2013) and graph embedding (Hamilton et al., 2017)—high inner product implies the representations share structure in the learned space.

Table B.10: Comparison between ground-truth graphs and inferred graphs in Junyi15 dataset. **pre** indicates evaluation against a prerequisite graph and **sim** evaluation against a similarity graph.

		DKT	DKTF	HKT	AKT	GKT	QIKT	PSI-KT
MRR ↑	expert pre	.0069	.0067	.0074	.0075	.0082	.0073	.0086
JS ↑	expert pre	1.46e-3	1.37e-3	1.47e-3	1.44e-3	1.46e-3	1.19e-3	1.86e-3
	crowd pre	4.60e-3	4.28e-3	4.66e-3	4.48e-3	3.44e-3	5.21e-4	9.48e-3
	crowd sim	5.90e-4	0.00	0.00	5.18e-4	3.43e-3	0.00	4.66e-3
nLL ↓	crowd pre	5.735	5.580	6.092	5.677	3.033	4.228	4.106
	crowd sim	6.598	4.039	4.042	9.100	9.028	10.622	2.352

the relationship between a candidate cause C and a candidate effect E (i.e., a pair of KCs), assuming an ever-present background cause B (i.e., the learner’s general ability and the influence of other nodes). The objectives are to determine the probability of a causal relationship between C and E , known as *causal support* (Equation (B.18)).

In our prerequisite graph scenario, we assume that if KC i is a prerequisite of KC k , the correctness on KC i contributes to the correctness on KC k . This implies the presence of a prerequisite relationship between KC i and KC k , signified by a causal link between their correctness levels. Consequently, for every pair of nodes, a candidate cause C corresponds to performance $y_n^i = 1$ at time t_n , and an effect E corresponds to $y_{n+1}^k = 1$ at time t_{n+1} , with inputs from all remaining nodes relegated to the background cause B .

When examining elemental causal induction, we adhere to the following two-step procedure: i) We establish the nature of the relationship through causal graphical models, and ii) we quantify the strength of the relationship, provided it exists, as a problem of inferring structural parameters. In the subsequent text, C and E variables are denoted using uppercase letters, while their specific instances are represented using lowercase letters. Specifically, c^+ and e^+ indicate the presence of the cause and effect (i.e., correct performance), whereas c^- and e^- signify their absence (i.e., incorrect performance).

Causal graphical models. Causal graphical models are a formalism for learning and reasoning about causal relationships (Glymour et al., 2019). Nodes in the graph represent variables, and directed edges represent causal connections between those variables. To identify whether a causal relationship exists between a pair of variables, we consider two directed graphs denoted Graph 0 $G_{C \rightarrow E} : B \rightarrow E$ and Graph 1 $G_{C \rightarrow E} : B \rightarrow E \leftarrow C$, as shown in Figure 5.8b. Thus, $G_{C \rightarrow E}$ represents the null hypothesis that there is no relationship between C and E (i.e., the effect E can be accounted for by background cause B), while $G_{C \rightarrow E}$ represents the alternative hypothesis that the causal relationship exists.

In our case, the cause C and the effect E are equivalent to KC i and KC k , respectively, for every pair of KCs. The process of inferring the underlying structure between KC i and KC k , whether the learners’ behavioral learning history $\mathcal{H}$ are generated by $G_{i \rightarrow k}$ or $G_{i \rightarrow k}$, can be cast in a Bayesian framework (Griffiths and Tenenbaum, 2005, 2009). *Causal support* quantifies the degree of evidence present in the data $\mathcal{H}$ that favors Graph 1 $G_{i \rightarrow k}$ over Graph 0 $G_{i \rightarrow k}$:

$$\text{support} = \log \frac{P(\mathcal{H} | G_{C \rightarrow E})}{P(\mathcal{H} | G_{C \rightarrow E})} = \log \frac{P(\mathcal{H} | G_{i \rightarrow k})}{P(\mathcal{H} | G_{i \rightarrow k})}. \quad (\text{B.18})$$

Intuitively, the joint presence of the cause and effect, i.e., correctness on KC i followed by correctness on KC k , offers support for a causal

link from node i to node k . Conversely, the absence of the cause, i.e., incorrectness on KC i but is followed by correctness on KC k , presents evidence against the notion that KC i is a prerequisite for KC k .

Causal support. Causal graphical models depict dependencies using conditional probabilities. Defining these probabilities entails parameterizing each edge, and this parameterization determines the functional expressions that govern causal relationships.

For Graph 0 $G_{i \rightarrow k}$ and Graph 1 $G_{i \rightarrow k}$, we define $P_0(y_{n+1}^k = 1 | B) = \omega_0$ and $P_1(y_{n+1}^k = 1 | y_n^i = 1) = \omega_1$ respectively. In other words, the probability of correctness on KC k given just background causes is ω_0 , and the probability of correctness on KC k given previous correctness on KC i is ω_1 ; and when both prerequisite KC i and background causes are present, they have independent opportunities to produce the effect.

For **Graph 0** $G_{C \rightarrow E}$, the sole parameter ω_0 denotes the likelihood of the effect being present given the background cause

$$P_0(e^+ | b^+; \omega_0) = \omega_0. \quad (\text{B.19})$$

The corresponding likelihood for the data $\mathcal{H}$ given Graph 0 $G_{i \rightarrow k}$ is accomplished by integrating over all possible parameters ω_0 with a uniform prior over ω_0 :

$$\begin{aligned} P(\mathcal{H} | G_{i \rightarrow k}) &= \int_0^1 P_0(\mathcal{H} | \omega_0, G_{i \rightarrow k}) P(\omega_0 | G_{i \rightarrow k}) d\omega_0 \\ &= \int_0^1 \omega_0^{N(e^+)} (1 - \omega_0)^{N(e^-)} d\omega_0 \\ &= \text{Beta}(N(e^+) + 1, N(e^-) + 1) \\ &= \text{Beta}(N(y_{n+1}^k = 1) + 1, N(y_{n+1}^k = 0) + 1). \end{aligned} \quad (\text{B.20})$$

Here $\text{Beta}()$ is the beta function, and $N(e^+)$ and $N(e^-)$ are the marginal frequencies of the effects.

17: Equation (B.21) is a *noisy-OR gate* (Pearl, 1994): each cause independently has a probability of activating the effect, and the effect fires if any cause fires. It enforces the constraint that C can only raise (never lower) the probability of E .

For **Graph 1** $G_{i \rightarrow k}$, the likelihood of the effect is given by¹⁷:

$$P_1(e^+ | b, c; \omega_0, \omega_1) = 1 - (1 - \omega_0)^b (1 - \omega_1)^c, \quad (\text{B.21})$$

where ω_0 again defines the influence of the background cause, and the additional parameter ω_1 defines the influence of the cause. Here b and c are binary, which means if cause C exists, then $c = 1$. We compute the likelihood of the data $P(\mathcal{H} | G_{i \rightarrow k})$ by integrating over parameters ω_0 and ω_1 . Each parameter value is defined by a prior probability, which when combined with the likelihood of the data, yields a joint posterior distribution over data and parameters for the structure. To determine the observed data likelihood for Graph 1 $G_{i \rightarrow k}$, we have

$$\begin{aligned} P(\mathcal{H} | G_{i \rightarrow k}) &= \int_0^1 \int_0^1 P_1(\mathcal{H} | \omega_0, \omega_1, G_{i \rightarrow k}) P(\omega_0, \omega_1 | G_{i \rightarrow k}) d\omega_0 d\omega_1 \\ &= \int_0^1 \int_0^1 \prod_{e,c} P_1(e | c, b^+; \omega_0, \omega_1)^{N(e,c)} P(\omega_0, \omega_1 | G_{i \rightarrow k}) d\omega_0 d\omega_1. \end{aligned} \quad (\text{B.22})$$

Here $N(e, c)$ represents the number of occurrences. To compute

$$\prod_{e,c} P_1(e | c, b^+; \omega_0, \omega_1)^{N(e,c)},$$

we iterate over all possible sets of (e, c) . Based on Equation (B.21), we get:

$$\begin{aligned} \prod_{e,c} P_1(e | c, b^+; \omega_0, \omega_1)^{N(e,c)} &= P_1(e^+ | c^+, b^+; \omega_0, \omega_1)^{N(e^+, c^+)} P_1(e^+ | c^-, b^+; \omega_0, \omega_1)^{N(e^+, c^-)} \\ &= (\omega_0 + \omega_1 - \omega_0 \omega_1)^{N(e^+, c^+)} \omega_0^{N(e^+, c^-)}. \end{aligned} \quad (\text{B.23})$$

While Equation (B.22) is not analytically tractable, it can be effectively approximated using Monte Carlo simulations. With uniform priors on ω_0 and ω_1 , a reliable estimation of $P(\mathcal{H} | G_{i \rightarrow k})$ can be obtained by generating m samples of ω_0 and ω_1 from a uniform distribution spanning the interval $[0, 1]$, followed by computation of:

$$\begin{aligned} P(\mathcal{H} | G_{i \rightarrow k}) &= \frac{1}{m} \sum_{i=1}^m P_1(\mathcal{H} | \omega_{0i}, \omega_{1i}, G_{i \rightarrow k}) \\ &= \frac{1}{m} \sum_{i=1}^m \prod_{e,c} P_1(e | c, b^+; \omega_{0i}, \omega_{1i})^{N(e,c)} \\ &= \frac{1}{m} \sum_{i=1}^m (\omega_{0i} + \omega_{1i} - \omega_{0i} \omega_{1i})^{N(e^+, c^+)} \omega_{0i}^{N(e^+, c^-)} \\ &= \frac{1}{m} \sum_{i=1}^m (\omega_{0i} + \omega_{1i} - \omega_{0i} \omega_{1i})^{N(y_{n+1}^k=1, y_n^i=1)} \omega_{0i}^{N(y_{n+1}^k=1, y_n^i=0)}. \end{aligned} \quad (\text{B.24})$$

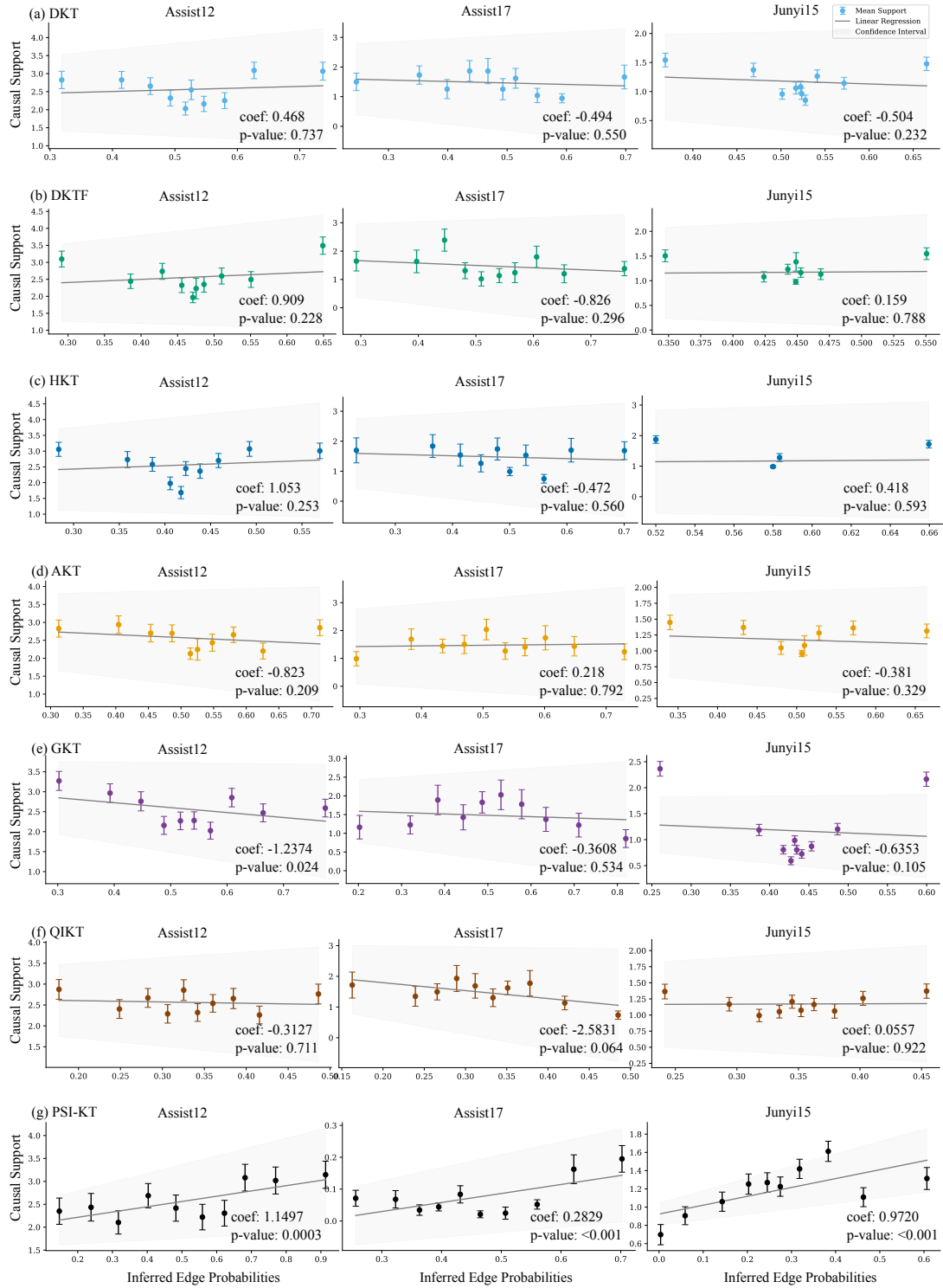


Figure B.5: Linear regressions relating causal support to the inferred edges for baseline models (a) DKT, (b) DKTF, (c) HKT, (d) AKT, (e) GKT, (f) QIKT, and (g) PSI-KT. The x-axis represents the normalized edge weights inferred by the respective baselines. The coefficient (coef) and its p -value are reported in the lower right of each panel.

B.7 Ablation study

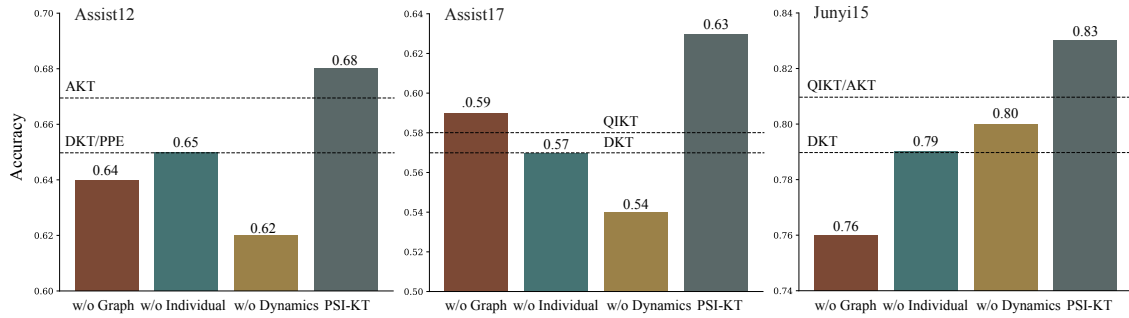


Figure B.6: Mean accuracy of PSI-KT vs. ablations of (a) the prerequisite structure (w/o graph), (b) individualized learner traits (w/o individual) and (c) time-dependent learner traits (w/o dynamics). Dashed lines indicate the accuracy of the two best-performing baselines.

To thoroughly examine the various elements of PSI-KT, including cognitive traits and the prerequisite graph, we executed three distinct ablation studies:

- Without the graph inference (w/o graph): We omit the graph inference process and the influence of prerequisite KCs on the long-term mean. Essentially, this approach treats each KC independently.
- Without individual cognitive traits (w/o individual): We alter the variational inference network in this scenario to produce a uniform distribution across all learners. This change effectively removes the consideration of individual differences in learners' cognitive traits.
- Without dynamic cognitive traits (w/o dynamics): We remove the dynamic transition distribution over the traits in the generative model. This assumes that each learner has static traits over time.

	Assist12	Assist17	junyi15
PSI-KT	.68 _{.017}	.63 _{.015}	.83 _{.015}
w/o Graph	-.04 _{.005}	-.04 _{.002}	-.07 _{.002}
w/o Individual traits	-.03 _{.002}	-.06 _{.006}	-.04 _{.001}
w/o Dynamic traits	-.06 _{.001}	-.09 _{.003}	-.03 _{.002}

Table B.11: The accuracy in three kinds of ablation study (mean $\pm$ SEM across random seeds). We show the accuracy gap compared with the complete PSI-KT model.

In Table B.11 and Figure B.6, we present the results of our three ablation studies. We observed that the contribution of prerequisite graphs, individualized traits, and dynamic traits varied across the datasets. These findings underscore the diversity inherent in educational datasets and simultaneously reinforce the effectiveness of our unified framework.

Additional Material for Chapter 6

C.

The appendix for Chapter 6 is organized as follows:

- ▶ **Section C.1** provides the full DreamerV3 world model architecture, including the complete RSSM equations, the training objective, the factorized discrete latent space structure, and implementation hyperparameters.
- ▶ **Section C.2** compares three formulations of information gain—actual, predicted (prospective), and retrospective—and explains why the predicted variant supports more effective exploration in tabular settings and why the retrospective approximation used in DreamerV3 is susceptible to aleatoric fixation.
- ▶ **Section C.3** addresses the nonstationarity of intrinsic reward signals and its implications for reinforcement learning algorithms, explaining why model-based architectures are essential and describing the failure modes we encountered in both tabular and Dreamer experiments.
- ▶ **Section C.4** describes the Blahut-Arimoto algorithm for exact one-step empowerment computation in the tabular setting and the factorized mutual information approximation used in DreamerV3, with a discussion of their limitations and potential extensions.

C.1 DreamerV3 architecture and implementation . . .	235
C.2 Information gain variants	237
C.3 Nonstationarity of intrinsic rewards	240
C.4 Blahut-Arimoto and factorized mutual information	243

C.1 DreamerV3 architecture and implementation

This appendix provides the full architecture description of the DreamerV3 world model used in Section 6.5, supplementing the overview in Section 6.5.1. We follow the implementation of Ferrao and Cunha (2025) but replace their CBET intrinsic reward with our novelty, information gain, and empowerment rewards, and use the partial pixel-based observation provided by MiniGrid rather than semantic state observations.

Recurrent State-Space Model (RSSM). The world model f_{wm} is a Recurrent State-Space Model with the following components:

$$\text{Sequence model: } h_t = f_\phi(h_{t-1}, s_{t-1}, a_{t-1}) \quad (\text{C.1})$$

$$\text{Encoder (posterior): } s_t \sim p_\phi(s_t | h_t, o_t) \quad (\text{C.2})$$

$$\text{Dynamics predictor (prior): } \hat{s}_t \sim p_\phi(\hat{s}_t | h_t) \quad (\text{C.3})$$

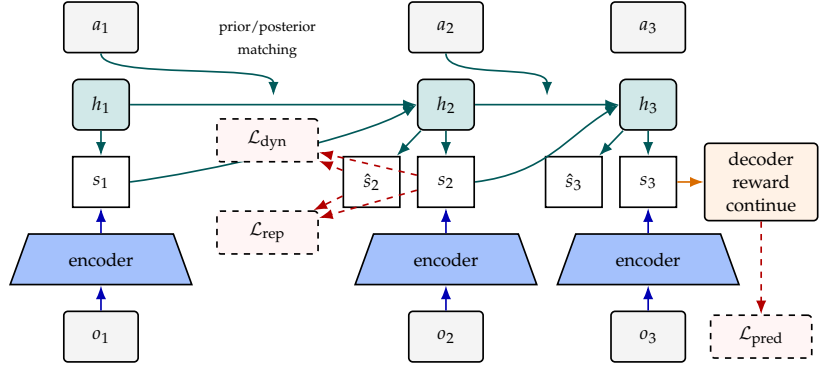
$$\text{Reward predictor: } \hat{r}_t \sim p_\phi(\hat{r}_t | h_t, s_t) \quad (\text{C.4})$$

$$\text{Continue predictor: } \hat{c}_t \sim p_\phi(\hat{c}_t | h_t, s_t) \quad (\text{C.5})$$

$$\text{Decoder: } \hat{o}_t \sim p_\phi(\hat{o}_t | h_t, s_t) \quad (\text{C.6})$$

The **sequence model** is a Gated Recurrent Unit (GRU) that maintains a deterministic recurrent state $h_t \in \mathbb{R}^d$, updated from the previous recurrent state, the previous stochastic latent, and the previous action. This recurrent component captures temporal dependencies and serves as a form of working memory, making the Markov property approximately hold even when the current observation alone is insufficient to identify

Figure C.1: Simplified illustration of DreamerV3 world model training. The recurrent state h_t aggregates past information, the encoder produces the posterior latent s_t from the current observation o_t , and the dynamics predictor produces the prior $\hat{s}_t$ from h_t alone. The prediction heads for observation, reward, and continuation contribute to $\mathcal{L}_{\text{pred}}$, while posterior-prior matching contributes to $\mathcal{L}_{\text{dyn}}$ and $\mathcal{L}_{\text{rep}}$.



the agent's location (e.g., when the agent faces a wall, the recurrent state encodes what it saw before turning).

The **encoder** maps the current observation o_t to a distribution over a discrete latent state s_t , conditioned on the current recurrent state h_t . This posterior distribution represents the agent's belief about the current state given all information available to it.

The **dynamics predictor** maps only the recurrent state h_t to a predicted distribution over s_t , without access to the current observation. This prior represents the agent's *prediction* of the current state based solely on memory (past observations and actions). The gap between encoder posterior and dynamics predictor prior is the core measure of model surprise and the basis for the information gain reward (Equation (6.11)).

The **reward predictor**, **continue predictor**, and **decoder** operate on the combined latent representation (h_t, s_t) to predict the immediate reward, the probability that the episode continues, and the current observation, respectively.

Training objective. The world model is trained by maximizing an evidence lower bound (ELBO) with three loss terms:

$$\mathcal{J}_{\text{pred}} = \mathbb{E} [\log p_{\phi}(o_t | h_t, s_t) + \log p_{\phi}(r_t | h_t, s_t) + \log p_{\phi}(c_t | h_t, s_t)], \quad (\text{C.7})$$

$$\mathcal{J}_{\text{dyn}} = \max(1, D_{\text{KL}}[\text{sg}(p_{\phi}(s_t | h_t, o_t)) \parallel p_{\phi}(\hat{s}_t | h_t)]), \quad (\text{C.8})$$

$$\mathcal{J}_{\text{rep}} = \max(1, D_{\text{KL}}[p_{\phi}(s_t | h_t, o_t) \parallel \text{sg}(p_{\phi}(\hat{s}_t | h_t))]), \quad (\text{C.9})$$

where $\text{sg}(\cdot)$ denotes the stop-gradient operator. The dynamics loss $\mathcal{J}_{\text{dyn}}$ trains the dynamics predictor to match the encoder (pulling the prior toward the posterior), while the representation loss $\mathcal{J}_{\text{rep}}$ trains the encoder to match the dynamics predictor (pulling the posterior toward the prior). The free-bits threshold of 1 nat prevents the KL losses from over-regularising the latent space.

Factorized discrete latent space. DreamerV3 represents s_t as a factorized discrete distribution: $L = 32$ independent categorical variables, each with $K = 32$ categories, yielding a total latent space of 32^{32} configurations. The encoder and dynamics predictor each output a 32×32 matrix of logits, from which each z_i is sampled independently.

The factorized structure enables efficient computation of all three intrinsic motivation signals. Novelty (Equation (6.9)) tracks per-factor counts and averages. Information gain (Equation (6.11)) computes and averages per-factor KL divergences. Empowerment (Equation (6.14)) computes and averages per-factor mutual information. All three measures therefore share a common unit of analysis—individual latent factors—ensuring

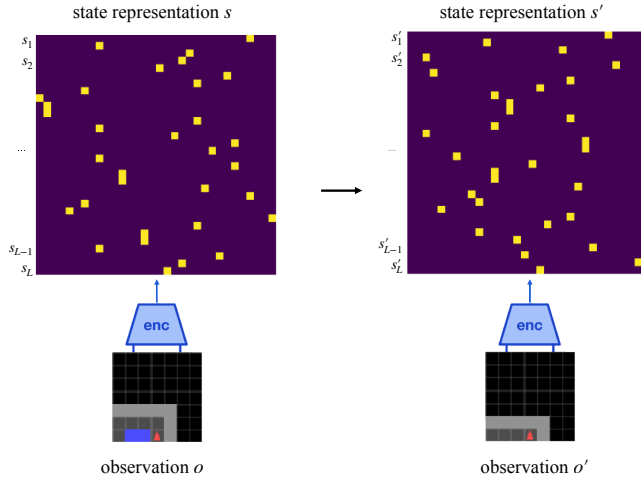


Figure C.2: Factorized discrete latent representations for two consecutive observations. Each of the $L = 32$ latent factors (rows) independently takes one of $K = 32$ discrete values (columns). From one observation to the next, some factors change (e.g. s_1, s_2 —encoding agent position or orientation) while others remain stable (e.g. s_L —encoding environmental context). The factorized structure enables per-factor computation of novelty, information gain, and empowerment, avoiding the combinatorial explosion that would arise from treating the full 32^{32} state as a single discrete variable.

that their magnitudes are comparable and that the sum and product hybrids combine signals at the same granularity.

Policy and value learning. The actor-critic policy operates over imagined trajectories sampled from the world model. Given a starting latent state (h_t, s_t) , the world model generates a sequence of imagined transitions by repeatedly applying the dynamics predictor and the actor policy. The actor and critic are trained on these imagined rollouts using the REINFORCE-with-baseline objective with λ -returns. In our setting, the intrinsic reward $r_{\text{intrinsic}}$ is computed at each environment step and passed to DreamerV3 in place of the extrinsic reward; the world model and policy then process this reward in exactly the same way as an extrinsic reward.

Implementation details. We follow the hyperparameters and architecture of Ferrao and Cunha (2025) without modification, with the exception of using MiniGrid’s partial pixel-based observation (a $7 \times 7 \times 3$ RGB image of the agent’s field of view) rather than the semantic tile encoding they used. Key hyperparameters include: replay buffer size 10^6 , batch size 16 sequences of length 64, recurrent state dimension $d = 512$, learning rates 3×10^{-4} for the world model and 3×10^{-5} for the actor and critic, and a discount factor $\gamma = 0.997$. Training runs for 10^6 environment steps per condition. Due to computational constraints, we run a single seed per condition in the Dreamer experiments.

C.2 Information gain variants

The information gain (IG) reward in Section 6.3.2 is implemented differently for the Tabular and Dreamer agents: the Tabular agent computes a *predicted* (prospective) IG, while the Dreamer agent uses a *retrospective* approximation. This section defines all three variants that appear in the literature—actual, predicted, and retrospective IG—and explains the consequences of this difference for exploration.

C.2.1 Three variants of information gain

Let $p(S' | s, a)$ denote the agent’s current transition model: the distribution over next states given state s and action a . Bayesian IG is defined as the reduction in posterior entropy over some unknown quantity Θ (here, the

1: Θ denotes the unknown parameters of the transition law, not the next-state variable itself. In the tabular case, Θ may be identified with the collection of multinomial parameters $\{p(\cdot | s, a)\}_{s,a}$

2: More precisely, $p(S'' | s, a, s')$ denotes the posterior predictive distribution after updating the belief over Θ with the observation (s, a, s') . Equivalently,

$$\begin{aligned} & p(S'' | s, a, s') \\ &= \int p(S'' | s, a, \Theta) p(\Theta | s, a, s') d\Theta \end{aligned} \quad (\text{C.11})$$

transition dynamics¹) after observing a new data point. In the model-learning setting, this corresponds to the change in the agent's model after observing a transition (s, a, s') .

Actual information gain. The simplest variant computes IG *retrospectively*, after observing a specific transition to s' :

$$r_{\text{IG}}^{\text{actual}}(s, a, s') = D_{\text{KL}}(p(S'' | s, a, s') \| p(S'' | s, a)), \quad (\text{C.10})$$

where $p(S'' | s, a, s')$ is the posterior distribution after updating the transition model with the observation of s' ², and $p(S'' | s, a)$ is the prior. In the tabular setting, this update is exact: incrementing $N(s, a, s')$ by one and recomputing via Equation (6.1) yields the exact posterior. This measure is nonzero only after observing a specific outcome, and so it can only be used as a retrospective reward.

Predicted (prospective) information gain. The prospective variant—the one we use for the tabular agent in Equation (6.10)—computes the *expected* actual IG before observing any specific outcome:

$$r_{\text{IG}}^{\text{pred}}(s, a) = \mathbb{E}_{\hat{s}' \sim p(S' | s, a)} [r_{\text{IG}}^{\text{actual}}(s, a, \hat{s}')] = \text{MI}(S'; \Theta | s, a), \quad (\text{C.12})$$

where the expectation is taken over the current predictive distribution. This quantity equals the mutual information between the unknown model Θ and the next state S' —the amount by which observing the transition is expected to reduce uncertainty about the model. The predicted IG is a prospective measure: it quantifies how much the agent *expects* to learn before taking an action. It can therefore be used for planning—the agent can compute $r_{\text{IG}}^{\text{pred}}$ for all state-action pairs and choose the action that maximizes expected model improvement, whereas actual IG is reward-only which can only be assigned after the transition is observed. Prior work (Gruaz et al., 2024; Little and Sommer, 2013) has shown that tabular agents using predicted IG improve their world models faster than those using other measures.

Retrospective KL (Dreamer). In a neural world model, computing the predicted IG (Equation (C.12)) would require: (a) running the dynamics predictor forward for each possible action, (b) computing the posterior for each possible next latent outcome, and (c) averaging over the current predictive distribution. Step (b) alone is intractable for DreamerV3's discrete latent state, which factorizes into $L = 32$ categorical variables with 32 categories each, yielding a joint support of size 32^{32} .

The approximation we use (Equation (6.11)) therefore replaces exact Bayesian IG with a posterior—prior KL in latent space:

$$r_{\text{IG}}^{\text{retro}}(s, a, s') = \frac{1}{L} \sum_{i=1}^L D_{\text{KL}}(p_{\phi}(s'_i | h', o') \| p_{\phi}(s'_i | h')). \quad (\text{C.13})$$

Here, s'_i denotes the i th categorical latent factor of the next-step latent state, and L is the number of such factors. The encoder posterior $p_{\phi}(s'_i | h', o')$ is the distribution over that factor after incorporating the new observation o' , while the dynamics prior $p_{\phi}(s'_i | h')$ is the model's prediction before seeing o' . Averaging over i yields a scalar summary of how strongly the observation revises the model's latent prediction across the factorized latent state.

This quantity is not exact Bayesian IG over transition parameters. Rather, it is a variational proxy: a measure of posterior—prior mismatch in the learned latent representation. When the latent model is well calibrated,

this KL is typically small on well-modelled transitions and larger on surprising or poorly predicted ones³.

C.2.2 Consequences for exploration

The key difference between predicted and retrospective IG is that the predicted variant is intrinsically *prospective*: it assigns value to a state-action pair before the outcome is observed, and can therefore be used directly for planning. By contrast, the retrospective variant is defined only after a transition has occurred. It can still influence future behavior through value propagation, but only indirectly, via experienced returns rather than advance evaluation of unseen transitions.

This asymmetry helps interpret the difference between the Tabular and Dreamer settings. In the Tabular agent, predicted IG can directly highlight actions that are expected to improve the model, including actions that lead toward uncertain but potentially high-empowerment regions. In the Dreamer agent, the retrospective KL signal becomes available only once informative transitions have been experienced, so any preference for routes leading to such regions must be learned from downstream reward propagation rather than anticipated immediately.

In bottlenecked environments, this can delay the discovery and exploitation of informative regions beyond the bottleneck. Predicted IG can mark the bottleneck-crossing action as promising before any successful traversal, if the model already assigns high expected information value beyond it. Retrospective IG usually requires experience on or beyond the bottleneck before that route acquires value.

A potential solution is *ensemble disagreement* (Sancaktar et al., 2025), which maintains multiple dynamics predictors and uses their disagreement as a prospective measure of IG. This approximates the expected IG (Equation (C.12)) without requiring enumeration of all possible next latents, and has been shown to successfully guide DreamerV3-based agents in complex environments. Implementing ensemble disagreement in place of the retrospective KL would likely strengthen the complementarity effect observed in our Dreamer experiments.

C.2.3 Information gain and aleatoric uncertainty

A further distinction concerns the treatment of irreducible (aleatoric) uncertainty. The predicted IG (Equation (C.12)) theoretically approaches zero for stochastic transitions: once the agent has learned the aleatoric variance σ^2 of a transition (e.g., that ice tiles produce uniformly random next states), no further data can reduce the model’s uncertainty about the transition dynamics, so the expected posterior entropy equals the prior entropy. In practice, this convergence is approximate: with a finite Dirichlet-like count model, the predicted IG for stochastic transitions decays slowly toward a small positive value as counts accumulate.

The retrospective KL (Equation (C.13)) is more susceptible to aleatoric confounding. Because it measures the divergence between the encoder posterior (which has access to the actual observation) and the dynamics predictor prior (which does not), it remains high whenever the observation is informative about the current latent state—even if the *transition* is inherently stochastic. If the world model cannot perfectly predict the next latent from the previous state and action alone (which is impossible for

3: A large posterior-prior KL can reflect epistemic uncertainty, but also representation drift, encoder mismatch, or observation noise. In learned latent models, these effects are not cleanly separable.

Remark C.1 (Surprise, IG, and learning progress.) These quantities are related but not equivalent. **Surprise** measures how unlikely an observed transition is under the agent’s current predictive model: $\text{Surprise}(s, a, s') = -\log p(s' | s, a)$. A transition may be highly surprising simply because it is rare under the current model. **Information gain** measures how much the observation updates the agent’s belief about the dynamics. In a Bayesian formulation, with Θ denoting the unknown transition-model parameters, the actual IG from observing (s, a, s') is Equation (C.10). Thus, IG measures model change, not merely outcome improbability. **Learning progress** measures improvement in predictive performance over time. A simple form is the reduction in prediction loss between two training iterations t and $t + 1$: $\text{LP}_{t+1}(s, a, s') = \ell_t(s, a, s') - \ell_{t+1}(s, a, s')$, where ℓ_t is a prediction loss, for example negative log-likelihood, $\ell_t(s, a, s') = -\log p_t(s' | s, a)$. Learning progress is therefore a temporal difference in model quality.

aleatoric transitions), the KL divergence will remain elevated indefinitely, regardless of how many times the transition has been experienced. This is the mechanism underlying the IG fixation on ice tiles observed in Figure 6.17.

C.3 Nonstationarity of intrinsic rewards

Intrinsic motivation signals differ from extrinsic task rewards in a fundamental way: they depend on the agent's *epistemic state*, and are therefore generally *nonstationary over learning*. An extrinsic reward—reaching a goal, collecting a coin—is a fixed property of the environment dynamics and task specification. Once the environment and task are fixed, the extrinsic reward attached to a transition does not change merely because the agent has become more knowledgeable. However, novelty depends on visitation counts, information gain depends on the agent's current posterior or predictive uncertainty, and empowerment estimates depend on a world model whose parameters are themselves being updated. The same transition can therefore receive a different intrinsic reward at different moments in training, not because the environment has changed, but because the agent's internal world model has changed.

This distinction matters because most RL analyses assume a stationary Markov decision process with a fixed reward function. For a learned intrinsic reward, the natural object is instead a time-indexed reward function

$$r_t(s, a, s'), \quad (\text{C.14})$$

where t denotes training time and indexes the current epistemic state of the agent. Holding the transition kernel fixed, the induced control problem is more accurately written as a sequence of MDPs,

$$\mathcal{M}_t = (S, \mathcal{A}, P, r_t, \gamma), \quad (\text{C.15})$$

rather than a single stationary MDP with one reward function r . Accordingly, under a fixed policy π , the relevant action-value function is also time-indexed:

$$Q_t^\pi(s, a) = \mathbb{E} \left[\sum_{k=0}^{\infty} \gamma^k r_{t+k}(s_k, a_k, s_{k+1}) \mid s_0 = s, a_0 = a, \pi \right]. \quad (\text{C.16})$$

4: Standard Q-learning instead behaves as if there were a single stationary target:

$$Q^\pi(s, a) = \mathbb{E} \left[\sum_{k=0}^{\infty} \gamma^k r(s_k, a_k, s_{k+1}) \right].$$

This is already a different object from the stationary value function assumed by standard temporal-difference methods⁴. Those methods are typically derived as if there were a single Bellman operator with a single fixed point. With intrinsic rewards, however, the Bellman operator itself drifts as learning changes the reward landscape. Thus the difficulty is that the target being bootstrapped toward moves during learning, so the algorithm must track a changing fixed point rather than converge to a static one.

We encountered this problem directly in both the Tabular and Dreamer experiments. The two cases differ in their mechanics, but both instantiate the same underlying mismatch between a nonstationary intrinsic objective and learning algorithms that were designed around stationary reward targets.

C.3.1 Tabular setting: stale Q-values

In the tabular setting, we initially experimented with a standard model-free Q-learning update:

$$Q(z, a) \leftarrow Q(z, a) + \eta \left[r(z, a, z') + \gamma \max_{a'} Q(z', a') - Q(z, a) \right], \quad (\text{C.17})$$

where $r(z, a, z')$ is the intrinsic reward at the current time step. This approach failed systematically.

The problem is that Q-values are updated incrementally and only for states that have been visited. When a curiosity-based intrinsic reward decays—as it should when a state has been visited many times—the Q-values for that state are slow to reflect this decay, because Q-learning bootstraps from past estimates. The agent therefore continues to plan paths toward states that are no longer intrinsically rewarding, wasting time chasing stale gradients.

For novelty in particular, the nonstationarity is global: when any state is visited, the visitation probability $p(z)$ for all states changes (since the normalizer $\sum_z N(\tilde{z})$ increases), causing the novelty reward to shift for states that were not visited at all. A model-free agent has no mechanism to propagate these global updates through its value estimates.

Solution: model-based prioritized sweeping. The solution we adopted—model-based Q-value computation via prioritized sweeping (Equation (6.2) in Section 6.3.2)—sidesteps this problem by recomputing Q-values from the current model at each time step. Because the world model is always up-to-date (reflecting the counts observed so far), and because Q-values are derived from the model rather than accumulated through TD bootstrapping, they automatically reflect the current intrinsic reward landscape. This requires more computation per step ($n_{\text{sweeps}} = 100|\mathcal{Z}|$ sweeping iterations in our implementation), but eliminates the stale-value problem entirely.

Even with model-based sweeping, we found that novelty requires more sweeping iterations than information gain or empowerment, because a single state visit updates the global normalizer and therefore changes novelty rewards for all states simultaneously. The Q-values must be updated for the entire state space on each visit, not just for states downstream of the visited state.

C.3.2 Dreamer setting: replay buffer contamination

In the Dreamer setting, the nonstationarity problem manifests differently and is harder to solve within the DreamerV3 framework. DreamerV3 stores all experienced transitions (o, a, o', r) in a replay buffer of size 10^6 , and trains its world model and policy by sampling random batches from this buffer.

When intrinsic rewards are stored in the replay buffer alongside transitions, a transition stored early in training carries the intrinsic reward computed by an *early* world model. Late in training, when the world model has substantially improved, the policy gradient is still trained on some fraction of these early transitions—with their outdated intrinsic rewards. For nonstationary intrinsic rewards, this contamination is systematic: early transitions carry high novelty or information gain rewards (when the model was uncertain), while the same states should have

near-zero reward late in training. The policy therefore continues to plan toward previously explored states long after exploration should have moved on.

We observed several specific behavioral signatures of this contamination:

- ▶ The novelty-driven agent periodically revisiting the starting room despite having explored further regions, because early transitions in the room carry high novelty rewards in the buffer.
- ▶ The empowerment-driven agent spending extended episodes repeating the same action in the same state, because the empowerment reward for that state-action pair is stable and the buffer contains many high-reward records of it.
- ▶ The hybrid (information gain + empowerment) agent appearing to reduce to a pure information-gain agent, because the combined reward early in training is dominated by information gain, biasing the sampled buffer.

Potential solutions. Several approaches could mitigate replay buffer contamination; they were not in scope for this work but are important directions for future implementations:

1. **Reward recomputation on sampling.** Rather than storing intrinsic rewards in the buffer, store only the observations and actions, and recompute intrinsic rewards at sampling time using the current world model. This is computationally expensive but eliminates stale rewards entirely.
2. **Recency-weighted replay.** Prioritize recent transitions when sampling from the buffer (Kauvar et al., 2023). Since recent transitions carry rewards computed by the current model, this reduces the contamination from early training.
3. **Prospective reward computation.** Compute expected intrinsic rewards before experiencing transitions, using the world model to imagine future dynamics. This is the approach taken by Plan2Explore (Sekar et al., 2020), which applies an ensemble-disagreement information gain bonus prospectively, allowing the agent to plan toward informative states before visiting them.
4. **Separate buffers.** Maintain a small recent buffer (for intrinsic reward learning) and a large archive buffer (for world model learning), sampling from the former for policy updates and from both for model training.

Taken together, these two failure modes illustrate the same technical principle. Intrinsic rewards do not merely inject additional stochasticity into an otherwise standard RL problem; they make the reward specification itself evolve with learning. Any method that caches past value estimates or past reward labels without correcting them for the current epistemic state will therefore accumulate stale training targets.

This is why intrinsic motivation signals fit especially naturally within model-based architectures. The advantage is not only that many such signals are computed from a model, but that a model-based agent can in principle recompute rewards, values, or imagined returns under its *current* beliefs rather than inherit obsolete targets from earlier phases of training. The tabular agent's prioritized sweeping is a simple but exact instance of this idea. Achieving the same correction in neural world-model agents remains substantially harder, because replay, function approximation, and joint representation learning all preserve traces of earlier epistemic states. Making nonstationary intrinsic objectives compatible with scalable deep RL is therefore not a minor implementation

detail, but a central open problem in the design of intrinsically motivated agents.

C.4 Blahut-Arimoto and factorized mutual information

Empowerment is defined as the channel capacity of the agent's actuation channel—the maximum mutual information between action sequences and resulting states. This appendix describes the computation methods used for the Tabular (Equation (6.13)) and Dreamer (Equation (6.14)) settings, their theoretical properties, and their limitations.

C.4.1 Empowerment via Blahut-Arimoto

In the tabular setting, we compute *one-step empowerment* at each state s as

$$\text{EMP}(s) = \max_{\omega(a|s)} \text{MI}(A; S' | s) = \max_{\omega(a|s)} [\text{H}(S' | s) - \text{H}(S' | A, s)], \quad (\text{C.18})$$

where $\omega(a|s)$ denotes the source policy over actions and the mutual information is taken with respect to the transition model $p(s'|s, a)$. Equivalently, for each fixed state s , this is the channel capacity of the discrete action-to-next-state channel with input alphabet $\mathcal{A}$ and output alphabet given by the reachable next states.

We solve Equation (C.18) with the Blahut-Arimoto algorithm (Arimoto, 1972; Blahut et al., 1972). Starting from a uniform source policy $\omega^{(0)}(a|s) = \mathcal{U}(\mathcal{A})$, the algorithm alternates between computing the successor-state marginal induced by the current source policy,

$$m^{(t)}(s' | s) = \sum_{a \in \mathcal{A}} \omega^{(t)}(a|s) p(s'|s, a), \quad (\text{C.19})$$

and updating the source policy according to

$$\omega^{(t+1)}(a|s) \propto \omega^{(t)}(a|s) \exp\left(D_{\text{KL}}(p(\cdot|s, a) \| m^{(t)}(\cdot|s))\right), \quad (\text{C.20})$$

followed by normalization over $a \in \mathcal{A}$. Intuitively, Equation (C.20) increases the probability of actions whose induced successor-state distributions are more distinguishable from the average successor-state distribution under the current source policy. For finite discrete channels, the Blahut-Arimoto iterations converge to the channel capacity, and therefore to the exact one-step empowerment value (Arimoto, 1972; Blahut et al., 1972).

The loop performs coordinate-ascent on $\text{MI}(A; S' | s)$, which is concave in ω , so convergence to the global maximum is guaranteed regardless of initialisation.

In our tabular environment, we have $|\mathcal{A}| = 3$ actions and approximately 372 states. As a result, the optimization in Equation (C.18) is inexpensive and converges in only a small number of iterations for each state. We therefore compute $\text{EMP}(s)$ exactly for every state in the tabular setting and use it directly as the intrinsic reward at time step t :

$$r_t^{\text{int}} = \text{EMP}(s_t). \quad (\text{C.21})$$

```

1   Input: transition model  $p(z' | z, a)$  for fixed state  $z$ 
2   convergence threshold  $\varepsilon > 0$ 
3   Output: one-step empowerment  $\mathcal{E}(z)$ 
4   Initialise  $\omega(a) \leftarrow \mathcal{U}(\mathcal{A})$  for all  $a \in \mathcal{A}$ 
5   repeat until  $T < \varepsilon$ :
6       // E-step
7        $\Phi(a | z') \leftarrow \frac{p(z' | z, a) \omega(a)}{\sum_{b \in \mathcal{A}} p(z' | z, b) \omega(b)}$ 
8       for all  $a \in \mathcal{A}, z'$ 
9       // M-step
10       $r(a) \leftarrow \exp(\sum_{z'} p(z' | z, a) \log \Phi(a | z'))$ 
11      for all  $a \in \mathcal{A}$ 
12       $\omega(a) \leftarrow r(a) / \sum_{b \in \mathcal{A}} r(b)$ 
13      for all  $a \in \mathcal{A}$ 
14       $C \leftarrow \log \sum_{a \in \mathcal{A}} r(a)$ 
15       $T \leftarrow \max_a \log(r(a) / \omega(a)) - C$ 
16  return  $C$ 
17

```

Figure C.3: Blahut–Arimoto procedure for one-step empowerment.

Thus, the reward depends only on the current state and not on the sampled action a_t or the realized successor state s_{t+1} .

Interpretation. One-step empowerment treats control as a property of the *current state*, rather than of a specific transition. The question it asks is: *if the agent were in state s and could freely choose its action distribution, how much influence could it exert over the next state in a single step?* States with many reliably distinguishable successor states have high empowerment, whereas states in which different actions produce identical or highly overlapping successor-state distributions have low empowerment.

This perspective is especially intuitive in our environment. Lava cells have zero empowerment because they are terminal and therefore admit no future options. Ice cells have low empowerment because the transition dynamics are highly stochastic, making the next state weakly dependent on the chosen action. By contrast, open floor cells typically have high empowerment because different actions reliably produce different successor states. More generally, empowerment can be interpreted as measuring the number of *reliably distinguishable futures* available from a state.

Deterministic special case. A useful sanity check is the deterministic case. If the dynamics at state s are deterministic, so that each action a maps to a unique successor state $f_s(a)$, then one-step empowerment reduces to

$$\text{EMP}(s) = \log |\{f_s(a) : a \in \mathcal{A}\}|, \quad (\text{C.22})$$

that is, the logarithm of the number of distinct next states reachable in one step. For example, if all actions lead to the same successor state, then $\text{EMP}(s) = 0$; if three actions lead deterministically to three different successor states, then $\text{EMP}(s) = \log 3$. This deterministic limit matches the intuition that empowerment quantifies the diversity of controllable immediate futures.

Remark on multi-step empowerment. We only consider one-step empowerment in the experiment. The multi-step⁵ one can be viewed as the channel capacity of the induced channel from length- k action sequences to the resulting state (or observation) after k steps. In principle, this larger optimization can still be solved exactly with the Blahut–Arimoto algorithm. However, the input alphabet now consists of all action sequences of length k , and therefore has size $|\mathcal{A}|^k$, so exact computation becomes exponentially more expensive in the horizon and is practical

5: A natural k -step generalization is

$$\begin{aligned} \text{EMP}^{(k)}(s) &= \max_{\omega(a_{0:k-1} | s)} \text{MI}(A_{0:k-1}; S_k | s) \end{aligned} \quad (\text{C.23})$$

only for very small tabular problems and short horizons. A further nuance is that multi-step empowerment is typically defined over *open-loop* action sequences rather than adaptive closed-loop policies. This keeps the information-theoretic channel formulation clean, but it is slightly different from asking how much control an agent has when it can react to intermediate observations at every step.

C.4.2 Factorized empowerment in DreamerV3

For the Dreamer agent, applying Blahut-Arimoto requires computing $p(s' | s, a)$ over the full discrete latent space $\mathcal{S} = K^L$. With $K = 32$ and $L = 32$, this space has $32^{32} \approx 10^{48}$ elements—completely intractable to enumerate.

We use two approximations, following Seitzer et al. (2021):

Approximation 1: uniform source policy. Rather than optimizing over source policies, we fix $\omega = \mathcal{U}(A)$, a uniform distribution over actions. This gives a lower bound on the true empowerment:

$$\text{MI}_{\mathcal{U}}(S'; A | s) \leq \text{EMP}(s). \quad (\text{C.24})$$

Equality holds when a uniform action distribution is capacity-achieving for the induced action-to-next-state channel. This occurs, for example, in symmetric environments where all actions are equally distinguishable. In our mixed-state playground, the optimal source policy assigns more weight to actions that lead toward floor cells and less weight to actions toward ice or walls; a uniform policy underestimates the agent’s true controllability.

Limitations. The uniform source policy is a fixed approximation that does not adapt to the agent’s actual policy. An alternative would be to use the agent’s current policy π as the source policy, computing “under my current behavior, how much control do I have?” rather than “under any possible behavior, how much control could I have?” This would measure the agent’s *actual* influence under its current behavior rather than its *potential* influence under an optimized source distribution. Such a variant may be a better match to intrinsic motivation during learning, where the agent’s current behavioral repertoire is often the relevant practical constraint.

Approximation 2: per-factor mutual information. Rather than computing mutual information over the joint latent space, we compute it separately for each latent factor s_i and average:

$$r_{\text{emp}}^{\text{dream}}(s) = \frac{1}{L} \sum_{i=1}^L \text{MI}_{\mathcal{U}}(S'_i; A | s) = \frac{1}{L} \sum_{i=1}^L [\text{H}(S'_i | s) - \text{H}(S'_i | A, s)], \quad (\text{C.25})$$

where the marginal entropy $\text{H}(S'_i | s) = \text{H}\left[\frac{1}{|\mathcal{A}|} \sum_a p_{\phi}(s'_i | h', a)\right]$ and the conditional entropy $\text{H}(S'_i | A, s) = \frac{1}{|\mathcal{A}|} \sum_a \text{H}[p_{\phi}(s'_i | h', a)]$ are both computed from the dynamics predictor’s predicted categorical distributions⁶.

Per-factor averaging assumes that the factors are approximately independent in their controllability, which is encouraged by the factorized structure of DreamerV3’s latent space. If the factors are independent, then per-factor MI averages to the joint MI (by the chain rule); if they are

Remark C.2 The exact quantity is $I(Z'; A | s)$, with $Z' = (Z'_1, \dots, Z'_L)$, whereas we approximate it by $\sum_i I(Z'_i; A | s)$. By the chain rule,

$$I(Z'; A | s) = \sum_i I(Z'_i; A | Z'_{<i}, s).$$

Thus, the factorwise sum is exact only when earlier factors do not make later ones redundant as sources of information about A .

6: Remind here h' denotes the recurrent deterministic state obtained by rolling the model forward from the current latent state under action a ; $p_{\phi}(s'_i | h', a)$ is the model’s predicted categorical distribution for factor i of the next latent state.

7: When two factors encode overlapping consequences of the action, each factor can individually appear informative about $\mathcal{A}$, so summing their mutual informations counts shared action-information multiple times.

correlated, per-factor averaging overestimates the joint MI (by double-counting⁷). In practice, some correlation between factors is expected, so the reward provides an overestimate of the true joint mutual information—but still a useful signal for directing the agent toward controllable states.

Connections to causal influence. The per-factor mutual information in Equation (C.25) is closely related to the Causal Influence metric of Seitzer et al. (2021), which computes per-object causal influence in an object-centric representation. Our formulation adapts this idea to a learned factorized latent space, where the “objects” are abstract representational units discovered by DreamerV3 rather than predefined entities. This suggests a path toward scaling empowerment-based intrinsic motivation to more complex environments: representations with approximately factorized structure may admit analogous per-factor empowerment approximations.

Additional Material for Chapter 7

D.

The appendix for [Chapter 7](#) is organized as follows:

- ▶ [Section D.1](#) gives the formal version of the framework: representation-state channels, the goal-projection construction, the coverage-gain proxy, and the minimum-cost sufficiency theorem behind the “good-enough, then compress” behavior.
- ▶ [Section D.2](#) documents the closed-vocabulary causal-learning environment shared by [Sections 7.4](#) and [7.5](#).
- ▶ [Section D.3](#) reports the human study protocol and supplementary analyses.
- ▶ [Section D.4](#) reports the matched simulations and the identifiability analysis behind the medium-granularity advantage.
- ▶ [Section D.5.1](#) documents the open-vocabulary Curator-Actor implementation, environments, training and evaluation protocol, baselines, transfer, and library-compactness measurements for [Section 7.6](#).

D.1 Formal results for representation empowerment	247
D.2 Closed-vocabulary causal-learning environment . .	252
D.3 Human study supplementary results	254
D.4 Matched-simulation supplementary results .	258
D.5 Open-vocabulary Curator-Actor implementation and results	261

D.1 Formal results for representation empowerment

This appendix gives the formal version of the theory used in [Section 7.3](#). The central distinction is between exact representation empowerment, a channel capacity over future internal representational states, and the grounded coverage proxy that the experiments actually compute. The latter is a practical approximation to the former. The results below clarify three points: why raw syntactic diversity is not an adequate empirical objective; why goal-achievement signatures supply a behaviorally relevant projection of representational states; and why the resulting coverage objective implements a “good-enough, then compress” preference under a small enough cost penalty.

D.1.1 Representation-state channels

Let $\mathcal{Z}$ be a space of internal representational states. A state $z \in \mathcal{Z}$ may be a task model M , a library $\mathcal{L}$, or the pair $(M, \mathcal{L})$ together with the current evidence. Let Ω be a finite set of representational edits, e.g., proposing a variable, testing an edge, refining an abstraction, composing two symbolic operators, specializing a predicate, or pruning an element. For a horizon T , write $u = \omega_{1:T} \in \Omega^T$ for an edit sequence. Conditional on the current environment and evidence, edits induce a stochastic representation-dynamics kernel

$$K_T(z' | z, u) = \Pr(Z_T = z' | Z_0 = z, U = u).$$

We suppress the conditioning on environment and evidence when it is not needed.

Definition D.1 (Exact representation empowerment) The T -step rep-

representation empowerment of representational state z is

$$\text{RepEmp}(z) = \mathcal{R}_T(z) = \sup_{\pi \in \Delta(\Omega^T)} \text{MI}_{\pi K_T}(U; Z_T \mid Z_0 = z),$$

where $U \sim \pi$ and $Z_T \sim K_T(\cdot \mid z, U)$.

This is the direct symmetry of environmental empowerment (Abel et al., 2025; Zhou et al., 2025). The input alphabet is a set of internal edits and the output alphabet is a set of future internal representations.

Proposition D.1 (Diversity-control decomposition) For every edit distribution π ,

$$\text{MI}(U; Z_T \mid Z_0 = z) = H(Z_T \mid Z_0 = z) - H(Z_T \mid U, Z_0 = z).$$

Representation empowerment is therefore high only when future representations are diverse *and* controllably reachable.

Proof. This is the chain-rule identity for conditional mutual information. $\square$

D.1.2 Grounding by goal projection

Exact empowerment over Z_T is usually too large to estimate directly. The agent has no behavior-free oracle for Z_T , and two representations that differ syntactically can produce identical behavior. We therefore observe a behaviorally relevant projection of Z_T . For environment e , let

$$\Gamma_e(z) = \left(\mathbb{I} \left[g \text{ is achieved by planning with } z \text{ and executing in } e \right] \right)_{g \in \mathcal{G}_e}$$

be the goal-achievement signature of z . Two representations with identical Γ_e may differ syntactically but are equivalent for the probe goals used to evaluate the task.

Definition D.2 (Grounded representation empowerment) The grounded representation empowerment of z in environment e is

$$\widehat{\mathcal{R}}_T(z; e) = \sup_{\pi \in \Delta(\Omega^T)} \text{MI}_{\pi K_T}(U; \Gamma_e(Z_T) \mid Z_0 = z).$$

Proposition D.2 (Projection lower bound) For every deterministic projection Γ_e ,

$$\widehat{\mathcal{R}}_T(z; e) \leq \mathcal{R}_T(z).$$

Proof. For any fixed π , the variables form a Markov chain $U \rightarrow Z_T \rightarrow \Gamma_e(Z_T)$ conditional on $Z_0 = z$. The data-processing inequality gives $\text{MI}(U; \Gamma_e(Z_T) \mid Z_0 = z) \leq \text{MI}(U; Z_T \mid Z_0 = z)$. Taking the supremum over π on both sides preserves the inequality. $\square$

Proposition D.3 (Raw syntactic diversity need not be useful) There are representation channels with positive exact representation em-

powerment but zero grounded representation empowerment.

Proof. Let $\Omega^T = \{1, \dots, n\}$ and suppose edit u deterministically maps z to a distinct representation z_u . Then the raw channel $U \mapsto Z_T$ is noiseless and has capacity $\log n$. Now suppose all z_u are syntactic variants of an irrelevant symbol, so $\Gamma_e(z_u)$ is the same vector for all u . The projected channel $U \mapsto \Gamma_e(Z_T)$ is constant and has capacity zero. $\square$

The edits in this construction give the agent control over which representation it writes down without giving it any control over goal-relevant consequences. Exact representation empowerment counts both kinds of distinction; the grounded quantity counts only those that change the achievable-goal signature. We therefore avoid using unprojected syntactic entropy as the empirical objective: it would reward producing arbitrary representational variants over expanding what the agent can do. The grounded quantity also makes compression well-defined, since representations with identical Γ_e are not distinguished by $\widehat{\mathcal{R}}_T(z; e)$, so any cost penalty (Theorem D.5) breaks the tie toward the cheaper representation.

D.1.3 From channel capacity to the coverage proxy

The goal-achievement signature induces the coverage score

$$\mathcal{F}_e(M) = \sum_{g \in \mathcal{G}_e} \Gamma_e(M)_g,$$

the count of probe goals achievable under M in e . This is a linear utility over the grounded output alphabet, used when estimating channel capacity directly is impractical. At time t , a modeling action a produces an observation o and an updated model $U(M_t, a, o)$. The expected coverage proxy is

$$\widehat{V}_t(a) = \mathbb{E}_{o \sim p_t(\cdot|a)} [\mathcal{F}_e(U(M_t, a, o)) - \mathcal{F}_e(M_t)] - \lambda \Delta \mathcal{C}_t(a),$$

where $\Delta \mathcal{C}_t(a)$ is the representational cost incurred by a .

For a validated element σ , the realized marginal proxy collapses to the score difference,

$$V_{\mathcal{G}}(\sigma; M, e) = \mathcal{F}_e(M \cup \{\sigma\}) - \mathcal{F}_e(M).$$

For an unvalidated candidate, let $q_t(\sigma)$ be its current probability of being valid and $W_t(\sigma) \subseteq \mathcal{G}_e$ its *witness set*, the goals currently unachievable under M_t whose achievement bit could flip on if σ were added. Goals outside $W_t(\sigma)$ contribute zero to $\mathcal{F}_e(M_t \cup \{\sigma\}) - \mathcal{F}_e(M_t)$, either because they are already achievable (no upside) or because adding σ alone cannot unlock them (no leverage). Restricting the sum to $W_t(\sigma)$ therefore loses no signal, and the proxy in Equation (7.8) becomes

$$\widehat{V}_t(\sigma) \approx q_t(\sigma) \sum_{g \in W_t(\sigma)} (\widehat{p}_t(g | M_t \cup \{\sigma\}) - \widehat{p}_t(g | M_t)) - \lambda c_{\sigma},$$

where $\widehat{p}_t(g | \cdot)$ is the empirical achievement rate of g returned by the Actor.

Proposition D.4 (Singleton witness channels) Consider a deterministic two-edit channel with edits $u_0 = \text{keep}$ and $u_1 = \text{add}(\sigma)$. If $\Gamma_e(M \cup \{\sigma\}) = \Gamma_e(M)$, the grounded capacity of this channel is zero. If the two signatures differ, the channel has positive grounded capacity.

Proof. If the signatures are equal, the output is constant for both inputs, so mutual information is zero for all input distributions. If the signatures differ, the input-output map is deterministic and injective on the two inputs; choosing a non-degenerate input distribution yields positive mutual information. $\square$

In the singleton deterministic case, a witness goal marks exactly the goal-relevant distinction that gives the projected channel positive capacity. The coverage proxy replaces direct capacity estimation with a finite score over these projected distinctions, and edits are favored when they reliably produce representation changes that unlock currently blocked goals.

Bundle additivity. For a multi-element proposal $\Delta M \subseteq \Sigma_t$, the realized score difference decomposes as

$$\mathcal{F}_e(M_t \cup \Delta M) - \mathcal{F}_e(M_t) = \sum_{\sigma \in \Delta M} [\mathcal{F}_e(M_t \cup \Delta M) - \mathcal{F}_e(M_t \cup \Delta M \setminus \{\sigma\})],$$

which is the sum of *order-dependent* marginals, not the sum of $V_G(\sigma; M_t, e)$. The two coincide when the witness sets $\{W_t(\sigma)\}_{\sigma \in \Delta M}$ are pairwise disjoint, since each goal g is then in the witness set of at most one σ and its achievement bit can only flip via that σ . When witness sets overlap, two candidates may share a goal g that either alone unlocks; summing the singleton proxies double-counts g , and the additive estimator

$$\widehat{V}_t(\Delta M) \approx \sum_{\sigma \in \Delta M} \widehat{V}_t(\sigma)$$

is an upper bound on the true joint gain. We use the additive form throughout because it is constant-time given the per-element scores and because empirical witness-set overlap is small in our experiments. The Curator-Actor loop accepts a bundle only if the realized $\mathcal{F}_e$ on the probe set increases, which catches the cases where the additive estimate over-promises.

D.1.4 Minimum-cost sufficiency for the coverage proxy

The next results apply to the finite coverage proxy rather than to raw unprojected representation entropy. They justify the “good-enough, then compress” behavior used in the empirical sections.

Definition D.3 (Goal sufficiency) A model M is goal-sufficient for environment e if

$$\mathcal{F}_e(M) = \mathcal{F}_e^\star := \max_{M' \subseteq \Sigma} \mathcal{F}_e(M').$$

It is minimum-cost goal-sufficient if it is goal-sufficient and no other goal-sufficient model has smaller cost.

Assume the cost is additive, $\mathcal{C}(M) = \sum_{\sigma \in M} c_\sigma$ with $c_\sigma > 0$.

Theorem D.5 (Minimum-cost goal sufficiency) Assume Σ is finite and that not every $M \subseteq \Sigma$ is goal-sufficient. Let

$$C_{\max} = \sum_{\sigma \in \Sigma} c_{\sigma}, \quad \Delta_e = \min_{M: \mathcal{F}_e(M) < \mathcal{F}_e^*} (\mathcal{F}_e^* - \mathcal{F}_e(M)) > 0.$$

If $0 < \lambda < \Delta_e / C_{\max}$, every maximizer of $\mathcal{F}_e(M) - \lambda \mathcal{C}(M)$ is a minimum-cost goal-sufficient model.

Proof. Let M^* maximize $\mathcal{F}_e(M) - \lambda \mathcal{C}(M)$ and suppose, for contradiction, that M^* is not goal-sufficient. Let $\tilde{M}$ be any goal-sufficient model, which exists since Σ itself attains $\mathcal{F}_e^*$. Then

$$[\mathcal{F}_e(\tilde{M}) - \lambda \mathcal{C}(\tilde{M})] - [\mathcal{F}_e(M^*) - \lambda \mathcal{C}(M^*)] = (\mathcal{F}_e^* - \mathcal{F}_e(M^*)) - \lambda (\mathcal{C}(\tilde{M}) - \mathcal{C}(M^*)).$$

The first term is at least Δ_e by definition of Δ_e and the assumption that M^* is not goal-sufficient. The cost difference satisfies $|\mathcal{C}(\tilde{M}) - \mathcal{C}(M^*)| \leq C_{\max}$. The display is therefore at least $\Delta_e - \lambda C_{\max} > 0$, contradicting the optimality of M^* . Every maximizer is goal-sufficient. On the set of goal-sufficient models, $\mathcal{F}_e$ is constant at $\mathcal{F}_e^*$, so maximizing $\mathcal{F}_e(M) - \lambda \mathcal{C}(M)$ reduces to minimizing $\mathcal{C}(M)$. Every maximizer is therefore a minimum-cost goal-sufficient model. $\square$

Scope. Theorem D.5 is a consistency property of the cost-regularized coverage proxy under idealized conditions: a finite candidate set, additive costs, exact Γ_e , and a regularizer chosen against a gap Δ_e the agent does not in general know. It establishes that the proxy is well-behaved under these conditions, namely that no element is retained that does not earn its cost and no goal-irrelevant element survives, rather than that representation empowerment is the unique resource-rational criterion for model construction. The substantive support for the objective is empirical (Sections 7.4 to 7.6); the theorem is the sanity check that the proxy does not contradict the desired “good-enough, then compress” behavior.

Definition D.4 (Goal-irrelevant element) An element $\sigma \in \Sigma$ is goal-irrelevant in environment e if

$$\mathcal{F}_e(M \cup \{\sigma\}) = \mathcal{F}_e(M) \quad \text{for all } M \subseteq \Sigma \setminus \{\sigma\}.$$

Corollary D.6 (Pruning of goal-irrelevant elements) If σ is goal-irrelevant, no maximizer from Theorem D.5 contains σ .

Proof. For any model containing σ , removing σ leaves $\mathcal{F}_e$ unchanged and strictly lowers cost. A model containing σ therefore cannot be a minimum-cost goal-sufficient maximizer. $\square$

Causal interpretation. In causal domains where probe goals are intervention targets and the goal-achievement score depends only on intervention-to-goal reachability, any variable or edge outside all intervention-to-goal paths is a sufficient example of a goal-irrelevant element. Learning that such an element exists may reduce uncertainty about the full environment, but retaining it does not change the goal-achievement signature and therefore does not enlarge the grounded empowerment channel.

D.1.5 Behavioral predictions for model construction

Proposition D.7 (Flat frontiers and heterogeneous optima) If M_1 and M_2 are both minimum-cost goal-sufficient models and $\mathcal{C}(M_1) = \mathcal{C}(M_2)$, then both maximize the grounded coverage proxy for every λ satisfying [Theorem D.5](#). More generally, if M_1 has higher score and higher cost than M_2 , there is a threshold value of λ at which the preferred model switches.

Proof. If both models are goal-sufficient and have equal cost, they have identical objective value. For the second claim, compare $\mathcal{F}_e(M_1) - \lambda\mathcal{C}(M_1)$ and $\mathcal{F}_e(M_2) - \lambda\mathcal{C}(M_2)$. Equality occurs at $\lambda^* = (\mathcal{F}_e(M_1) - \mathcal{F}_e(M_2)) / (\mathcal{C}(M_1) - \mathcal{C}(M_2))$, assuming both numerator and denominator are positive. Preference changes when λ crosses this threshold. $\square$

Proposition D.8 (Refinement during discovery, compression after sufficiency) There exists a two-stage model-construction problem in which the fine representation has higher expected proxy value during discovery, but a coarser representation has higher value after the relevant structure has been learned.

Proof. Let the agent choose between a fine representation M_f and a coarse representation M_c with $\mathcal{C}(M_f) > \mathcal{C}(M_c)$. In stage 1, suppose the relevant causal relation is observable only under M_f . A diagnostic edit under M_f then increases expected goal score, while the corresponding edit under M_c does not. The fine representation has higher expected proxy value during discovery. In stage 2, suppose the relevant relation has been identified and can be encoded compactly in either representation, so $\mathcal{F}_e(M_f) = \mathcal{F}_e(M_c) = \mathcal{F}_e^*$. Since M_c is cheaper, the cost-regularized proxy prefers M_c . $\square$

The proxy therefore predicts refinement during discovery and compression after sufficiency rather than a monotonic preference for either representation. Together, these propositions help explain why the human experiment of [Section 7.4](#) can exhibit both stable heterogeneity across participants and a directional compression trend within participants.

D.2 Closed-vocabulary causal-learning environment

This appendix documents the causal-learning environment shared by [Sections 7.4](#) and [7.5](#). The environment is adapted from causal-discovery benchmarks used in model-based RL (Ke et al., 2021) and extended with multi-granularity observation, where the granularity is part of the learner’s action rather than a fixed analyst choice. Both experiments share the variable-and-role state schema, the discretization maps ϕ_ℓ , and the linear-effect SEM mechanism. They differ in graph instantiation. [Section 7.4](#) uses a single hand-designed eight-variable SEM with a fixed four-task curriculum. [Section 7.5](#) samples a random ten-variable SEM per seed under a procedural curriculum, stress-testing the granularity and source findings on graphs unfamiliar to the agent at every seed.

D.2.1 Shared environment primitives

Variables and state. The environment instantiates a linear SEM over a set of latent variables $\mathcal{V} = \{V_1, \dots, V_n\}$. The size and connectivity of $\mathcal{V}$ are experiment-specific. Each variable carries a fundamental scalar value $x \in \{1, 2, \dots, 16\}$ together with three property roles, *strong*, *weak*, and *noise*. Only *strong* and *weak* participate in mechanisms; *noise* is a distractor role included in the action space so that participants and agents must discriminate causally relevant roles from irrelevant ones.

In [Section 7.4](#), the mapping from surface labels (e.g., “color,” “shape”) to property roles is randomized across participants and held fixed within a participant. This prevents surface-label familiarity (such as a prior association between “color” and causal centrality) from biasing which properties are probed first or trusted more. Any aggregate effect of role identity on participant behavior therefore reflects the environment’s causal structure rather than label priors. In [Section 7.5](#) role identity is fixed across seeds since the simulated agent has no surface-label priors to control for.

Representation granularities. Each fundamental value x is observed through one of three discretization maps,

$$\phi_{\text{fine}}(x) = \lceil x/2 \rceil \in \{1, \dots, 8\}, \quad \phi_{\text{medium}}(x) = \lceil x/4 \rceil \in \{1, \dots, 4\}, \quad \phi_{\text{coarse}}(x) = \begin{cases} \text{Low}, & x \leq 8, \\ \text{High}, & x > 8. \end{cases}$$

The granularity $\ell \in \mathcal{L} = \{\text{coarse}, \text{medium}, \text{fine}\}$ is chosen by the learner on every step and determines only how effects are reported; the underlying propagation in the SEM always uses fundamental values.

Mechanisms. All edges in the SEM are linear with additive effect noise,

$$x_{\text{child}} = e_{\text{child}} + w \cdot x_{\text{parent}} + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma^2).$$

The exogenous component e_{child} is back-solved from the task’s initial state so that pre-intervention observations are consistent with the SEM. The noise scale σ is set per observation granularity as described in the simulation protocol below.

Action space and task model. On every step the learner first selects an observation granularity $\ell \in \mathcal{L}$, then conditional on ℓ chooses a variable-role pair $(v, r) \in \mathcal{V}_\ell \times \mathcal{R}$ to intervene on. The joint candidate space is $\Sigma_\ell = \mathcal{L} \times (\mathcal{V}_\ell \times \mathcal{R})$. An intervention sets $v.r$ to a learner-specified fundamental value $x \in \{1, \dots, 16\}$. The SEM propagates this value through its edges (with additive Gaussian noise per edge) to all downstream variables, and the learner observes each affected child through ϕ_ℓ . The running task model M_t is a directed graph over visible variable-role pairs with confidence-weighted edges derived from accumulated evidence.

D.2.2 Experiment 1: human study

The human experiment uses a fixed eight-variable, nine-edge causal graph. [Table D.1](#) lists the full edge set with mechanism class, weight, and sign. The four-task curriculum with visible edges and goals is given in [Table D.2](#); tasks share variables across the curriculum so that overlapping windows expose the same underlying SEM through different visible subsets, supporting the cross-task curation analyses of [Section D.3.5](#).

Table D.1: Ground-truth SEM for Section 7.4. Eight latent variables, nine directed edges with weights and signs. Edges 1-8 appear in at least one task window; edge 9 ($V_3 \rightarrow V_7$) is never jointly observable.

#	Edge	Weight	Sign
1	$V_1.\text{strong} \rightarrow V_2.\text{strong}$	0.70	+
2	$V_1.\text{strong} \rightarrow V_3.\text{strong}$	0.60	+
3	$V_2.\text{strong} \rightarrow V_4.\text{strong}$	0.70	+
4	$V_3.\text{strong} \rightarrow V_4.\text{strong}$	0.50	+
5	$V_4.\text{strong} \rightarrow V_5.\text{strong}$	0.70	+
6	$V_5.\text{strong} \rightarrow V_6.\text{weak}$	0.85	+
7	$V_5.\text{strong} \rightarrow V_7.\text{weak}$	-0.70	-
8	$V_6.\text{strong} \rightarrow V_8.\text{strong}$	0.85	+
9	$V_3.\text{strong} \rightarrow V_7.\text{weak}$	0.50	+

Table D.2: Task curriculum for Section 7.4. Four tasks share variables across consecutive windows to enable cross-task transfer. Each task exposes a different structure (fork, merge, chain) and provides two planning goals reachable in ≤ 5 steps.

Task	Variables $\mathcal{V}_t$	Visible edges	Structure
T_1	$\{V_1, V_2, V_3\}$	$V_1 \rightarrow V_2, V_1 \rightarrow V_3$	Fork
T_2	$\{V_2, V_3, V_4, V_5\}$	$V_2 \rightarrow V_4, V_3 \rightarrow V_4, V_4 \rightarrow V_5$	Merge + chain
T_3	$\{V_4, V_5, V_6, V_7\}$	$V_4 \rightarrow V_5, V_5 \rightarrow V_6(+), V_5 \rightarrow V_7(-)$	Fork with negative edge
T_4	$\{V_5, V_6, V_7, V_8\}$	$V_5 \rightarrow V_6(+), V_5 \rightarrow V_7(-), V_6 \rightarrow V_8$	Deep chain

D.2.3 Experiment 2: simulation

Random graph per seed. Section 7.5 replaces the fixed eight-variable graph with a ten-variable graph randomly sampled per seed. For each of 30 seeds we draw a SEM over $n_{\text{var}} = 10$ variables with edge probability $p_e = .25$, instantiated under the same SEM mechanism (linear edges, additive Gaussian noise) and observed through the same ϕ_ℓ maps as the human experiment. This avoids any dependence of the headline finding on a single hand-designed graph and tests whether the granularity and source mechanisms identified in the human study still dominate when the structure is unfamiliar at every seed.

Procedural curriculum. Each seed’s curriculum is generated procedurally rather than hand-curated. Five training tasks are followed by five held-out test tasks, each task exposing an overlapping window of 4-7 variables drawn from the seed’s graph. Visible edges in a task are the induced subgraph on its visible variables. Consecutive tasks share variables and edges, supporting the same cross-task transfer analyses as the human curriculum. Per-task generation parameters (window-size distribution, difficulty assignment, observation budget) are documented in Section D.4.

D.3 Human study supplementary results

This appendix documents the human study reported in Section 7.4, including recruitment, protocol, and the analyses summarized in the main text.

D.3.1 Participants and protocol

Participants. We recruited $n = 70$ adult participants (50% female; age range 18-40, $M = 30.68$, $SD = 6.27$) through Prolific. All participants provided informed consent and received compensation above the local minimum wage, calibrated to the median completion time of the session. No demographic exclusions were applied beyond Prolific’s baseline eligibility filters.

Procedure. Each session proceeded through three phases: consent and instructions with a worked-example walkthrough of the intervention interface and the graph editor; the four-task sequence $T_1 \rightarrow T_4$ with a

short break between tasks; and a debrief with completion code. Each task T_i has three phases. An exploration phase allowed participants to select any ℓ followed by any (v, r) for up to twenty steps. A reporting phase asked them to submit a causal graph over the visible variables using a drag-and-drop editor. A test phase evaluated two goal-directed tasks against the reported graph. Throughout, participants saw only the granularity they selected; they were never shown the underlying fundamental values.

Data collected and quality filters. For each step we logged the chosen granularity ℓ , the chosen $(v, r) \mid \ell$, the observed effects at granularity ℓ , and timing. At task end we logged the participant-reported causal graph from the editor. For the action-prediction analysis (Section D.3.3), we retained only steps in which all alternatives in Σ_t were validly admissible and all three scoring rules returned finite scores under the participant's current belief state.

D.3.2 Level-selection policies

Policy clusters. We cluster participants by their granularity-use fraction vector $(p_{\text{fine}}, p_{\text{medium}}, p_{\text{coarse}})$ pooled across tasks, using k -means with k chosen by silhouette score over $k \in \{2, \dots, 6\}$. The silhouette maximum is at $k = 3$ (silhouette .494), and the three centroids correspond to fine-, medium-, and coarse-leaning policies. Within-participant consistency (the rate at which a participant's per-task dominant granularity equals their across-task dominant granularity) is .545 vs. participant-specific chance .428 (paired t -test $p < .001$); only 44.8% of dominant-granularity transitions across tasks are switches. Participants thus adopt a stable representational policy that survives task changes rather than re-deriving granularity choice from scratch.

Stickiness. At the step granularity, the participant-specific chance rate for "same granularity as previous step" is $\sum_{\ell} \hat{p}(\ell)^2$, where $\hat{p}(\ell)$ is the participant's empirical granularity frequency. Participants stay at the same granularity on 67% of consecutive steps versus a 46% shuffle baseline (excess +20.5%; paired $t = 13.24$, $p < 10^{-6}$); 65/70 participants are stickier than their own baseline, ruling out a small-subset driver. This is the step-granularity evidence behind the structured-policy claim in the main text.

Next-granularity predictors and source coupling. We fit a multinomial logistic regression of current granularity on (previous granularity, whether an effect was observed on the previous step, step number, task index), with cluster bootstrap by participant for inference. Previous granularity dominates the LR-test deviance contribution (6.12, $p < .001$); previous-step effect observed (.13) and step number (.06) are not significant after controlling for previous granularity. The gap between previous granularity and evidence-based predictors is the quantitative basis for the policy-not-evidence interpretation in the main text. For consecutive steps that share a (v, r) source, the same-granularity rate is .51 vs. participant-specific chance .46 (Wilcoxon signed-rank $p < .05$), and $P(\text{same source} \mid \text{consecutive same-granularity steps}) = .186$ vs. chance .190 ($p = .79$). Both effects are small and the second is null, ruling out source-locked sweeps as the mechanism behind granularity persistence.

D.3.3 Action prediction

Decomposed action space. At each step a participant makes two decisions, a granularity choice $\ell \in \mathcal{L}$ specifying the representation granularity and then a source choice $(v, r) \in \mathcal{S}_t$ specifying which variable-property pair to intervene on. We argue that the two decisions are governed by different computational objectives, and we evaluate this claim by scoring each decision under separate models.

Closed-vocabulary specialization of $\widehat{V}_t$. In Sections 7.4 and 7.5 the candidate model Σ is finite (variable-property tiles $\mathcal{S}_t$ on abstraction granularities $\mathcal{L}$) and $\mathcal{G}_e$ is enumerable, so the open-vocabulary estimator in Equation (7.8) reduces to its $\lambda = 0$ exact form. Writing σ for a candidate intervention (v, r, ℓ) , $\widehat{V}_t$ becomes

$$\widehat{V}_t(\sigma) = \sum_{g \in W_t(\sigma)} w_g (\widehat{p}_t(g \mid \widehat{G}_t \oplus \sigma) - \widehat{p}_t(g \mid \widehat{G}_t)), \quad (\text{D.1})$$

where $\widehat{G}_t$ plays the role of M_t (the participant's running causal belief graph) and $\sigma \oplus \widehat{G}_t$ folds the evidence from intervention σ into the belief. Because edges in this environment are deterministic conditional on the prior $p_e = .15$, the achievement-rate gain $\Delta \widehat{p}_t$ collapses to an edge-prior-weighted count of newly plannable goals. Under this deterministic-edge limit, Equation (D.1) coincides with the channel mutual information $\text{MI}(\text{do}(\sigma); \Gamma_e(M_{t+1}) \mid \widehat{G}_t)$, which is why the rules below are named RepEmp.

Scoring rules. We define four scoring rules, each returning a real-valued score per alternative.

1. Environmental empowerment for source is the MI between intervening on (v, r) and the resulting downstream state under the current belief, $\text{EnvEmp}_{\text{src}}(v, r) = \text{MI}(\text{do}(v, r); X_{\text{desc}(v, r; \widehat{G}_t)} \mid \widehat{G}_t)$. Under deterministic edges with known structure, this MI is order-preserving in the number of variable-property pairs downstream of (v, r) in $\widehat{G}_t$. Our implementation uses this count as the score, predicting exploitation: choose sources with known downstream paths.
2. Representation empowerment for source is the MI between intervening on (v, r) and the resulting goal-achievement signature, evaluated under the belief, $\text{RepEmp}_{\text{src}}(v, r) = \text{MI}(\text{do}(v, r); \Gamma_e(M_{t+1}) \mid \widehat{G}_t)$. For an untested source, enumerating candidate new edges $(v, r) \rightarrow (v', r')$ at edge prior $p_e = 0.15$ yields a witness-set sum identical in form to Equation (7.8). Re-testing a tested source has zero grounded gain under deterministic-edge beliefs. The rule predicts exploration: choose untested sources with high bridging potential.
3. Environmental empowerment for granularity is the channel capacity of ϕ_ℓ under a uniform prior, $\text{EnvEmp}_\ell(\ell) = \text{MI}(X; \phi_\ell(X)) = \log_2 B(\ell)$, where $B(\ell) \in \{2, 4, 8\}$ is the bin count. The rule always predicts fine.
4. Representation empowerment for granularity is the MI between a causal effect ΔX and its granularity- ℓ observation, capped at the SEM's signal bandwidth B^* (in bits) with an over-resolution penalty when $\log_2 B(\ell)$ exceeds B^* . We set $B^* = 2$, the resolution at which SEM effects are reliably detectable above observation noise, and apply a unit penalty for capacity wasted on bins finer than the signal can populate.

Baseline models. In addition to the four rules above, we evaluate three baselines that score the joint action space $\mathcal{S}_t \times \mathcal{L}$ with a single rule. *Joint environmental empowerment*: $\text{EnvEmp}(v, r, \ell) = \text{EnvEmp}_{\text{src}}(v, r) \times B(\ell)$. *Joint information gain*: expected reduction in entropy over $\widehat{G}_t$, weighted by a detection factor that mildly favors coarse. *Joint novelty*: binary score $\mathbb{I}[(v, r) \notin \mathcal{T}_t]$. All three treat source and granularity as a single flat decision space.

Factored model. The factored model decomposes the joint choice into a source component scored by environmental empowerment and a granularity component scored by representation empowerment:

$$P_{\text{factored}}(v, r, \ell) \propto \exp\left(\beta_s \widetilde{\text{EnvEmp}}_{\text{src}}(v, r \mid \ell) + \beta_\ell \widetilde{\text{RepEmp}}_\ell(\ell)\right),$$

where $\widetilde{f}(\cdot)$ denotes min-max normalization within the step, putting the two components on a comparable scale before they enter the softmax. This form encodes the hypothesis that source and granularity decisions are driven by different objectives and that no single scoring rule applied to the flat action space Σ_t captures both. The latter claim is tested against the EnvEmp-alone and RepEmp-alone rows in Table 7.1.

Normalized rank and cross-validated ΔLL . For each step we compute the normalized rank of the participant’s chosen action under each rule’s score over the joint 27-action space, $\text{nr} = (\text{rank} - 1) / (|\mathcal{S}_t \times \mathcal{L}| - 1)$, with ties broken at the midrank. A normalized rank of 0 is a perfect prediction; .5 is the chance level under a uniform alternative. We turn each rule’s scores into a softmax choice distribution with a per-participant inverse-temperature β (for the factored model, a shared β scales the combined score). β is fit by maximum likelihood on three of the four tasks and the held-out task is scored under the fitted β ; we cycle the held-out task to obtain a per-participant cross-validated log-likelihood. Reported ΔLL is the held-out log-likelihood minus the uniform baseline summed over participants.

D.3.4 Compression-dominated dynamics

Within-task granularity dynamics. Per-environment dynamics disaggregate the within-task granularity profile by environment. The same qualitative pattern holds across all four tasks: participants open with a fine-leaning mix—exploring the high-resolution variable/property tiles to see what each manipulation does—and then settle onto medium as the dominant granularity, with fine use declining and coarse use creeping up. The crossover is visible in every environment, including the shorter tasks where the aggregate signal is weaker, and is most pronounced in the longer tasks (T_3 - T_4) where there is more room for the abstraction to consolidate. Per-task Spearman rank correlations between step index and fine (resp. coarse) use are computed per participant and aggregated to a population correlation with its p -value; participants with too few fine/coarse choices to support a stable correlation are dropped per task.

Refine ratio. For each pair of consecutive steps that share a (v, r) source, we classify the granularity transition as a *refinement* (coarse $\rightarrow$ medium, medium $\rightarrow$ fine, coarse $\rightarrow$ fine), a *simplification* (the reverse direction), or unchanged. The refine ratio is $\# \text{refinements} / (\# \text{refinements} + \# \text{simplifications})$; values below 0.5 indicate compression-dominated drift. Significance is by Wilcoxon signed-rank against 0.5.

Table D.3: Within-task curation cycle.

Each step is labeled as *explore* (first test of a (v, r) source), *refine* (re-test at a different granularity), or *consolidate* (re-test of a source shared with a prior task and previously reliable). Rows are step bins.

Phase	Consolidate	Explore	Refine
Steps 1–5	7%	72%	21%
Steps 6–10	1%	58%	40%
Steps 11–15	1%	46%	54%
Steps 16–20	1%	22%	77%

Drift across the task. Participants begin fine-leaning, with roughly 57% of early actions selecting fine, and shift away over the course of each task. The drift is strongest in the longer tasks. In T_3 , fine use declines (Spearman $\rho = -.60$, $p = .005$) while coarse rises ($\rho = .51$, $p = .022$); in T_4 , fine use also declines ($\rho = -.48$, $p = .033$). Among consecutive same-source steps that change granularity, simplifications outnumber refinements (845 vs. 685; refine ratio 0.448, Wilcoxon $p = .005$). This rules out two alternative accounts. A simplicity prior would predict coarse choices from the start rather than a drift toward them. A pure information-gain account would predict no systematic direction bias after experience accumulates, since at that point both directions resolve uncertainty equally well. The asymmetric drift therefore reflects a goal-projected criterion that becomes more selective once enough structure has been learned to render fine resolution unnecessary, consistent with Proposition D.8.

D.3.5 Curation phase classification

We label each step as one of three within-task curation actions. *Explore*: first test of a (v, r) source within the current task. *Refine*: a re-test of a previously tested (v, r) source at a different granularity. *Consolidate*: a re-test of a (v, r) source that is shared with a prior task and has produced a reliable effect in the running cross-task library. Phase counts are aggregated by step bin (1-5, 6-10, 11-15, 16-20). Table D.3 reports the within-task phase distribution by step bin: early steps are dominated by exploration of new sources, while later steps shift toward refinement and (for variables shared across tasks) consolidation.

D.4 Matched-simulation supplementary results

D.4.1 Simulation protocol

Environment variant. The simulation uses the same modeling primitives as Section 7.4—variables with three property roles (strong, weak, noise), the three discretization maps ϕ_ℓ , and linear-effect SEM mechanisms with additive Gaussian noise—but replaces the fixed eight-variable curriculum with a larger, randomly sampled graph in order to (i) prevent the result from depending on a single hand-designed structure and (ii) stress-test the mechanism in a regime that is denser and combinatorially harder than the human one. For every seed we sample a fresh DAG over $n_{\text{var}} = 10$ variables with edge probability $p_e = .25$. The full graph is never observed at once; each task exposes an overlapping window of 4-7 variables drawn by a procedural curriculum generator, so consecutive tasks share variables and edges. This multiplies the joint hypothesis space relative to the human experiment and tests whether the granularity and source mechanisms identified there still dominate when the structure is unfamiliar to the agent at every seed.

Agent configurations. All conditions share the same components, namely a Bayesian belief tracker over the causal graph (initialized with edge prior $p_0 = .3$), an actor that selects interventions with 5% ϵ -greedy uncertainty-based exploration, and the same total budget of 1,000 interventions per task (10 episodes $\times$ 100 steps). Each step, the curator proposes a representation granularity and a goal; the actor then executes interventions and updates its belief. The belief tracker maintains a probability for each directed edge between visible variable-feature pairs, updated via Bayesian inference from observed intervention effects.

Conditions. Each condition pairs a granularity rule with a source rule, both reusing the scoring rules defined in [Section D.3.3](#).

Source-rule comparison.

- **Factored.** Source scored by the human-experiment hybrid rule, $\alpha \text{EnvEmp}_{\text{src}} + (1 - \alpha) \text{RepEmp}_{\text{src}}$ with $\alpha = .5$ and per-step min-max normalization.
- **RepEmp.** Source scored by the closed-vocabulary $\text{RepEmp}_{\text{src}}$ in belief mode at the current edge prior.
- **EnvEmp.** Source scored by $\text{EnvEmp}_{\text{src}}$, the deterministic-edge count of reachable variable-role pairs.

Information-gain and random baselines.

- **Bayesian IG.** Source chosen to maximize the expected reduction in entropy over edge probabilities of the current belief, jointly optimized with granularity. For each candidate granularity, the IG score is capped by $\log_2(\text{feature_dim})$ per outgoing edge to reflect the granularity's resolving power, and the granularity with highest average capped IG is selected. This is the canonical structure-learning baseline, lifted to the granularity-aware setting.
- **Novelty IG.** Source chosen to maximize an inverse visit-count bonus $\log_2(n + 2) - \log_2(n + 1)$ over the target variable, with the same granularity-aware joint optimization and granularity cap as Bayesian IG.
- **Random.** Source drawn uniformly at random under a fixed representation. No-strategy baseline.

D.4.2 Evaluation and transfer

Sample sizes. Each condition is evaluated on 150 training-task instances (5 training tasks $\times$ 30 graph seeds) plus 150 held-out test-task instances (5 test tasks $\times$ 30 seeds) on the same per-seed graphs. Conditions, seeds, and tasks are crossed, so all condition-vs-condition comparisons are paired at the (seed, task) granularity.

Graph recovery metrics. At each logged step we compare the learner's belief graph to the seed's ground-truth DAG and report four metrics: precision $\text{TP}/(\text{TP} + \text{FP})$, recall $\text{TP}/(\text{TP} + \text{FN})$, F1 (harmonic mean), and normalized SHD $((\text{FP} + \text{FN})/|E_{\text{gt}}|)$. An edge is counted as learned if its posterior probability under the learner's belief tracker exceeds .5. Because each condition operates at a chosen representation granularity ℓ , comparing learner edges against the obs-granularity ground truth would penalize coarser conditions for failing to recover detail their representation cannot express. We therefore evaluate every condition in its own curator representation space. The obs-granularity ground-truth edge set is projected through ϕ_ℓ to the corresponding granularity- ℓ feature space (a many-to-one map between obs-feature indices and the $B(\ell)$ -bin

Table D.4: Identifiability of ground-truth edges across granularities (135 edges). *medium* maximizes identifiability not because it has the highest resolution but because it is bandwidth-matched to the causal signal.

Level	% Identifiable	Avg. effect probability	Avg. discrimination
<i>fine</i>	2.2%	.96	−.03
<i>medium</i>	71.1%	.71	−.04
<i>coarse</i>	44.4%	.44	−.06

granularity features), with self-loops induced by the projection collapsed. The learner’s belief is compared directly against this projected ground truth. We never up-project the learner’s belief to obs, which would invent unsupported detail.

Generalization metrics. Transfer is tested at the boundary between tasks within a seed. After the training sequence, the learner’s full state—belief graph and library—is frozen and carried into the five held-out test tasks (sampled from the same per-seed graph by the same procedural generator), so test tasks may share variables and edges with the training tasks. The learner re-runs the same intervention budget on each test task starting from this frozen state, and we compute the per-task F1 as before. The transfer test compares first-task to last-task F1 within a condition by paired *t*-test across the 30 seeds: a positive slope is evidence that the representation generalizes.

Statistical comparisons. For each metric we run paired *t*-tests across matched (seed, task) instances, taking the lowest-F1 condition as the reference. To control the family-wise error rate over the 9 pairwise contrasts within a study we apply a Bonferroni correction ($\alpha_{\text{fwe}} = .05/9$). Effect sizes are reported as Cohen’s *d* on the paired differences, and we accompany each test with the 95% confidence interval on the mean difference. Effect-decomposition tables report the F1 increment of each condition over the random baseline as both an absolute difference and a percentage of the total ΔF1 between baseline and the best condition.

D.4.3 Mechanistic grounding

The F1 differences across observation granularities in Table 7.2 are not a behavioral artifact of the grounded estimator $\widehat{V}_t$. They reflect a structural property of the environment that the estimator successfully tracks. We make this explicit through two analyses on the same 30-graph evaluation set.

Identifiability. For each edge (X_i, X_j) in the ground-truth graph, we compute whether the edge is theoretically identifiable at granularity ℓ , that is, whether intervening on X_i produces a distribution over X_j under ϕ_ℓ that is statistically distinguishable from the no-edge distribution under the noise scale σ_ℓ . An edge is counted as identifiable if a two-sample test between the two distributions rejects at $\alpha = .05$ given the simulation’s noise level.

The near-zero identifiability at *fine* explains the F1 ceiling for the EnvEmp model. No exploration strategy can recover edges that are not statistically distinguishable from no-edge under the observation noise. The *medium* granularity maximizes identifiability not because it has the highest resolution but because it is bandwidth-matched to the causal signal: fine enough to preserve distinct effects, coarse enough that noise does not dominate. The RepEmp estimator of Equation (7.5), without access to ground truth, selects this granularity.

Phase	Granularity (EnvEmp $\rightarrow$ RepEmp)	Source (RepEmp $\rightarrow$ Factored)
Early (steps 1–100)	+0.073	−0.093
Late (steps 900–1000)	+0.138	+0.014

Table D.5: Phase decomposition of the F1 differential between EnvEmp and RepEmp at the granularity slot, and between RepEmp and Factored at the source slot.

Learning dynamics. The granularity effect grows over the course of learning, and the source effect emerges only after the granularity has stabilized. Two patterns are worth noting. First, the granularity advantage roughly doubles from early to late learning, mirroring the within-task compression dynamics in the human data: both humans and the simulated agent benefit increasingly from operating at the goal-sufficient granularity as evidence accumulates, consistent with [Proposition D.8](#). Second, the source effect provides no benefit early (−0.093) but a small positive benefit late (+0.014), suggesting that smarter exploration over sources only helps once the representation itself is stable.

D.5 Open-vocabulary Curator-Actor implementation and results

This appendix documents the open-vocabulary experiment of [Section 7.6](#). [Section D.5.1](#) specifies the Curator-Actor implementation in PDDL. [Section D.5.2](#) describes the BabyAI and Zelda environments. [Section D.5.4](#) details the baselines (PPO, Motif, WorldCoder, Hoarder). [Section D.5.7](#) reports the per-environment library size. [Sections D.5.5](#) and [D.5.6](#) report the backward and held-out transfer trajectories.

D.5.1 Curator-Actor implementation

Library and task model. The persistent library $\mathcal{L}$ is a typed PDDL knowledge store with types, type-hierarchy edges, predicates, and operators. Types are atomic symbols (e.g., agent, key, door, object). Type-hierarchy edges record subtype relations (key $\rightarrow$ object, door $\rightarrow$ object). Predicates carry a PDDL signature and a Python grounding closure that evaluates the predicate on raw observations:

```
(:predicates
  (holding ?a - agent ?o - object)
  (next_to ?a - agent ?o - object)
  (locked ?d - door)
  (is_key_for ?k - key ?d - door))
```

Operators are PDDL action schemas with typed preconditions and effects:

```
(:action unlock_door
  :parameters (?a - agent ?k - key ?d - door)
  :precondition (and (holding ?a ?k)
                     (next_to ?a ?d)
                     (locked ?d)
                     (is_key_for ?k ?d))
  :effect       (and (not (locked ?d))))
```

Every element carries a status flag in $\{\text{candidate}, \text{trusted}\}$. The task-local model M_t is the PDDL domain assembled from the trusted subset of $\mathcal{L}$ relevant to the current environment, plus any candidate elements currently under evaluation, plus the grounding code needed to evaluate predicates on raw observations.

Curator. The Curator is the LLM-backed component, running the loop in Section 7.3.4. *Propose.* From the experience buffer $\mathcal{D}_t$, the current M_t , and $\mathcal{L}$, an LLM proposer emits candidate types, predicates (with Python grounding code), and operator schemas. The proposer also revises existing elements (refine signature, split, compose) and emits top-environment goal predicates that are expressible under $M_t \cup \Sigma_t$ but not currently achievable under M_t . *Probe.* For each candidate a , the Curator constructs a probe goal set $\mathcal{G}_t(a)$: goals blocked under M_t but expressible under $M_t \oplus a$ (positive probes) and goals already solvable that act as regression checks (negative probes). For each $g \in \mathcal{G}_t(a)$, the Curator compiles g into a PDDL problem against M_t and calls Fast Downward (Helmert, 2006). When the PDDL compile fails (e.g., a candidate operator has malformed typing) or the external solver times out, the Curator falls back to a fuzzy beam-search planner that operates directly on the operator schemas in M_t . In both cases the output is an operator sequence $\pi_g^{M_t} = (a_1, \dots, a_k)$, which is handed to the Actor along with g . *Score.* Candidates are scored by the open-vocabulary estimator of $\widehat{V}_t$ in Equation (7.8). The probabilities $\widehat{p}_t$ are empirical achievement rates of Γ_e on the probe set returned by the Actor, and $W_t(a)$ is the set of currently blocked witness goals that adding a could unlock. To save Actor calls, candidates are first pre-ranked by a cheap surrogate that estimates the same gain from $\mathcal{L}$, $\mathcal{D}_t$, and the candidate's typed signature; only top-ranked candidates have their probe goals dispatched to the Actor. *Curate.* Candidates that meet minimum thresholds on $\widehat{V}_t$, on reliability (success rate across probe attempts), and on a minimum test count are promoted to trusted. At the cross-environment refactor stage, the same $\widehat{V}_t$ is re-evaluated against the cross-environment probe buffer using the `imagined_goal` estimator that samples a fixed budget of imagined state-goal pairs from the buffer and scores reachability under $\mathcal{L}$ vs. $\mathcal{L} \cup \{\sigma\}$. The trusted set is greedily re-ranked under the library budget by marginal $\widehat{V}_t$.

Actor and local world model. The Actor receives a $(M_t, g, \pi_g^{M_t})$ triple from the Curator and returns evidence about $\Gamma_e(M_t)_g$. No LLM is involved at plan-and-execute time, and the Actor never plans symbolically; its job is to realize the operator sequence $\pi_g^{M_t} = (a_1, \dots, a_k)$ in primitive actions and report whether each a_i landed its effects.

The Actor grounds each operator into a primitive sequence by heuristic search over the agent's primitive action set. The search uses an agent-egocentric local world model, a small representation (`patch`, `carried_slot`) of the agent's neighborhood, where `patch` is a fixed-radius window of cells rotated so the agent heading is up, and `carried_slot` encodes the carried object outside the cell grid. Predicates that bind to in-window objects, plus agent-indexed predicates (e.g., `holding`, `facing`), are projected into the local model on demand by running their grounding closures.

The search heuristic counts unmet operator preconditions under the projected predicates; this is admissible for conjunctive precondition sets with unit primitive costs. The local model is learned online: a self-supervised predictor maps (`patch`, `carried_slot`, `primitive`) to the local predicate-flip set, trained from the Actor's own primitive trajectories in an experience buffer. Calls into the live environment during search

are amortized by rollouts in this learned model. Every primitive that has been executed contributes a training example, so the local model improves monotonically with interaction without any extra environment budget.

D.5.2 Environments

BabyAI. We use four BabyAI environments for training, ordered by approximate task complexity so that later environments involve longer action sequences than earlier ones: `GoToLocal`, `Pickup`, `OpenRedDoor`, and `UnlockLocal`. All four share the same primitive action space (turn, move, pickup, drop, toggle) but differ in the objects present, the preconditions for goal achievement, and the goal templates. The ordering gives the Curator opportunities to compose elements promoted at earlier environments (e.g., navigation routines reused for key acquisition before unlocking), though the curriculum does not enforce any specific decomposition.

A held-out split of four environments tests compositional transfer to configurations whose goal templates differ from the training environments. `PutNextLocal` introduces spatial object-object relations, `MoveTwoAcrossS5N2` requires multi-object manipulation, `PickupAbove` changes the relative orientation of agent and target, and `Unlock` is a variant of `UnlockLocal` with different layout and object constraints.

Zelda. We use the Zelda game from the EMPA suite (Tsividis et al., 2021), which provides typed objects (player, monster, key, door, wall) and a native reward function. The first three environments are used for training and the last two are held out for evaluation. Held-out environments share the primitive dynamics of training but introduce new layouts and precondition configurations, testing whether predicates and operators promoted during training (e.g., key acquisition, monster avoidance) transfer to settings where the same goals must be reached through different spatial arrangements.

D.5.3 Training and evaluation

Curriculum and budget. Each training run specifies an ordered curriculum of environments and a per-environment interaction budget. Within an environment, the Curator-Actor loop runs until either the candidate set stabilizes (no new promotions for $K_{\text{stab}} = 3$ consecutive cycles) or the per-environment LLM-call budget is exhausted. We allocate 200 LLM calls per environment on BabyAI and 400 on Zelda, matching WorldCoder’s budget on each family. Motif’s pipeline issues an LLM call for every labeled trajectory pair and consequently uses $\sim 5,000$ calls per environment; this is a structural cost of preference labeling rather than a compute advantage. PPO has no LLM calls and is matched to Motif on environment-step budget.

The LLM-call budget translates into a small environment-step footprint. Curator and WorldCoder each consume on the order of thousands of primitive environment steps per environment, two orders of magnitude below the $\sim 500k$ and $\sim 1m$ steps used by Motif and PPO, because each LLM call drives at most a handful of probe-goal executions rather than continuous policy rollout. We treat this gap as a property of symbolic methods rather than a tuning advantage but flag it explicitly so the transfer comparisons are not read as unequal-compute results. Immediately after an environment finishes training, we record a self-evaluation score on

the same environment using the then-current $\mathcal{L}$; this score anchors the backward-transfer comparison.

Promotion criteria. A candidate element σ is promoted to `trusted` when it meets three thresholds simultaneously: reliability $\geq .70$ (success rate of σ 's witness goals across probe attempts), $\widehat{V}_t \geq .50$ (the coverage-gain proxy of Equation (7.8)), and a minimum test count of ≥ 3 probe dispatches. The $\widehat{V}_t$ threshold operationalizes the cardinality regularizer λc_σ in Equation (7.8): with $c_\sigma \equiv 1$ and $\lambda = .50$, accepting iff $\widehat{V}_t > .50$ is equivalent to accepting iff the penalized score $\widehat{V}_t - \lambda c_\sigma > 0$. At the environment boundary, $\widehat{V}_t$ is re-evaluated against a cross-environment probe buffer using the `imagined_goal` estimator. We sample $N_{\text{imag}} = 50$ state-goal pairs uniformly from the buffer, plan against $\mathcal{L}$ and $\mathcal{L} \cup \{\sigma\}$ for each, and set σ 's contribution equal to the number of newly reachable goals.

Forward, backward, and held-out transfer. For each adjacent pair $(\text{env}_i, \text{env}_{i+1})$ in the training curriculum, we freeze $\mathcal{L}$ at the end of env_i and evaluate it on env_{i+1} without further library updates: the agent receives no training trajectories on env_{i+1} and the Curator does not propose new candidates. The local world model continues to update from primitive trajectories at no extra environment budget. This isolates the short-range generalization of promoted symbols. Backward transfer is the difference between an environment's success rate immediately after it was trained and its success rate after all subsequent training has finished. Negative values indicate forgetting; positive values indicate retroactive improvement, where abstractions promoted at later environments make earlier environments easier. Held-out transfer freezes the final $\mathcal{L}$ and evaluates zero-shot on disjoint environments not used during training. No training trajectories, no candidate proposals; only library lookup and Actor execution.

LLM and serving. All LLM-backed components (Curator's proposer, scorer, and curator; WorldCoder's synthesizer; Motif's labeler) use the same model: `deepseek-r1:70b` served locally via Ollama on a single A100 GPU. Temperature is fixed at 0.6 for proposal generation and 0.0 for scoring/curation calls. We use the same model across baselines to control for proposer quality; differences in performance reflect how each method uses the LLM, not which LLM each method has access to. We report mean and standard deviation across three seeds per condition, with seeds controlling LLM sampling and environment stochasticity. Within a seed, evaluation is averaged over $N_{\text{eval}} = 50$ rollouts per (environment, stage) cell on BabyAI and Zelda. The success metric, `reward_success`, is the fraction of evaluation rollouts in which the agent reaches the environment's primary goal as defined by the environment's native reward function. A rollout terminates on goal achievement, agent death (Zelda), or the step limit ($T_{\text{max}} = 256$ primitive steps).

D.5.4 Baselines

PPO. We use Proximal Policy Optimization (Schulman et al., 2017) with intrinsic motivation as a model-free continual RL baseline. The agent is trained with both extrinsic task reward and an intrinsic novelty bonus from Random Network Distillation (Burda et al., 2018), which rewards visiting states whose features the predictor network cannot yet match. For intrinsic-reward selection we varied three conditions (no intrinsic, count-based, RND-based) and reported the RND variant since it performs

best across the three. The agent follows the same sequential curriculum as all other baselines: a single policy is trained on each environment in order for a fixed step budget, then frozen for cross-environment evaluation before moving to the next environment. The policy network is a three-layer CNN (channels 16-32-64, kernel size 2) followed by an AdaptiveAvgPool2d(4×4) that produces a fixed 1024-dimensional feature regardless of input resolution, then a 64-unit fully connected layer feeding separate actor and critic heads. For both BabyAI and EMPA the observation is the symbolic full-observation grid (object type, color index, state), with per-channel normalization matching the integer ranges of each field. PPO hyperparameters follow standard settings: clipped surrogate objective ($\epsilon = .2$), GAE advantage estimation ($\gamma = .99, \lambda = .95$), four optimization epochs per rollout batch, and linear learning-rate annealing within each environment phase.

Motif. Motif (Klissarov et al., 2023) constructs an intrinsic reward by eliciting trajectory preferences from an LLM and training a reward model on those preferences. The pipeline runs four steps per curriculum environment and follows the same sequential continual-learning structure as PPO. *Collect.* A fresh RND agent (independent of the Motif policy) explores the current environment and saves the resulting episodes. *Label.* 5000 trajectory pairs are sampled from those episodes, converted to a text caption listing the objects the agent sees, their distances, and the chosen action at each step, and sent to an LLM. The LLM returns a binary preference ($A > B$, $B > A$, or $A = B$) without access to the mission string. *Train reward model.* An MLP reward model is trained from scratch on the labeled pairs using a Bradley-Terry preference loss. *RL with intrinsic reward.* The policy is trained with PPO using the reward model’s scalar output as the intrinsic bonus, together with extrinsic task reward.

Why Motif fails in our setting. Motif was originally demonstrated on Crafter, where the LLM can assess behavioral quality from universal, goal-agnostic signals (health loss, resource gain, crafting events) without knowing the specific task. Our experimental setting deliberately withholds direct optimization for extrinsic missions or rewards from the agent during training, because an agent that generalizes across open-ended task distributions must not be hard-wired to any particular goal (Colas et al., 2022). An agent that relies on task-specific reward shaping to learn a behavior will simply relearn that behavior from scratch when the goal changes, and the knowledge it acquires is goal-indexed rather than compositional. Motif’s LLM-labeling approach is a natural attempt at task-agnostic intrinsic motivation but cannot produce a useful signal here. The LLM has no principled basis for ranking one trajectory over another since every primitive action is potentially correct depending on a goal it does not know. The reward model consequently achieves 50.2% preference-prediction accuracy on a held-out validation split, indistinguishable from chance, and all downstream policy evaluations yield the lowest performance across training environments.

WorldCoder. We run WorldCoder (Tang et al., 2024) from its public release with no algorithmic changes. Our only modifications are swapping the OpenAI backend for the local Ollama server serving deepseek-r1:70b (the same backbone used by the Curator and Motif), and wrapping the per-episode driver in a sequential-curriculum loop so that the same step budget, evaluation cadence, and held-out splits used for PPO and Motif apply.

WorldCoder maintains its world model as two pieces of Python code: a transition function `transit(state, action)` that returns the next

state, and, for each mission string seen so far, a reward function `reward(old_state, action, new_state)` returning `(reward, done)`. The state passed into both functions is the same object-list representation the BabyAI/EMPA wrappers expose, so the generated code reads symbolic fields directly rather than pixel arrays.

The synthesized code (transition function plus the dictionary of permission reward functions) is carried over verbatim between curriculum environments. Only the experience buffer and the `mission_accomplished` set are cleared at the environment boundary. When a new environment introduces transitions inconsistent with the carried-over transition function, the LLM edits that single Python file in place. There is no separate library object that can prune or quarantine elements, so all prior code paths persist unless the LLM happens to delete them during a repair. This is the mechanism that drives the library-bloat comparison.

Hoarder. The Hoarder ablates the empowerment criterion from the full Curator while keeping its proposer, verifier, and library mechanics intact. Concretely, $\widehat{V}_t$ is replaced by an admit-all rule: every candidate that passes reliability thresholds and basic verification is promoted to trusted, regardless of its coverage gain. The library therefore grows monotonically with proposer volume and the cardinality penalty λ_{C_σ} has no effect. This isolates the empowerment-based selection from the LLM proposer’s volume and quality.

D.5.5 Cross-stage transfer

We evaluate each method’s carried representation at every training stage on every BabyAI environment. On the hardest environment `env3`, the Curator reaches $\sim.30$ success before `env3` training begins and finishes at $\sim.45$ after its own training stage. The library accumulated from `env0`-`env2` already handles a substantial fraction of `env3` zero-shot, because predicates and operators promoted at earlier environments (e.g., `navigate_to`, `pickup`, `toggle`) compose into the longer action chains `env3` requires. WorldCoder, by contrast, starts `env3` near zero, since its mission-specific reward functions don’t compose.

On environments `env0`-`env2`, the Curator preserves earlier-environment success after the matched training stage, and on `env2` shows clear retroactive improvement: predicates promoted at `UnlockLocal` (notably `unlocked(?door)`) tighten operator preconditions on `Pickup` and `OpenRedDoor`, so earlier environments become *easier* as the curriculum progresses. WorldCoder and PPO show the textbook forgetting profile: peak at the matched training stage, then decay (e.g., on `env1`, WorldCoder peaks $\sim.69$ at `+env1` and drops to $\sim.10$ by `+env3` as later edits to the carried code break earlier transitions). The empowerment-ablated Hoarder shows neither forward nor backward gains: it tracks the Curator’s curve at `env0` briefly but flattens out and never recovers on later environments. This isolates the empowerment-based selection criterion as the source of the Curator’s transfer.

The Zelda curriculum shows the same patterns. The Curator transfers forward into `env1` ($\sim.20$ before `env1` training begins, climbing to $\sim.49$ at `+env1` and staying at $\sim.43$ after `+env2`) and into `env2`, while WorldCoder peaks lower ($\sim.36$ on `env1`) and decays. The Hoarder again fails on both axes.

D.5.6 Held-out transfer

On `env4` (`PutNextLocal` simplified), `env6` (`PickupAbove`), and `env7` (`Unlock` variant), the Curator reaches $\sim.45-.75$ success without ever having trained on these configurations, ahead of `WorldCoder` by 15-30 absolute points and ahead of all policy baselines by an order of magnitude. The library composes: predicates promoted at `UnlockLocal` (e.g., `unlocked(?door)`, `is_key_for`) and operators learned at `Pickup/OpenRedDoor` re-bind to new objects and layouts in the held-out environments without re-synthesis.

On `env5` (`MoveTwoAcrossS5N2`), every method drops below .15, including the Curator. The environment requires achieving a conjunction of binary inter-object spatial relations (`next_to(o1, o2)` and `next_to(o3, o4)` simultaneously), but every training-environment goal is unary or agent-indexed. The bottleneck is the LLM proposer’s goal diversity: even when it generates a candidate `next_to(?o1, ?o2)` predicate during training, the witness goals available on training environments never test compositional object-object relations, so the predicate fails the empowerment threshold and is not promoted. The criterion behaves correctly given the probes it sees; the probes simply do not cover the structurally distinct goals `env5` demands.

On both held-out Zelda environments, the Curator reaches $\sim.48$ success and outperforms `WorldCoder` by 8-12 absolute points. Policy baselines remain near floor, and `Hoarder` collapses to $\sim.06$. The smaller margin reflects that Zelda’s training-and-held-out gap is narrower (layout differences with the same primitive dynamics), so `WorldCoder`’s mission-keyed reward functions transfer better than they do on BabyAI’s compositionally distinct held-out environments.

`Hoarder`, which keeps every LLM-proposed element rather than curating, drops to $\sim.03-.06$ on every held-out environment in both families. The Curator’s held-out gains therefore cannot be attributed to the LLM proposer producing useful candidates but to the empowerment-based selection that keeps $\mathcal{L}$ compact and broadly applicable. This is the cleanest piece of evidence that selection, not proposal, is the mechanism behind held-out transfer.

D.5.7 Library compactness and planning load

We measure the carried-forward representation in characters of source code, broken into its two natural components on each side. `WorldCoder` carries a world model (Python transition function) plus a reward model containing one Python reward function per mission seen so far. The Curator carries the final PDDL domain file the planner compiles against, plus the grounding code attached to each promoted predicate.

Table D.6 reports both decompositions per environment, mean over seeds. `WorldCoder`’s reward model grows monotonically across the four BabyAI training environments, taking the reward-side character count from 6.2 KB to 20.4 KB (3.3 $\times$). The world-model side stays around 5 KB throughout, since the transition function is edited in place rather than appended to. Total carried representation grows from 11.2 to 24.7 KB. The Curator’s carried representation stays small and roughly flat. Across the EMPA/Zelda environments the symbolic side is 2-3 KB and the grounding code is 2-4 KB, for a total of 2-6 KB per environment, about 4-10 $\times$ smaller than `WorldCoder`’s BabyAI reward bundle and approximately

Table D.6: Carried-forward representation size across training environments. Representation size is measured as source-code characters retained after each environment. WorldCoder size includes its generated world-model and reward code; Curator size includes the PDDL domain and predicate-grounding code used by the planner.

Family	Level	WorldCoder			Curator		
		WM.py	rwd.py	total	PDDL	grnd.py	total
BabyAI	env0 (GoToLocal)	4 982	6 207	11 189	2 811	3 883	6 694
BabyAI	env1 (Pickup)	4 449	12 305	16 754	2 811	3 883	6 694
BabyAI	env2 (OpenRedDoor)	5 218	15 443	20 661	5 475	4 091	9 566
BabyAI	env3 (UnlockLocal)	4 337	20 360	24 697	5 141	4 091	9 232
Zelda	env0 (empa_250)	2 799	1 169	3 969	2 173	2 109	4 283
Zelda	env1 (empa_251)	5 906	2 686	8 592	2 275	2 226	4 501
Zelda	env2 (empa_252)	4 986	3 871	8 857	2 572	1 893	4 465

constant in the curriculum step. The trusted-element counts behind those byte numbers are 5-6 predicates and 0-3 operator schemas per environment. WorldCoder’s growth comes from the fact that each new mission requires a new mission-keyed reward branch, and prior branches cannot be deleted without breaking the planner’s ability to score earlier goals. The Curator carries typed predicates and operator schemas instead, so a new environment reuses the same handful of trusted symbols rebound to new objects rather than appending a new code path.

Additional Material for Chapter 8

E.

The appendix for Chapter 8 is organized as follows:

- ▶ Section E.1 describes the navigation environment design and the map construction procedure used in the experiments.
- ▶ Section E.2 gives the full LLM compilation protocol used to translate natural-language explanations into executable guidance.
- ▶ Section E.3 reports participant recruitment details and the pre-registration information for the human experiments.
- ▶ Section E.4 provides additional details on the utility-guided generation procedure used in Experiment 2.

E.1 Environment design and map construction	269
E.2 Experiment 1: LLM compilation protocol	271
E.3 Experiment 1: Participant details and pre-registration	275
E.4 Utility-guided generation details	277

E.1 Environment design and map construction

E.1.1 Grid world specification

The navigation task is implemented as a discrete two-dimensional grid world loaded from Tiled JSON map files. Overall, we designed 24 maps organized into 12 near-matched pairs. Each pair shares the same broad spatial layout but differs by a localized structural edit, such as adding an obstacle, removing a corridor, or shifting the goal location (Figure 8.3). This pairing design lets us vary local navigational demands while keeping global map structure approximately fixed.

The maps were constructed to satisfy three requirements. First, every map is solvable from start to goal. Second, paths are non-trivial, with shortest paths long enough to require genuine planning. Third, the map set spans a broad range of structural difficulty, from open maps with many recoverable route options to brittle maps with corridors and costly dead ends.

Each map is cropped to the bounding box of occupied floor tiles and then converted into a symbolic grid with four tile types used by the simulation code: free floor ('0'), obstacle or wall ('1'), agent start ('2'), and treasure ('3'). Agent movement is deterministic and four-connected, with action space

$$\mathcal{A} = \{\text{up, down, left, right}\}.$$

Attempting to move into an obstacle or beyond the map boundary leaves the agent in place.

In the simulation environment used for explanation evaluation, rewards are normalized to +1 for reaching the goal and -0.01 otherwise. Episodes terminate when the treasure is reached or when the step budget is exhausted. Because the analyses in the main text evaluate explanations through replanning, path efficiency, and success, these reward values primarily serve the online execution environment rather than the reported utility metric itself.

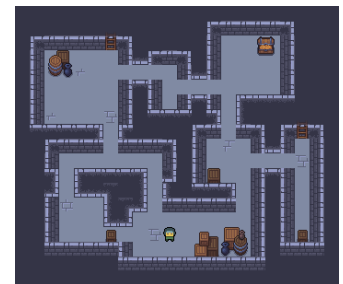


Figure E.2: Example navigation map. A complete dungeon-style grid world as used in the explanation experiments. Dark cells are walls (tile type 1); light cells are traversable floor (0); the agent start (2) and treasure goal (3) are marked.

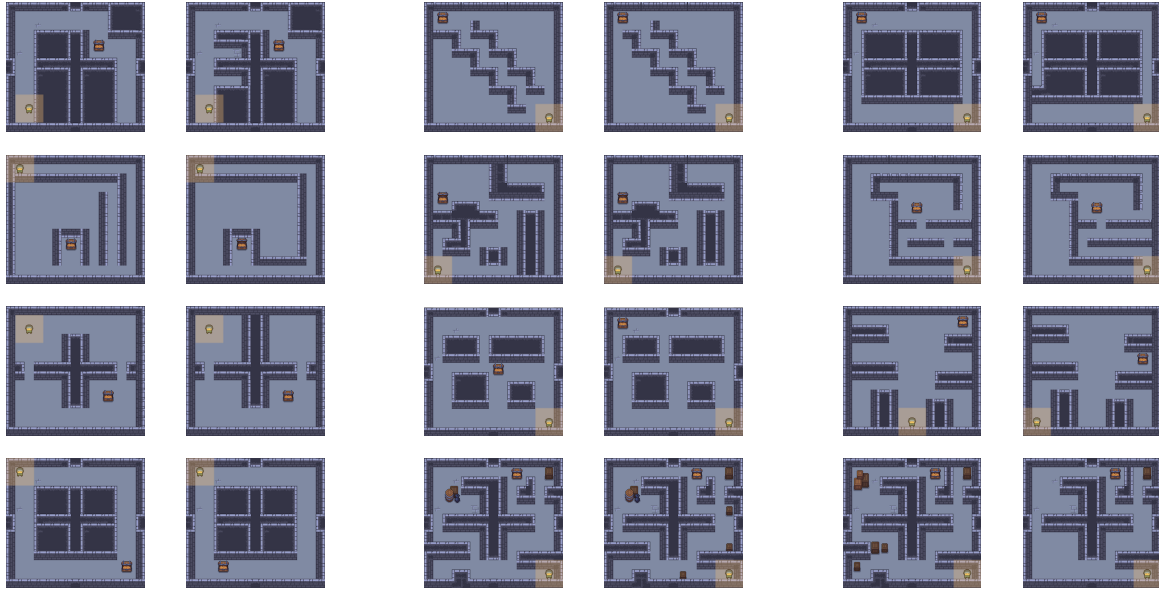


Figure E.1: All 24 navigation maps arranged into 12 near-matched pairs. Maps are displayed in a 4×6 grid; adjacent column pairs (columns 1-2, 3-4, 5-6 within each row) constitute one matched pair. Maps within a pair share the same broad spatial layout but differ by a localized structural edit—such as adding a wall segment, removing a connecting corridor, or repositioning the goal—while keeping the global topology approximately fixed. The four rows group pairs 1-3 (row 1), 4-6 (row 2), 7-9 (row 3), and 10-12 (row 4). Maps range from open floor plans with many alternative routes (e.g., pairs 1 and 10) to corridor-heavy layouts with narrow critical paths and costly dead ends (e.g., pairs 6 and 11). Structural statistics for all 24 maps are reported in Table E.1.

E.1.2 Observation model and memory

Explainers are assumed to have full access to the map layout and goal location. Explainees, by contrast, act under partial observability. At each time step, the listener receives a local square field of view (FoV) centered on the current position. The default setting for the translation-and-navigation evaluation pipeline uses $\text{FoV}=3$.

The agent also maintains two internal state representations over time. First, it stores a *memory grid* containing all previously observed cells, so explored regions accumulate monotonically over the episode. Second, it stores a *belief map* over unexplored cells. This belief map is initialized from the linguistic analysis of the explanation and renormalized after each observation so that already explored non-goal cells receive zero probability mass.

Example of internal representations.

Memory grid (what the agent has seen):

```
# # # # #
# . . . #
# . v ? ?
# . . ? ?
# # # ? ?
```

. = floor # = wall v = agent ? = unknown

Belief map (probability mass over unexplored cells), initialized from the phrase “the treasure is in the bottom-right corner”:

```
0 0 0 0 0
0 0 0 0 0
0 0 - .1 .1
0 0 0 .3 .2
0 0 0 .1 .2
```

Explored cells (‘.’, ‘#’) receive zero mass; remaining mass is proportional to the linguistically inferred prior.

E.1.3 Map structural metrics

To characterize the maps more precisely, we computed graph-based metrics over the reachable state space induced by each map. For each reachable cell, we define its degree as the number of traversable neighbors under four-connected movement. From these local degrees we derive several interpretable summary statistics:

- ▶ *reachable states*: the total number of cells reachable from the start;
- ▶ *shortest-path length*: the geodesic distance from start to treasure;
- ▶ *dead-end fraction*: the proportion of reachable cells with degree 1;
- ▶ *corridor fraction*: the proportion of reachable cells with degree 2;
- ▶ *decision-point fraction*: the proportion of reachable cells with degree ≥ 3 .

Map	Pair	S	d^*	DE%	Cor%	Dec%	Brit
1	1	66	18	0.0	12.1	87.9	0.12
2	1	72	18	2.8	16.7	80.6	0.19
3	2	123	21	0.0	26.0	74.0	0.26
4	2	124	21	0.0	25.0	75.0	0.25
5	3	68	21	0.0	29.4	70.6	0.29
6	3	66	21	1.5	27.3	71.2	0.29
7	4	117	31	0.9	38.5	60.7	0.39
8	4	121	32	0.8	30.6	68.6	0.31
9	5	88	21	2.3	36.4	61.4	0.39
10	5	91	21	1.1	38.5	60.4	0.40
11	6	112	11	1.8	42.9	55.4	0.45
12	6	111	15	1.8	44.1	54.1	0.46
13	7	116	18	0.0	13.8	86.2	0.14
14	7	110	18	0.0	16.4	83.6	0.16
15	8	101	10	0.0	15.8	84.2	0.16
16	8	101	21	0.0	15.8	84.2	0.16
17	9	103	15	1.0	21.4	77.7	0.22
18	9	103	17	1.0	21.4	77.7	0.22
19	10	84	21	0.0	4.8	95.2	0.05
20	10	84	12	0.0	4.8	95.2	0.05
21	11	80	20	1.3	58.8	40.0	0.60
22	11	76	20	1.3	68.4	30.3	0.70
23	12	75	20	4.0	54.7	41.3	0.59
24	12	93	20	2.2	31.2	66.7	0.33

Table E.1: Structural statistics for all 24 navigation maps. Rows follow the same reading order as Figure E.1 (left to right, top to bottom); a horizontal rule separates each gallery row of six maps. Pair: matched-pair identifier. |S|: number of reachable states. d^* : shortest-path length (steps). DE%: dead-end fraction (degree-1 cells, %). Cor%: corridor fraction (degree-2 cells, %). Dec%: decision-point fraction (degree ≥ 3 cells, %; equals openness). Brit: brittleness (DE% + Cor%, capturing the low-degree, low-redundancy portion of the graph).

The main text uses two composite measures built from these quantities. *Brittleness* captures the prevalence of narrow, low-redundancy states and is operationalized from the low-degree portion of the reachable graph. *Openness* captures the prevalence of high-choice states and is operationalized from the fraction of reachable cells with degree ≥ 3 . We also computed path-conditioned versions of these metrics, such as the average degree and number of decision points along the shortest path. These are useful for distinguishing globally open maps from maps whose critical route remains narrow even when the map as a whole contains many branches.

E.2 Experiment 1: LLM compilation protocol

Conceptually, the compiler maps a natural-language explanation into an action-supporting representation with two components: a *policy prior*, which biases what the listener should do locally, and a *value map*, which biases where the listener should search globally. In the implemented simulation pipeline, this compilation is realized through an online prompting loop that repeatedly infers action-relevant guidance from the explanation, the agent’s current memory state, and its prior failures.

The pipeline has two stages. First, the model analyzes the instruction to infer coarse information about goal location and spatial reference. Second, during navigation, the model repeatedly produces a JSON-structured action plan conditioned on the current explored map and the original explanation. The simulation uses DeepSeek-R1 (70B) as the language model backend, with two fixed prompt templates and repeated stochastic sampling. This execution-and-replanning loop is the mechanism through which the policy-prior and value-map information becomes behaviorally effective.

E.2.1 Belief initialization

Before online planning begins, the first prompt analyzes the explanation once to extract three aspects:

- ▶ a coarse hypothesis of goal location (quadrant, edge, or corner), used to initialize the probabilistic belief map over likely treasure regions;
- ▶ a label for the dominant spatial reference type, such as corner, room, or corridor;
- ▶ a flag (`is_egocentric`) indicating whether directional terms should be interpreted relative to the listener's facing direction or as absolute map directions.

The belief map provides the global, value-like component of the guidance representation: it upweights unexplored cells consistent with the inferred location hypothesis and is renormalized as exploration proceeds. The egocentric flag shapes every subsequent planning prompt, which treats terms like “left” and “right” differently depending on the inferred frame of reference.

The full text of this prompt is shown below.

Procedure E.1: Prompt template `translation_analyze_instruction.txt`. Placeholders `{{GRID_SHAPE_0}}`, `{{GRID_SHAPE_1}}`, and `{{INSTRUCTION}}` are filled at runtime. The model must return a JSON object with fields `goal_location`, `spatial_reference`, and `is_egocentric`.

You are an expert in natural language understanding. Your task is to analyze a treasure-hunt instruction and extract information about the goal's potential location on a `{{GRID_SHAPE_0}}`×`{{GRID_SHAPE_1}}` grid.

Respond in JSON format with these keys:

1. `"goal_location"`: Location hint if present
2. `"spatial_reference"`: Type of spatial description used (e.g., "corner", "edge", "room", "corridor")
3. `"is_egocentric"`: true if instruction uses egocentric references (left/right/forward from agent's perspective), false if allocentric (absolute directions)

For `"goal_location"`, use one of these schemas:

- `{"type": "quadrant", "value": "top-left"|"top-right"|"bottom-left"|"bottom-right"}`
- `{"type": "edge", "value": "top"|"bottom"|"left"|"right"}`
- `{"type": "corner", "value": "top-left"|"top-right"|"bottom-left"|"bottom-right"}`
- If no location information: null

Common spatial terms:

- "corner/room" -> corner of the room (quadrant edge) or open enclosed area
- "corridor/hallway" -> narrow passage
- "end of" -> edge/boundary
- "side" -> edge

Example 1:

Instruction: "The chest is in the bottom-right corner of the room."
`{"goal_location": {"type": "corner", "value": "bottom-right"}, "spatial_reference": "corner", "is_egocentric": null}`

Example 2:

Instruction: "Turn left and go to your right until the end of the corridor."
`{"goal_location": null, "spatial_reference": "corridor", "is_egocentric": true}`

Example 3:

Instruction: "Go up and then go to the right and you will find the room"
`{"goal_location": {"type": "quadrant", "value": "top-right"}, "spatial_reference": "room", "is_egocentric": false}`

Now, analyze the following instruction:

Instruction: "{{INSTRUCTION}}"

E.2.2 Prompted action primitives

The second prompt operates online during navigation. At each replanning step, it receives the agent's current memory grid, the original explanation, and a log of prior successful and failed plans, and asks the model to return a JSON list of structured action commands. Constraining the output to a fixed schema—rather than free text—serves two purposes: it ensures that every generated plan is directly executable without further parsing, and it forces the model to commit to a specific primitive type and parameterization rather than producing vague directional prose.

The implemented primitives are summarised in Table E.2; together they form the policy-like component of the compiler, translating the explanation into a structured sequence of local actions while the belief map encodes which unexplored regions are globally worth targeting.

Table E.2: Action primitives available to the LLM planner.

Primitive	Behaviour	Rationale
turn_left , turn_right	Rotate 90° in-place without moving; updates the agent's facing direction.	Human explanations routinely use egocentric language ("turn left at the junction"). Without explicit reorientation, the agent cannot act on instructions that use relative directions.
move	Move in a direction by an exact step count, a fuzzy quantifier (a_few, several, ...), or a proportional fraction (0–1) of the available path.	Route descriptions vary widely in precision: "go 3 steps", "go a few steps", and "go halfway" all express the same intent at different levels of specificity. Fuzzy and proportional modes ground vague expressions without requiring the model to resolve exact counts.
move_until	Keep moving in a direction until a stopping condition: wall, unknown area, or a spatial landmark (turning, intersection, opening) with an optional occurrence count.	People describe routes by landmarks rather than step counts ("go right until you hit a wall", "continue to the second intersection"). This primitive directly encodes those stopping conditions rather than requiring the model to estimate equivalent distances.
explore , explore_until	Directed search toward unexplored cells or high-belief regions, with the same distance and stopping-condition parameters as move and move_until.	When the explanation provides insufficient local guidance, the agent would otherwise stall. These primitives keep navigation active by driving toward belief-map hotspots until a landmark or newly visible cell disambiguates the next planned move.

The full text of this prompt is shown below.

```
You are a treasure-hunting robot navigating a 2D dungeon to find
treasure. You can only see areas you've explored.

=== CURRENT SITUATION ===
{{MAP_TEXT}}

HUMAN INSTRUCTION: "{{INSTRUCTION}}"
{{INSTRUCTION_CONTEXT}}{{SPATIAL_CONTEXT}}{{HISTORY_CONTEXT}}{{
  REPLAN_SECTION}}

=== NAVIGATION RULES ===
- Each move (up/down/left/right) changes position by exactly ONE cell
- You CANNOT move through # (walls) or outside the grid
- You CAN move through . (explored floor) and move towards ? or *
  (unknown areas)

=== YOUR MISSION ===
Generate a multi-step plan (one or multiple commands) to find the
treasure. Prioritize:
1. Follow the human instruction adaptively based on your CURRENT
   position and what you've already done
2. Explore unknown areas (?) or high-probability areas (*) when the
```

Procedure E.2: Prompt template `translate_plan_next_action.txt`. Placeholders `{{MAP_TEXT}}`, `{{INSTRUCTION}}`, and context blocks are filled at runtime from the agent's current memory state and execution history. The model must return a JSON object with a `plan` field containing a list of structured action commands.


```

instruction is ambiguous or you are stuck
3. Create useful and sufficient plans to minimize replanning

=== AVAILABLE PRIMITIVES ===
1. TURNING (for egocentric instructions):
  {"action": "turn_left"} or {"action": "turn_right"}
  - Rotate facing direction 90 degrees without moving
  - ONLY USE for egocentric instructions

2. EXACT MOVEMENT:
  {"action": "move", "direction": "up|down|left|right", "steps": N}
  - Move N steps in the specified absolute direction

3. FUZZY MOVEMENT:
  {"action": "move", "direction": "...", "steps": "a_little|a_few|
  several|some|many"}
  - Vague quantifiers are automatically grounded to concrete steps

4. PROPORTIONAL MOVEMENT:
  {"action": "move", "direction": "...", "steps": 0.0-1.0}
  - Move a fractional distance along the available path
  - 0.5 = halfway, 0.25 = quarter way, 0.75 = three-quarters

5. CONDITIONAL MOVEMENT:
  {"action": "move_until", "direction": "...", "condition": "wall|
  unknown"}
  - Keep moving until hitting a wall (#) or reaching unexplored area
  (?)

6. LANDMARK MOVEMENT:
  {"action": "move_until", "direction": "...",
  "condition": "turning|intersection|opening", "counting": N}
  - Move until the Nth occurrence of a spatial landmark
  - turning: junction with 2 perpendicular directions available
  - intersection: junction with 3+ directions available
  - opening: union of turning and intersection

7. EXPLORATION:
  {"action": "explore", "direction": "...", "steps": N|"fuzzy
  "|0.0-1.0}
  {"action": "explore_until", "direction": "...", "condition": "wall|
  unknown"}
  {"action": "explore_until", "direction": "...",
  "condition": "turning|intersection|opening", "counting": N}
  - Use when the instruction is unclear or ambiguous and you are stuck
  ,
  especially when spatial features (room, corridor, corner) are
  mentioned or * markers are visible

=== EXAMPLES ===
Allocentric: "Go right then down"
{"plan": [{"action": "move_until", "direction": "right", "condition": "
  wall"},
  {"action": "explore", "direction": "down", "steps": 3}]}

Egocentric: "Turn around and go forward 5 steps"
{"plan": [{"action": "turn_left"},
  {"action": "turn_left"},
  {"action": "move", "direction": "forward", "steps": 5}]}

Vague quantifier: "Go right a few steps"
{"plan": [{"action": "move", "direction": "right", "steps": "a_few"}]}

Proportional: "Go halfway down"
{"plan": [{"action": "move", "direction": "down", "steps": 0.5}]}

Ordinal landmark: "Turn right at the second turning"
{"plan": [{"action": "move_until", "direction": "left",
  "condition": "turning", "counting": 2},
  {"action": "turn_right"}]}

=== OUTPUT FORMAT ===

```

```
Respond ONLY with valid JSON:
{"plan": [{"action": ...}, ...]}

Generate your plan now:
```

Memory, history, and replanning context

Each planning prompt conditions on a text rendering of the agent's current *memory grid* (explained in [Section E.1](#)). This rendering marks explored floor, walls, unknown cells, the current agent position and orientation, and high-probability unexplored cells identified by the belief map. The prompt also includes:

- ▶ the original explanation;
- ▶ the inferred instruction type (egocentric or allocentric);
- ▶ the inferred spatial reference, when available;
- ▶ recent successful plans;
- ▶ failed plans and repeated-failure warnings.

This design serves two purposes. First, it gives the model the current local context needed to interpret the explanation adaptively rather than statically. Second, it makes replanning explicit: when prior plans fail, that failure becomes part of the next prompt.

Execution and failure handling

Generated plans are executed command by command in the dungeon environment. A plan is treated as failed if its first required movement is impossible, if an invalid direction is proposed, or if execution repeatedly fails to make progress. Successful and failed plan fragments are logged separately, allowing subsequent prompts to avoid repeating the same ineffective proposals.

From the perspective of the computational framework, these failures operationalize the replanning cost of an explanation. An explanation that yields stable, executable plans produces low replanning cost; an explanation that repeatedly induces invalid or unhelpful plans produces high replanning cost.

Reliability and interpretation

The implementation is stochastic in two senses. First, the language model itself is sampled stochastically. Second, some primitives, such as fuzzy or proportional movement commands, are grounded into concrete step counts with additional variability. For this reason, explanation quality is estimated over repeated compilations and repeated executions rather than from a single deterministic rollout.

The key interpretive point is that the compiler should not be understood as extracting a literal symbolic program from the text once and for all. Rather, it maps explanations into a structured, action-guiding interface whose policy-like and value-like consequences can be observed over repeated grounded executions.

E.3 Experiment 1: Participant details and pre-registration

All four studies were preregistered on OSF using the standard pre-data-collection workflow. Across studies, participants were recruited on Prolific, paid approximately \$11 per hour in the preregistration plan.

OSF preregistrationsExp. 1: osf.io/udgc3Exp. 2: osf.io/pckumExp. 3: osf.io/pmzw8Exp. 4: osf.io/jhp5z

The preregistrations also specified tutorial examples and comprehension checks before the main task blocks.

Preregistration 1: explanation collection

The goal of Preregistration 1 was to build a corpus of free-text explanations. Participants saw the full map and wrote one explanation for each of 24 maps after completing practice and comprehension checks. We targeted at and finally recruited $n = 50$ participants via Prolific (mean age 29.4 ± 7.1 years). All participants completed a brief practice phase before the main task, and mean completion time was 25.64 ± 11.70 minutes.

Preregistration 2: helpfulness judgments

The Preregistration 2 treated explanation quality as a three-level within-map factor (Good, Medium, Bad) and predicted a monotonic judgment pattern:

$$\text{Good} > \text{Medium} > \text{Bad}.$$

Participants viewed each map and rated the helpfulness of three model-selected explanations on a continuous 0 – 100 slider. The confirmatory analysis plan specified a linear mixed-effects model of the form

$$\text{Rating} \sim \text{ExplanationQuality} + (1 \mid \text{Participant}) + (1 \mid \text{Map}).$$

We targeted at and finally recruited 50 participants (mean age 31.2 ± 8.3 years), none of whom had participated in Preregistration 1. Mean completion time was 17.43 ± 6.82 minutes.

Preregistration 3: navigation without explanations

Preregistration 3 was set as a no-explanation baseline navigation study. Participants navigated all 24 maps under partial observability without any message from an explainer. The preregistration stated two main goals: to establish that the task was non-trivial, and to quantify reliable between-map variability in path length. The planned confirmatory analysis was a mixed-effects model over log path length with random intercepts for participant and map.

We targeted at and finally recruited 50 participants (mean age 28.7 ± 6.9 years). Six participants who failed to complete all maps were excluded and replaced. Mean completion time was 10.89 ± 4.80 minutes.

Preregistration 4: navigation with explanations

Preregistration 4 was set to test whether explanation quality affects navigation performance under partial observability. The preregistration specified two confirmatory hypotheses:

- within the explanation condition, path length should follow $\text{Good} < \text{Medium} < \text{Bad}$;
- Good explanations should improve navigation relative to the no-explanation baseline from Preregistration 3.

We targeted at and finally recruited $n = 150$ participants (mean age 30.1 ± 7.5 years). We created a balanced exposure scheme in which each participant saw 24 maps with 8 Good, 8 Medium, and 8 Bad explanations. The confirmatory analysis plan specified both a within-experiment quality model and a joint model comparing Exp. 3 baseline trials against Exp. 4 explanation conditions.

E.4 Utility-guided generation details

E.4.1 Utility-guided generation details

This appendix describes the planned utility-guided generation pipeline for Experiment 2.

Zero-shot generation prompt

For each of the 24 maps, the generator receives the full map, the start location, the goal location, and the listener's field-of-view constraint. The generator is instructed to write one message for a partner who can only observe the local area around their current position. The prompt is:

```
You are helping a partner navigate a dungeon.
You can see the full map, but your partner cannot.
Your partner starts at {{START_LOCATION}} and can only see cells within
{{FIELD_OF_VIEW}} of their current position.
The treasure is at {{GOAL_LOCATION}}.

Map legend:
- S = partner's start location
- T = treasure
- . = open floor
- # = wall
- ? = unknown to the partner at the beginning

Full map:
{{MAP_TEXT}}

Write one message that will help your partner find the treasure.

Requirements:
- Do not reveal the map as a grid or coordinate list.
- Do not assume the partner can see the full map.
- Use local directions, landmarks, corridors, rooms, turns, or warnings
  when they help the partner act under partial observability.
- Include goal-region information when it would help the partner
  recover
  from a wrong turn.
- Avoid unnecessary detail. The message should be useful during
  navigation,
  not a complete map description.
- Return only the message, with no preamble or explanation of your
  reasoning.
```

Procedure E.3: Prompt template `generate_explanation_zero-shot.txt`. Placeholders `{{MAP_TEXT}}`, `{{START_LOCATION}}`, `{{GOAL_LOCATION}}`, and `{{FIELD_OF_VIEW}}` are filled at runtime. The model returns one natural-language explanation for a partially informed listener.

We sample $M = 50$ zero-shot explanations per map using temperature .7 and $\text{top-}p = .9$. The same model is used for all maps. No utility feedback is provided at this stage.

Utility-guided refinement prompt

Each candidate explanation is compiled into executable guidance and evaluated by the simulated listener. The refinement prompt receives the original explanation, its utility score, and a verbal diagnosis based on the utility components. The feedback is deliberately local: it does not reveal the optimal explanation, only the failure mode detected by the listener model.

```
You are revising a navigation message for a partially informed partner.
The partner can only see nearby cells while moving.
The message will be evaluated by a simulated listener that tries to use
it
```

Procedure E.4: Prompt template `generate_explanation_refine.txt`. Placeholders `{{MAP_TEXT}}`, `{{CANDIDATE}}`, `{{UTILITY}}`, `{{SUCCESS}}`, `{{PATH_LENGTH}}`, `{{REPLAN_COST}}`, and `{{FEEDBACK}}` are filled at runtime. The model returns a revised natural-language explanation.

Table E.3: Simulation summary for utility-guided generation.

Source	Utility	Success	Path length	Replans
Human corpus	.46 ± .21	.58 ± .27	28.9 ± 9.8	4.1 ± 2.0
Human top 10%	.71 ± .09	.80 ± .14	23.6 ± 5.9	2.2 ± 1.1
LLM zero-shot	.57 ± .16	.69 ± .20	26.4 ± 7.1	3.4 ± 1.6
LLM utility-guided	.74 ± .11	.82 ± .13	22.8 ± 5.3	1.9 ± .9

to reach the treasure.

Full map:
{{MAP_TEXT}}

Current message:
{{CANDIDATE}}

Simulation score:
- utility: {{UTILITY}} out of 1
- success rate: {{SUCCESS}}
- mean path length on success: {{PATH_LENGTH}}
- mean replanning cost: {{REPLAN_COST}}

Main feedback:
{{FEEDBACK}}

Revise the message.

Requirements:
- Preserve any useful guidance from the current message.
- Fix the failure described in the feedback.
- Add local decision-point guidance if the listener gets lost or replans.
- Add goal-region or landmark information if it helps recovery.
- Remove detours or details that do not improve navigation.
- Do not reveal the map as a grid or coordinate list.
- Return only the revised message, with no preamble or explanation of your reasoning.

The failure type is selected from the lowest-performing component. High replanning cost produces feedback about ambiguous landmarks or missing decision points. High path inefficiency produces feedback about unnecessary detours. Low success produces feedback about missing route constraints, dead-end warnings, or insufficient goal information.

Selection and stopping rule

The search procedure uses a small beam over natural-language candidates. At each round, all candidates are scored by $U(e, w)$. The top $K = 5$ candidates are retained, and each retained candidate is used to sample 10 revised candidates. This restores the pool to $M = 50$ candidates per map. We run three refinement rounds and select the highest-utility candidate as the final utility-guided explanation.

The refinement curve showed most improvement after the first round. Mean best-candidate utility increased from .62 at round 0 to .70 after round 1, .73 after round 2, and .74 after round 3. This suggests that the first round corrects the most visible failures, while later rounds mostly polish route efficiency.

Example refinement trajectory

The following example illustrates the intended form of the analysis. It is a fabricated trajectory for exposition, not an observed model output.

Zero-shot candidate. “You should start by going north from your starting position and continue along the corridor until you reach the first possible place where the path opens up. At that point, look carefully for a leftward passage, because the treasure is not in the lower or central part of the map. Do not spend time exploring the right-hand side unless you are blocked, because most of those paths lead away from the treasure. After entering the left passage, keep following it upward and slightly left until you reach the room in the upper-left part of the dungeon. The treasure should be somewhere in that upper-left room, so once you get there, explore the nearby cells until you find it.”

Model diagnosis. The explanation succeeds on many rollouts but produces inefficient paths. It gives several redundant global descriptions and asks the listener to “look carefully” or “explore nearby cells,” which the compiler turns into broad search behavior. The main feedback is high path length rather than low success.

Utility-guided revision. “Go north to the first junction, then take the left passage. Follow that corridor until it opens into the upper-left room. Do not explore the right branch; it leads away from the treasure.”

The revised explanation keeps the useful policy and value information but removes redundant global description. It gives the critical local decision, the goal region, and one warning about the misleading branch. This is the pattern the final analysis should quantify across maps.

Bibliography

- Abel, D., M. Bowling, A. Barreto, W. Dabney, S. Dong, S. Hansen, A. Harutyunyan, K. Khetarpal, C. Lyle, R. Pascanu, et al. (2025). "Plasticity as the Mirror of Empowerment". *arXiv preprint arXiv:2505.10361*.
- (2026). "Plasticity as the mirror of empowerment". *Advances in Neural Information Processing Systems* 38, pages 164340–164374.
- Abel, D., M. K. Ho, and A. Harutyunyan (2024). "Three dogmas of reinforcement learning". *arXiv preprint arXiv:2407.10583*.
- Ackerman, P. L. (1988). "Determinants of individual differences during skill acquisition: Cognitive abilities and information processing." *Journal of experimental psychology: General* 117.3, page 288.
- Ackerman, T. A. (2014). "Multidimensional item response theory models". *Wiley StatsRef: Statistics Reference Online*.
- Aeronautiques, C., A. Howe, C. Knoblock, I. D. McDermott, A. Ram, M. Veloso, D. Weld, D. W. Sri, A. Barrett, D. Christianson, et al. (1998). "Pddl—the planning domain definition language". *Technical Report, Tech. Rep.*
- Ahn, M., A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, C. Fu, K. Gopalakrishnan, K. Hausman, et al. (2022). "Do as i can, not as i say: Grounding language in robotic affordances". *arXiv preprint arXiv:2204.01691*.
- Albalak, A., Y. Elazar, S. M. Xie, S. Longpre, N. Lambert, X. Wang, N. Muennighoff, B. Hou, L. Pan, H. Jeong, et al. (2024). "A survey on data selection for language models". *arXiv preprint arXiv:2402.16827*.
- Amarel, S. (1968). *On Representations and Modeling in Problem Solving and on Future Directions for Intelligent Systems*. Technical report.
- Amodei, D., C. Olah, J. Steinhardt, P. Christiano, J. Schulman, and D. Mané (2016). "Concrete problems in AI safety". *arXiv preprint arXiv:1606.06565*.
- Anderson, J. R. (1996). "ACT: A simple theory of complex cognition." *American psychologist* 51.4, page 355.
- Anderson, J. R., A. T. Corbett, K. R. Koedinger, and R. Pelletier (1995). "Cognitive tutors: Lessons learned". *The journal of the learning sciences* 4.2, pages 167–207.
- Anderson, P., Q. Wu, D. Teney, J. Bruce, M. Johnson, N. Sünderhauf, I. Reid, S. Gould, and A. Van Den Hengel (2018). "Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments". *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3674–3683.
- Andreas, J., M. Rohrbach, T. Darrell, and D. Klein (2016). "Neural module networks". *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 39–48.
- Archer, E., I. M. Park, L. Buesing, J. Cunningham, and L. Paninski (2015). "Black-box variational inference for state space models". *arXiv preprint arXiv:1511.07367*.
- Arimoto, S. (1972). "An algorithm for computing the capacity of arbitrary discrete memoryless channels". *IEEE Transactions on Information Theory* 18.1, pages 14–20.
- Arjovsky, M., L. Bottou, I. Gulrajani, and D. Lopez-Paz (2019). "Invariant risk minimization". *arXiv preprint arXiv:1907.02893*.
- Åström, K. J. (1995). "Adaptive control". *Mathematical System Theory: The Influence of RE Kalman*. Springer, pages 437–450.
- Attias, H. (1999). "A variational Bayesian framework for graphical models". *Advances in neural information processing systems* 12.
- Aubret, A., L. Matignon, and S. Hassas (2023). "An information-theoretic perspective on intrinsic motivation in reinforcement learning: A survey". *Entropy* 25.2, page 327.
- Auer, P., N. Cesa-Bianchi, and P. Fischer (2002). "Finite-time analysis of the multiarmed bandit problem". *Machine learning* 47.2, pages 235–256.
- Averell, L. and A. Heathcote (2011). "The form of the forgetting curve and the fate of memories". *Journal of mathematical psychology* 55.1, pages 25–35.
- Bacon, P.-L., J. Harb, and D. Precup (2017). "The option-critic architecture". *Proceedings of the AAAI conference on artificial intelligence*. Volume 31. 1.
- Bai, Y., S. Kadavath, S. Kundu, A. Askell, J. Kernion, A. Jones, A. Chen, A. Goldie, A. Mirhoseini, C. McKinnon, et al. (2022). "Constitutional ai: Harmlessness from ai feedback". *arXiv preprint arXiv:2212.08073*.
- Baker, C. L., J. Jara-Ettinger, R. Saxe, and J. B. Tenenbaum (2017). "Rational quantitative attribution of beliefs, desires and percepts in human mentalizing". *Nature Human Behaviour* 1.4, page 0064.
- Baker, C. L., R. Saxe, and J. B. Tenenbaum (2009). "Action understanding as inverse planning". *Cognition* 113.3, pages 329–349.

- Baker, R. S. d., A. T. Corbett, and V. Aleven (2008). "More accurate student modeling through contextual estimation of slip and guess probabilities in bayesian knowledge tracing". *Intelligent Tutoring Systems: 9th International Conference, ITS 2008, Montreal, Canada, June 23-27, 2008 Proceedings* 9. Springer, pages 406–415.
- Barnes, J. (1994). "Posterior analytics".
- Barrett, L. F. (2017). "The theory of constructed emotion: an active inference account of interoception and categorization". *Social cognitive and affective neuroscience* 12.1, pages 1–23.
- Barsalou, L. W. (1999). "Perceptual symbol systems". *Behavioral and brain sciences* 22.4, pages 577–660.
- Bassett, D. S. and M. G. Mattar (2017). "A network neuroscience of human learning: potential to inform quantitative theories of brain and behavior". *Trends in cognitive sciences* 21.4, pages 250–264.
- Bates, C. J. and R. A. Jacobs (2020). "Efficient data compression in perception and perceptual memory." *Psychological Review* 127.5, page 891.
- Bates, C. J., R. A. Lerch, C. R. Sims, and R. A. Jacobs (2019). "Adaptive allocation of human visual working memory capacity during statistical and categorical learning". en. *Journal of Vision* 19.2, page 11.
- Beal, M. J. and Z. Ghahramani (2003). "The variational Bayesian EM algorithm for incomplete data: with application to scoring graphical model structures". *Bayesian statistics* 7, pages 453–464.
- Bengio, Y., A. Courville, and P. Vincent (2013). "Representation learning: A review and new perspectives". *IEEE transactions on pattern analysis and machine intelligence* 35.8, pages 1798–1828.
- Bengio, Y., J. Louradour, R. Collobert, and J. Weston (2009). "Curriculum learning". *Proceedings of the 26th annual international conference on machine learning*, pages 41–48.
- Berger, T. (1971). "Rate-distortion theory: A mathematical basis for data compression". Prentice-Hall.
- Berlyne, D. E. (1950). "Novelty and curiosity as determinants of exploratory behaviour". *British journal of psychology* 41.1, page 68.
- (1960). "Stimulus Selection and Conflict."
- Bhui, R. and S. J. Gershman (2018). "Decision by sampling implements efficient coding of psychoeconomic functions." *Psychological Review* 125.6, page 985.
- Blahut, R. E. et al. (1972). "Computation of channel capacity and rate-distortion functions". *IEEE Trans. Inf. Theory* 18.4, pages 460–473.
- Blei, D. M., A. Kucukelbir, and J. D. McAuliffe (2017). "Variational inference: A review for statisticians". *Journal of the American statistical Association* 112.518, pages 859–877.
- Bommasani, R., D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M. S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill, et al. (2021). "On the opportunities and risks of foundation models". *arXiv preprint arXiv:2108.07258*.
- Bonawitz, E., P. Shafto, H. Gweon, N. D. Goodman, E. Spelke, and L. Schulz (2011). "The double-edged sword of pedagogy: Instruction limits spontaneous exploration and discovery". *Cognition* 120.3, pages 322–330.
- Botvinick, M., S. Ritter, J. X. Wang, Z. Kurth-Nelson, C. Blundell, and D. Hassabis (2019). "Reinforcement learning, fast and slow". *Trends in cognitive sciences* 23.5, pages 408–422.
- Botvinick, M. M. and D. C. Plaut (2006). "Short-term memory for serial order: a recurrent neural network model." *Psychological review* 113.2, page 201.
- Brady, T. F., T. Konkle, and G. A. Alvarez (2009). "Compression in visual working memory: using statistical regularities to form more efficient memory representations." *Journal of Experimental Psychology: General* 138.4, page 487.
- Bramley, N. R., P. Dayan, T. L. Griffiths, and D. A. Lagnado (2017). "Formalizing Neurath's ship: Approximate algorithms for online causal learning." *Psychological review* 124.3, page 301.
- Brändle, F, M Binz, and E Schulz (2022). "Exploration beyond bandits". *The Drive for Knowledge: The Science of Human Information-Seeking*. Cambridge University Press, pages 147–168.
- Breese, J. S., D. Heckerman, and C. Kadie (2013). "Empirical analysis of predictive algorithms for collaborative filtering". *arXiv preprint arXiv:1301.7363*.
- Bricken, T., A. Templeton, J. Batson, B. Chen, A. Jermyn, T. Conerly, N. Turner, C. Anil, C. Denison, A. Askeell, et al. (2023). "Towards monosemanticity: Decomposing language models with dictionary learning". *Transformer Circuits Thread* 2.5, page 6.
- Brown, T., B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askeell, et al. (2020). "Language models are few-shot learners". *Advances in neural information processing systems* 33, pages 1877–1901.
- Bruner, J. S. (1973). "Organization of early skilled action". *Child development*, pages 1–11.
- Burda, Y., H. Edwards, A. Storkey, and O. Klimov (2018). "Exploration by random network distillation". *arXiv preprint arXiv:1810.12894*.

- Burkholder, L. (2021). "Equipoise and ethics in educational research". *Theory and Research in Education* 19.1, pages 65–77.
- Byrne, R. M. (2016). "Counterfactual thought". *Annual review of psychology* 67.1, pages 135–157.
- Cancho, R. Ferrer i and R. V. Solé (2003). "Least effort and the origins of scaling in human language". *Proceedings of the National Academy of Sciences of the United States of America* 100.3, pages 788–791.
- Carey, S. (2000). "The origin of concepts". *Journal of Cognition and Development* 1.1, pages 37–41.
- Celisse, A., G. Marot, M. Pierre-Jean, and G. Rigai (2017). "New efficient algorithms for multiple change-point detection with kernels". *arXiv preprint arXiv:1710.04556*.
- Cepeda, N. J., H. Pashler, E. Vul, J. T. Wixted, and D. Rohrer (2006). "Distributed practice in verbal recall tasks: A review and quantitative synthesis." *Psychological bulletin* 132.3, page 354.
- Chang, H.-S., H.-J. Hsu, and K.-T. Chen (2015). "Modeling exercise relationships in e-learning: A unified approach." *EDM*, pages 532–535.
- Chase, W. G. and H. A. Simon (1973). "Perception in chess". *Cognitive psychology* 4.1, pages 55–81.
- Chater, N., J. B. Tenenbaum, and A. Yuille (2006). "Probabilistic models of cognition: Conceptual foundations". *Trends in cognitive sciences* 10.7, pages 287–291.
- Chater, N. and P. Vitányi (2003). "Simplicity: a unifying principle in cognitive science?" *Trends in Cognitive Sciences* 7.1, pages 19–22.
- Chen, J., Z. Liu, S. Huang, Q. Liu, and W. Luo (2023). "Improving interpretability of deep sequential knowledge tracing models with question-centric cognitive representations". *arXiv preprint arXiv:2302.06885*.
- Chen, T., S. Cheyette, K. Allen, J. Tenenbaum, and K. Smith (2026). "Just in Time" World Modeling Supports Human Planning and Reasoning". *arXiv preprint arXiv:2601.14514*.
- Chevalier-Boisvert, M., D. Bahdanau, S. Lahlou, L. Willems, C. Saharia, T. H. Nguyen, and Y. Bengio (2018). "Babyai: A platform to study the sample efficiency of grounded language learning". *arXiv preprint arXiv:1810.08272*.
- Chevalier-Boisvert, M., B. Dai, M. Towers, R. de Lazcano, L. Willems, S. Lahlou, S. Pal, P. S. Castro, and J. Terry (2023). "Minigrid & Miniworld: Modular & Customizable Reinforcement Learning Environments for Goal-Oriented Tasks". *CoRR abs/2306.13831*.
- Chi, M. T., M. Bassok, M. W. Lewis, P. Reimann, and R. Glaser (1989). "Self-explanations: How students study and use examples in learning to solve problems". *Cognitive science* 13.2, pages 145–182.
- Chi, M. T., N. De Leeuw, M.-H. Chiu, and C. LaVancher (1994). "Eliciting self-explanations improves understanding". *Cognitive science* 18.3, pages 439–477.
- Choi, Y., Y. Lee, J. Cho, J. Baek, B. Kim, Y. Cha, D. Shin, C. Bae, and J. Heo (2020). "Towards an appropriate query, key, and value computation for knowledge tracing". *Proceedings of the seventh ACM conference on learning@ scale*, pages 341–344.
- Chomsky, N. (1956). "Three models for the description of language". *IRE Transactions on information theory* 2.3, pages 113–124.
- Christiano, P. F., J. Leike, T. Brown, M. Martic, S. Legg, and D. Amodei (2017). "Deep reinforcement learning from human preferences". *Advances in neural information processing systems* 30.
- Chu, J. and L. E. Schulz (2020). "Play, curiosity, and cognition". *Annual Review of Developmental Psychology* 2.1, pages 317–343.
- Church, A. (1932). "A set of postulates for the foundation of logic". *Annals of mathematics* 33.2, pages 346–366.
- (1936). "An unsolvable problem of elementary number theory". *American Journal of Mathematics* 58.2, e1008367.
- Clark, A. (2013). "Whatever next? Predictive brains, situated agents, and the future of cognitive science". *Behavioral and brain sciences* 36.3, pages 181–204.
- Clark, A. and D. Chalmers (1998). "The extended mind". *analysis* 58.1, pages 7–19.
- Clark, H. H. and S. E. Brennan (1991). "Grounding in communication."
- Clark, H. H. and D. Wilkes-Gibbs (1986). "Referring as a collaborative process". *Cognition* 22.1, pages 1–39.
- Colas, C., T. Karch, O. Sigaud, and P.-Y. Oudeyer (2022). "Autotelic agents with intrinsically motivated goal-conditioned reinforcement learning: a short survey". *Journal of Artificial Intelligence Research* 74, pages 1159–1199.
- Conway, C. M., A. Bauernschmidt, S. S. Huang, and D. B. Pisoni (2010). "Implicit statistical learning in language processing: Word predictability is the key". *Cognition* 114.3, pages 356–371.
- Cook, C., N. D. Goodman, and L. E. Schulz (2011). "Where science starts: Spontaneous experiments in preschoolers' exploratory play". *Cognition* 120.3, pages 341–349.
- Corbett, A. T. and J. R. Anderson (1994a). "Knowledge tracing: Modeling the acquisition of procedural knowledge". *User modeling and user-adapted interaction* 4, pages 253–278.

- Corbett, A. T. and J. R. Anderson (1994b). "Knowledge tracing: Modeling the acquisition of procedural knowledge". *User modeling and user-adapted interaction* 4, pages 253–278.
- Correa, C. G., T. L. Griffiths, and N. D. Daw (2024). "Program-based strategy induction for reinforcement learning". *arXiv preprint arXiv:2402.16668*.
- Correa, C. G., S. Sanborn, M. K. Ho, F. Callaway, N. D. Daw, and T. L. Griffiths (2023). "Exploring the hierarchical structure of human plans via program generation". *arXiv preprint arXiv:2311.18644*.
- Cover, T. M. (1999). "Elements of information theory". *John Wiley & Sons*.
- Culbertson, J., P. Smolensky, and G. Legendre (2012). "Learning biases predict a word order universal". *Cognition* 122.3, pages 306–329.
- Dai*, F., H. Zhou*, and A. Gopnik (2025). "Hierarchical World Models with Causal Curation for Generalizing Agents". *NeurIPS 2025 Workshop on Bridging Language, Agent, and World Models for Reasoning and Planning*.
- Dasgupta, I., E. Schulz, N. D. Goodman, and S. J. Gershman (2018). "Remembrance of inferences past: Amortization in human hypothesis generation". *Cognition* 178, pages 67–81.
- Daunizeau, J., H. E. Den Ouden, M. Pessiglione, S. J. Kiebel, K. E. Stephan, and K. J. Friston (2010). "Observing the observer (I): meta-bayesian models of learning and decision-making". *PloS one* 5.12, e15554.
- Davidson, G., T. M. Gureckis, and B. Lake (2022). "Creativity, compositionality, and common sense in human goal generation". *Proceedings of the annual meeting of the cognitive science society*. Volume 44. 44.
- Daw, N. D. (2011). "Trial-by-trial data analysis using computational models". *Decision Making, Affect, and Learning: Attention and Performance XXIII*. Edited by M. R. Delgado, E. A. Phelps, and T. W. Robbins. *Oxford University Press*, pages 3–38.
- Daw, N. D., S. J. Gershman, B. Seymour, P. Dayan, and R. J. Dolan (2011). "Model-based influences on humans' choices and striatal prediction errors". *Neuron* 69.6, pages 1204–1215.
- Daw, N. D., Y. Niv, and P. Dayan (2005). "Uncertainty-based competition between prefrontal and dorsolateral striatal systems for behavioral control". *Nature neuroscience* 8.12, pages 1704–1711.
- Dayan, P. and N. D. Daw (2008). "Decision theory, reinforcement learning, and the brain". *Cognitive, Affective, & Behavioral Neuroscience* 8.4, pages 429–453.
- Dayan, P. and G. E. Hinton (1992). "Feudal reinforcement learning". *Advances in neural information processing systems* 5.
- Deci, E. L. and R. M. Ryan (1985). "Conceptualizations of intrinsic motivation and self-determination". *Intrinsic motivation and self-determination in human behavior*. *Springer*, pages 11–40.
- Degen, J. (2023). "The Rational Speech Act Framework". *Annual Review of Linguistics* 9, pages 519–540.
- Dehaene, S., F. Al Roumi, Y. Lakretz, S. Planton, and M. Sablé-Meyer (2022). "Symbols and mental programs: a hypothesis about human singularity". *Trends in Cognitive Sciences* 26.9, pages 751–766.
- DeJong, G. and R. Mooney (1986). "Explanation-based learning: An alternative view". *Machine learning* 1.2, pages 145–176.
- Dekker, R. B., F. Otto, and C. Summerfield (2022). "Curriculum learning for human compositional generalization". *Proceedings of the National Academy of Sciences* 119.41, e2205582119.
- Dempster, A. P., N. M. Laird, and D. B. Rubin (1977). "Maximum likelihood from incomplete data via the EM algorithm". *Journal of the Royal Statistical Society. Series B (methodological)*, pages 1–38.
- Denis, M. (1997). "The description of routes: A cognitive approach to the production of spatial discourse". *Current psychology of cognition* 16, pages 409–458.
- Devlin, J., J. Uesato, S. Bhupatiraju, R. Singh, A.-r. Mohamed, and P. Kohli (2017). "Robustfill: Neural program learning under noisy i/o". *International conference on machine learning*. PMLR, pages 990–998.
- Dewey, J. (1910). "The child and the curriculum". *University of Chicago Press Chicago*.
- (1938). "Logic". *Holt New York*.
- Dickinson, A. (1985). "Actions and habits: the development of behavioural autonomy". *Philosophical Transactions of the Royal Society of London. B, Biological Sciences* 308.1135, pages 67–78.
- Dilokthanakul, N., P. A. Mediano, M. Garnelo, M. C. Lee, H. Salimbeni, K. Arulkumaran, and M. Shanahan (2016). "Deep unsupervised clustering with gaussian mixture variational autoencoders". *arXiv preprint arXiv:1611.02648*.
- Doucet, A., N. De Freitas, and N. Gordon (2001). "An introduction to sequential Monte Carlo methods". *Sequential Monte Carlo methods in practice*. *Springer*, pages 3–14.
- Dubey, R. and T. L. Griffiths (2020). "Reconciling novelty and complexity through a rational analysis of curiosity." *Psychological Review* 127.3, page 455.
- Dulac-Arnold, G., N. Levine, D. J. Mankowitz, J. Li, C. Paduraru, S. Gowal, and T. Hester (2021). "Challenges of real-world reinforcement learning: definitions, benchmarks and analysis". *Machine Learning* 110.9, pages 2419–2468.

- Dupuis, F., W. Yu, and F. M. Willems (2004). "Blahut-Arimoto algorithms for computing channel capacity and rate-distortion with side information". *International Symposium on Information Theory, 2004. ISIT 2004. Proceedings*. IEEE, page 179.
- Ebbinghaus, H. (1885). "Über das Gedächtnis: Untersuchungen zur experimentellen Psychologie". Leipzig: *Duncker & Humblot*.
- Ellis, K., A. Albright, A. Solar-Lezama, J. B. Tenenbaum, and T. J. O'Donnell (2022). "Synthesizing theories of human language with Bayesian program induction". *Nature communications* 13.1, page 5024.
- Ellis, K., C. Wong, M. Nye, M. Sablé-Meyer, L. Morales, L. Hewitt, L. Cary, A. Solar-Lezama, and J. B. Tenenbaum (2021). "Dreamcoder: Bootstrapping inductive program synthesis with wake-sleep library learning". *Proceedings of the 42nd acm sigplan international conference on programming language design and implementation*, pages 835–850.
- Elman, J. L. (1993). "Learning and development in neural networks: The importance of starting small". *Cognition* 48.1, pages 71–99.
- Epstude, K. and N. J. Roesse (2008). "The functional theory of counterfactual thinking". *Personality and social psychology review* 12.2, pages 168–192.
- Everitt, T., M. Hutter, R. Kumar, and V. Krakovna (2021). "Reward tampering problems and solutions in reinforcement learning: A causal influence diagram perspective". *Synthese* 198.Suppl 27, pages 6435–6467.
- Fan, L., G. Wang, Y. Jiang, A. Mandlekar, Y. Yang, H. Zhu, A. Tang, D.-A. Huang, Y. Zhu, and A. Anandkumar (2022). "Minedojo: Building open-ended embodied agents with internet-scale knowledge". *Advances in Neural Information Processing Systems* 35, pages 18343–18362.
- Feldman, J. (2000). "Minimization of Boolean complexity in human concept learning". *Nature* 407.6804, pages 630–633.
- Ferns, N., P. Panangaden, and D. Precup (2004). "Metrics for Finite Markov Decision Processes." *Uai*. Volume 4, pages 162–169.
- Ferrao, J. L. and R. F. Cunha (2025). "World Model Agents with Change-Based Intrinsic Motivation". *Northern Lights Deep Learning Conference*. PMLR, pages 66–74.
- Finn, C., P. Abbeel, and S. Levine (2017). "Model-agnostic meta-learning for fast adaptation of deep networks". *International conference on machine learning*. PMLR, pages 1126–1135.
- Fiser, J. and R. N. Aslin (2002). "Statistical learning of higher-order temporal structure from visual shape sequences." *Journal of Experimental Psychology: Learning, Memory, and Cognition* 28.3, page 458.
- Fitch, W. T. and M. D. Martins (2014). "Hierarchical processing in music, language, and action: Lashley revisited". *Annals of the New York Academy of Sciences* 1316.1, pages 87–104.
- Fodor, J. A. (1975). "The Language of Thought". *Harvard University Press*.
- Fodor, J. A. (1983). "The modularity of mind". *MIT press*.
- Fodor, J. A. and Z. W. Pylyshyn (1988). "Connectionism and cognitive architecture: A critical analysis". *Cognition* 28.1-2, pages 3–71.
- Fogarty, L., N. Creanza, and M. W. Feldman (2015). "Cultural evolutionary perspectives on creativity and human innovation". *Trends in ecology & evolution* 30.12, pages 736–754.
- Fonollosa, J., E. Neftci, and M. Rabinovich (2015). "Learning of chunking sequences in cognition and behavior". *PLoS computational biology* 11.11, e1004592.
- Frank, M. C. and N. D. Goodman (2012). "Predicting pragmatic reasoning in language games". *Science* 336.6084, pages 998–998.
- Fränken, J.-P., N. C. Theodoropoulos, and N. R. Bramley (2022). "Algorithms of adaptation in inductive inference". *Cognitive Psychology* 137, page 101506.
- Frazier, B. N., S. A. Gelman, and H. M. Wellman (2009). "Preschoolers' search for explanatory information within adult-child conversation". *Child development* 80.6, pages 1592–1611.
- Freiesleben, T., G. König, C. Molnar, and A. Tejero-Cantero (2022). "Scientific inference with interpretable machine learning: Analyzing models to learn about real-world phenomena". *arXiv preprint arXiv:2206.05487*.
- Fried, D., R. Hu, V. Cirik, A. Rohrbach, J. Andreas, L.-P. Morency, T. Berg-Kirkpatrick, K. Saenko, D. Klein, and T. Darrell (2018). "Speaker-follower models for vision-and-language navigation". *Advances in neural information processing systems* 31.
- Frosch, C. A., T. McCormack, D. A. Lagnado, and P. Burns (2012). "Are causal structure and intervention judgments inextricably linked? A developmental study". *Cognitive Science* 36.2, pages 261–285.
- Gao, L., S. Biderman, S. Black, L. Golding, T. Hoppe, C. Foster, J. Phang, H. He, A. Thite, N. Nabeshima, et al. (2020). "The pile: An 800gb dataset of diverse text for language modeling". *arXiv preprint arXiv:2101.00027*.
- Garcia-Valencia, S., A. Betancourt, and J. G. Lalinde-Pulido (2020). "Sequence Generation using Deep Recurrent Networks and Embeddings: A study case in music". *ArXiv e-prints* abs/2012.0. arXiv: 2012.01231 [cs.SD].

- Gaunt, A. L., M. Brockschmidt, N. Kushman, and D. Tarlow (2017). "Differentiable programs with neural libraries". *International Conference on Machine Learning*. PMLR, pages 1213–1222.
- Geisler, W. S. and R. L. Diehl (2003). "A Bayesian approach to the evolution of perceptual and cognitive systems". *Cognitive Science* 27.3, pages 379–402.
- Gelada, C., S. Kumar, J. Buckman, O. Nachum, and M. G. Bellemare (2019). "Deepmdp: Learning continuous latent space models for representation learning". *International conference on machine learning*. PMLR, pages 2170–2179.
- Gershman, S. J. (2019). "What does the free energy principle tell us about the brain?" *arXiv preprint arXiv:1901.07945*.
- (2021). "The Rational Analysis of Memory". *Oxford Handbook of Human Memory*. Oxford, UK: Oxford University Press.
- Gershman, S. J., E. J. Horvitz, and J. B. Tenenbaum (2015). "Computational rationality: A converging paradigm for intelligence in brains, minds, and machines". *Science* 349.6245, pages 273–278.
- Gervet, T., K. Koedinger, J. Schneider, T. Mitchell, et al. (2020). "When is deep learning the best approach to knowledge tracing?" *Journal of Educational Data Mining* 12.3, pages 31–54.
- Ghahramani, Z. (2013). "Bayesian non-parametrics and the probabilistic approach to modelling". *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 371.1984.
- Ghahramani, Z. and G. E. Hinton (2000). "Variational learning for switching state-space models". *Neural computation* 12.4, pages 831–864.
- Ghosh, A., N. Heffernan, and A. S. Lan (2020). "Context-aware attentive knowledge tracing". *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2330–2339.
- Gibson, J. J. (2014). "The ecological approach to visual perception: classic edition". *Psychology press*.
- Gigerenzer, G. (1991). "From tools to theories: A heuristic of discovery in cognitive psychology." *Psychological review* 98.2, page 254.
- (2004). "Fast and frugal heuristics: The tools of bounded rationality". *Blackwell handbook of judgment and decision making* 62, page 88.
- Giron, A. P., S. Ciranka, E. Schulz, W. van den Bos, A. Ruggeri, B. Meder, and C. M. Wu (2023). "Developmental changes in exploration resemble stochastic optimization". *Nature Human Behaviour*.
- Gläscher, J., N. Daw, P. Dayan, and J. P. O'Doherty (2010). "States versus rewards: dissociable neural prediction error signals underlying model-based and model-free reinforcement learning". *Neuron* 66.4, pages 585–595.
- Glymour, C., K. Zhang, and P. Spirtes (2019). "Review of causal discovery methods based on graphical models". *Frontiers in genetics* 10, page 524.
- Gmytrasiewicz, P. J. and P. Doshi (2005). "A framework for sequential planning in multi-agent settings". *Journal of Artificial Intelligence Research* 24, pages 49–79.
- Gold, E. M. (1967). "Language identification in the limit". *Information and control* 10.5, pages 447–474.
- Golomb, S. (1966). "Run-length encodings (corresp.)" *IEEE transactions on information theory* 12.3, pages 399–401.
- Goodfellow, I. J., Y. Bengio, and A. Courville (2016). "Deep Learning".
- Goodman, N. D. (2013). "The principles and practice of probabilistic programming". *ACM SIGPLAN Notices* 48.1, pages 399–402.
- Goodman, N. D. and M. C. Frank (2016). "Pragmatic language interpretation as probabilistic inference". *Trends in cognitive sciences* 20.11, pages 818–829.
- Goodman, N. D. and A. Stuhlmüller (2013). "Knowledge and implicature: Modeling language understanding as social cognition". *Topics in cognitive science* 5.1, pages 173–184.
- Goodman, N. D., J. B. Tenenbaum, J. Feldman, and T. L. Griffiths (2008). "A rational analysis of rule-based concept learning". *Cognitive Science* 32.1, pages 108–154.
- Gopnik, A. (2007). "A. Gopnik, & J. Tenenbaum (2007). Bayesian networks, Bayesian learning and cognitive development. *Developmental Science* (special section on Bayesian and Bayes-Net approaches to development) 10 (3): 281-287." *development* 10.3, pages 281–287.
- (2012). "Scientific thinking in young children: Theoretical advances, empirical research, and policy implications". *Science* 337.6102, pages 1623–1627.
- (2020). "Childhood as a solution to explore–exploit tensions". *Philosophical Transactions of the Royal Society B* 375.1803, page 20190502.
- (2023). "Caregiving in philosophy, biology & political economy". *Dædalus* 152.1, pages 58–69.
- (2024). "Empowerment as Causal Learning, Causal Learning as Empowerment: A bridge between Bayesian causal hypothesis testing and reinforcement learning."

- Gopnik, A., C. Glymour, D. M. Sobel, L. E. Schulz, T. Kushnir, and D. Danks (2004). "A theory of causal learning in children: causal maps and Bayes nets." *Psychological review* 111.1, page 3.
- Gopnik, A. and A. N. Meltzoff (1997). "Words, thoughts, and theories". *Mit Press*.
- Gopnik, A. and H. M. Wellman (2012). "Reconstructing constructivism: causal models, Bayesian learning mechanisms, and the theory theory." *Psychological bulletin* 138.6, page 1085.
- Gottlieb, J., P.-Y. Oudeyer, M. Lopes, and A. Baranes (2013). "Information-seeking, curiosity, and attention: computational and neural mechanisms". *Trends in cognitive sciences* 17.11, pages 585–593.
- Grice, H. P. (1975). "Logic and conversation". *Speech acts*. Brill, pages 41–58.
- Griffiths, T. L., N. Chater, C. Kemp, A. Perfors, and J. B. Tenenbaum (2010). "Probabilistic models of cognition: Exploring representations and inductive biases". *Trends in cognitive sciences* 14.8, pages 357–364.
- Griffiths, T. L., F. Lieder, and N. D. Goodman (2015). "Rational use of cognitive resources: Levels of analysis between the computational and the algorithmic". *Topics in cognitive science* 7.2, pages 217–229.
- Griffiths, T. L. and J. B. Tenenbaum (2005). "Structure and strength in causal induction". *Cognitive psychology* 51.4, pages 334–384.
- (2009). "Theory-based causal induction." *Psychological review* 116.4, page 661.
- Gruaz, L., A. Modirshanechi, and J. Brea (2024). "Merits of curiosity: a simulation study". *PsyArXiv*.
- Grünwald, P. D. (2007). "The Minimum Description Length Principle". *MIT press*.
- Grünwald, P. D., P. Vitányi, et al. (2008). "Algorithmic information theory".
- Ha, D. and J. Schmidhuber (2018a). "Recurrent world models facilitate policy evolution". *Advances in neural information processing systems* 31.
- (2018b). "World models". *arXiv preprint arXiv:1803.10122*.
- Hafner, D. (2021). "Benchmarking the spectrum of agent capabilities". *arXiv preprint arXiv:2109.06780*.
- Hafner, D., J. Pasukonis, J. Ba, and T. Lillicrap (2023). "Mastering diverse domains through world models". *arXiv preprint arXiv:2301.04104*.
- (2025). "Mastering diverse control tasks through world models". *Nature*, pages 1–7.
- Halahakoon, D. C. and M. Browning (2023). "Pramipexole for the Treatment of Depression: Efficacy and Mechanisms". *Emerging Neurobiology of Antidepressant Treatments*. Springer, pages 49–65.
- Hamilton, W. L., R. Ying, and J. Leskovec (2017). "Representation learning on graphs: Methods and applications". *arXiv preprint arXiv:1709.05584*.
- Harasim, D. (2020). "The learnability of the grammar of jazz: Bayesian inference of hierarchical structures in harmony". PhD thesis. EPFL.
- Hayes, C. C. (1989). "A Model of Planning for Plan Efficiency: Taking Advantage of Operator Overlap." *IJCAI*, pages 949–953.
- Helmert, M. (2006). "The fast downward planning system". *Journal of Artificial Intelligence Research* 26, pages 191–246.
- Hempel, C. G. and P. Oppenheim (1948). "Studies in the Logic of Explanation". *Philosophy of science* 15.2, pages 135–175.
- Henson, R. N. (1998). "Short-term memory for serial order: The start-end model". *Cognitive psychology* 36.2, pages 73–137.
- Higgins, I., L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner (2017). "beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework". *International Conference on Learning Representations*.
- Hill, L. H. (2005). "Concept mapping to encourage meaningful student learning". *Adult Learning* 16.3-4, pages 7–13.
- Hilton, D. J. and B. R. Slugoski (1986). "Knowledge-based causal attribution: The abnormal conditions focus model." *Psychological review* 93.1, page 75.
- Hinton, G. E., P. Dayan, B. J. Frey, and R. M. Neal (1995). "The "wake-sleep" algorithm for unsupervised neural networks". *Science* 268.5214, pages 1158–1161.
- Hinton, G. E. and D. Van Camp (1993). "Keeping the neural networks simple by minimizing the description length of the weights". *Proceedings of the sixth annual conference on Computational learning theory*, pages 5–13.
- Ho, M. K., D. Abel, C. G. Correa, M. L. Littman, J. D. Cohen, and T. L. Griffiths (2022). "People construct simplified mental representations to plan". *Nature* 606.7912, pages 129–136.
- Ho, M. K., M. Littman, J. MacGlashan, F. Cushman, and J. L. Austerweil (2016). "Showing versus doing: Teaching by demonstration". *Advances in neural information processing systems* 29.
- Hochreiter, S. and J. Schmidhuber (1997). "Long short-term memory". *Neural computation* 9.8, pages 1735–1780.
- Houthoofd, R., X. Chen, Y. Duan, J. Schulman, F. De Turck, and P. Abbeel (2016). "Vime: Variational information maximizing exploration". *Advances in neural information processing systems* 29.

- Howes, A., R. L. Lewis, and A. Vera (2009). "Rational adaptation under task and processing constraints: implications for testing theories of cognition and action." *Psychological review* 116.4, page 717.
- Huang, W., P. Abbeel, D. Pathak, and I. Mordatch (2022). "Language models as zero-shot planners: Extracting actionable knowledge for embodied agents". *International conference on machine learning*. PMLR, pages 9118–9147.
- Hume, D. (1739). "A Treatise of Human Nature: Volume 1: Texts".
- Hutchins, E. (1995). "Cognition in the Wild". MIT press.
- Irving, G., P. Christiano, and D. Amodei (2018). "AI safety via debate". *arXiv preprint arXiv:1805.00899*.
- Jacoby, N., R. Polak, J. A. Grahm, D. J. Cameron, K. M. Lee, R. Godoy, E. A. Undurraga, T. Huanca, T. Thalwitzer, N. Doumbia, et al. (2024). "Commonality and variation in mental representations of music revealed by a cross-cultural comparison of rhythm priors in 15 countries". *Nature Human Behaviour*, pages 1–32.
- Jang, E., S. Gu, and B. Poole (2016). "Categorical reparameterization with Gumbel-softmax". *arXiv preprint arXiv:1611.01144*.
- Johnson, J., B. Hariharan, L. Van Der Maaten, J. Hoffman, L. Fei-Fei, C. Lawrence Zitnick, and R. Girshick (2017). "Inferring and executing programs for visual reasoning". *Proceedings of the IEEE international conference on computer vision*, pages 2989–2998.
- Johnson, M., T. Griffiths, and S. Goldwater (2006). "Adaptor grammars: A framework for specifying compositional nonparametric Bayesian models". *Advances in Neural Information Processing Systems* 19.
- Jordan, M. I., Z. Ghahramani, T. S. Jaakkola, and L. K. Saul (1999). "An introduction to variational methods for graphical models". *Machine learning* 37.2, pages 183–233.
- Kaelbling, L. P., M. L. Littman, and A. R. Cassandra (1998). "Planning and acting in partially observable stochastic domains". *Artificial intelligence* 101.1-2, pages 99–134.
- Kalman, R. E. (1960). "A new approach to linear filtering and prediction problems".
- Kant, I. (1908). "Critique of pure reason. 1781". *Modern Classical Philosophers, Cambridge, MA: Houghton Mifflin*, pages 370–456.
- Karmiloff-Smith, B. A. (1994). "Beyond modularity: A developmental perspective on cognitive science". *European journal of disorders of communication* 29.1, pages 95–105.
- Karuza, E. A., S. L. Thompson-Schill, and D. S. Bassett (2016). "Local patterns to global architectures: influences of network topology on human learning". *Trends in cognitive sciences* 20.8, pages 629–640.
- Käser, T., S. Klingler, A. G. Schwing, and M. Gross (2017). "Dynamic Bayesian networks for student modeling". *IEEE Transactions on Learning Technologies* 10.4, pages 450–462.
- Kauvar, I., C. Doyle, L. Zhou, and N. Haber (2023). "Curious replay for model-based adaptation". *arXiv preprint arXiv:2306.15934*.
- Ke, N. R., A. Didolkar, S. Mittal, A. Goyal, G. Lajoie, S. Bauer, D. Rezende, Y. Bengio, M. Mozer, and C. Pal (2021). "Systematic evaluation of causal discovery in visual model based reinforcement learning". *arXiv preprint arXiv:2107.00848*.
- Kearns, M. and S. Singh (2002). "Near-optimal reinforcement learning in polynomial time". *Machine learning* 49.2, pages 209–232.
- Kelemen, D. (2012). "Teleological minds". *Evolution challenges: Integrating research and practice in teaching and learning about evolution* 66.
- Kemp, C. and J. B. Tenenbaum (2008). "The discovery of structural form". *Proceedings of the National Academy of Sciences* 105.31, pages 10687–10692.
- (2009). "Structured statistical models of inductive reasoning." *Psychological review* 116.1, page 20.
- Kidd, C. and B. Y. Hayden (2015). "The psychology and neuroscience of curiosity". *Neuron* 88.3, pages 449–460.
- Kidd, C., S. T. Piantadosi, and R. N. Aslin (2012). "The Goldilocks effect: Human infants allocate attention to visual sequences that are neither too simple nor too complex". *PloS one* 7.5, e36399.
- Kim, H. and A. Mnih (2018). "Disentangling by factorising". *International Conference on Machine Learning*. PMLR, pages 2649–2658.
- Kingma, D. P. and J. Ba (2014). "Adam: A method for stochastic optimization". *arXiv preprint arXiv:1412.6980*.
- Kingma, D. P. and M. Welling (2014). "Auto-Encoding Variational Bayes". *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.
- Kinzler, K. D. and E. S. Spelke (2007). "Core systems in human cognition". *Progress in brain research* 164, pages 257–264.
- Kipf, T. N. and M. Welling (2016). "Semi-supervised classification with graph convolutional networks". *arXiv preprint arXiv:1609.02907*.

- Kirkpatrick, J., R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, et al. (2017). "Overcoming catastrophic forgetting in neural networks". *Proceedings of the national academy of sciences* 114.13, pages 3521–3526.
- Kitcher, P. (1981). "Explanatory unification". *Philosophy of science* 48.4, pages 507–531.
- Klissarov, M., P. D'Oro, S. Sodhani, R. Raileanu, P.-L. Bacon, P. Vincent, A. Zhang, and M. Henaff (2023). "Motif: Intrinsic motivation from artificial intelligence feedback". *arXiv preprint arXiv:2310.00166*.
- Klyubin, A. S., D. Polani, and C. L. Nehaniv (2005). "Empowerment: A universal agent-centric measure of control". *2005 IEEE congress on evolutionary computation*. Volume 1. IEEE, pages 128–135.
- Koedinger, K. R., E. Brunskill, R. S. Baker, E. A. McLaughlin, and J. Stamper (2013). "New potentials for data-driven intelligent tutoring system development and optimization". *Ai Magazine* 34.3, pages 27–41.
- Koh, P. W. and P. Liang (2017). "Understanding black-box predictions via influence functions". *International conference on machine learning*. PMLR, pages 1885–1894.
- Koller, D. and N. Friedman (2009). "Probabilistic graphical models: principles and techniques". MIT press.
- Kool, W., J. T. McGuire, Z. B. Rosen, and M. M. Botvinick (2010). "Decision making and the avoidance of cognitive demand." *Journal of experimental psychology: general* 139.4, page 665.
- Koponen, I. T. and M. Nousiainen (2018). "Concept networks of students' knowledge of relationships between physics concepts: Finding key concepts and their epistemic support". *Applied network science* 3.1, pages 1–21.
- Krakovna, V., J. Uesato, V. Mikulik, M. Rahtz, T. Everitt, R. Kumar, Z. Kenton, J. Leike, and S. Legg (2020). "Specification gaming: the flip side of AI ingenuity". *DeepMind Blog* 3, pages 40–53.
- Kripke, S. A. (1991). "Wittgenstein on rules and private language: An elementary exposition". John Wiley & Sons.
- Kucukelbir, A., D. Tran, R. Ranganath, A. Gelman, and D. M. Blei (2017). "Automatic differentiation variational inference". *Journal of machine learning research* 18.14, pages 1–45.
- L Griffiths, T., C. Kemp, and J. B. Tenenbaum (2008). "Bayesian models of cognition".
- Lai, L. and S. J. Gershman (2021). "Policy compression: An information bottleneck in action selection". *Psychology of Learning and Motivation*. Volume 74. Elsevier, pages 195–232.
- Laird, J., P. Rosenbloom, and A. Newell (2012). "The SOAR architecture". *Universal Subgoalting and Chunking: The Automatic Generation and Learning of Goal Hierarchies*. Springer, pages 17–55.
- Laird, J. E., P. S. Rosenbloom, and A. Newell (1986). "Chunking in Soar: The anatomy of a general learning mechanism". *Machine learning* 1.1, pages 11–46.
- Lake, B. and M. Baroni (2018). "Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks". *International conference on machine learning*. PMLR, pages 2873–2882.
- Lake, B. M. and M. Baroni (2023). "Human-like systematic generalization through a meta-learning neural network". *Nature* 623.7985, pages 115–121.
- Lake, B. M., R. Salakhutdinov, and J. B. Tenenbaum (2015). "Human-level concept learning through probabilistic program induction". *Science* 350.6266, pages 1332–1338.
- Lake, B. M., T. D. Ullman, J. B. Tenenbaum, and S. J. Gershman (2017). "Building machines that learn and think like people". *Behavioral and brain sciences* 40, e253.
- Lan, C. L., S. Tu, A. Oberman, R. Agarwal, and M. G. Bellemare (2022). "On the generalization of representations in reinforcement learning". *arXiv preprint arXiv:2203.00543*.
- Lashley, K. S. et al. (1951). "The problem of serial order in behavior". Volume 21. Bobbs-Merrill Oxford.
- Lassiter, D. and N. D. Goodman (2017). "Adjectival vagueness in a Bayesian model of interpretation". *Synthese* 194.10, pages 3801–3836.
- Lee, K., D. Ippolito, A. Nystrom, C. Zhang, D. Eck, C. Callison-Burch, and N. Carlini (2022). "Deduplicating training data makes language models better". *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8424–8445.
- Lee, S. W., S. Shimojo, and J. P. O'doherty (2014). "Neural computations underlying arbitration between model-based and model-free learning". *Neuron* 81.3, pages 687–699.
- Leike, J., D. Krueger, T. Everitt, M. Martic, V. Maini, and S. Legg (2018). "Scalable agent alignment via reward modeling: a research direction". *arXiv preprint arXiv:1811.07871*.
- Lesort, T., N. Díaz-Rodríguez, J.-F. Goudou, and D. Filliat (2018). "State representation learning for control: An overview". *Neural Networks* 108, pages 379–392.
- Li, L., T. J. Walsh, and M. L. Littman (2006). "Towards a unified theory of state abstraction for MDPs." *AI&M* 1.2, page 3.
- Li, M., P. Vitányi, et al. (2008). "An introduction to Kolmogorov complexity and its applications". Volume 3. Springer.

- Liang, J., W. Huang, F. Xia, P. Xu, K. Hausman, B. Ichter, P. Florence, and A. Zeng (2023). "Code as policies: Language model programs for embodied control". *2023 IEEE International conference on robotics and automation (ICRA)*. IEEE, pages 9493–9500.
- Liang, P., M. I. Jordan, and D. Klein (2010). "Learning programs: A hierarchical Bayesian approach". *Proceedings of the 27th international conference on machine learning (icml-10)*, pages 639–646.
- Lidayan, A., Y. Du, E. Kosoy, M. Rufova, P. Abbeel, and A. Gopnik (2025). "Intrinsically-motivated humans and agents in open-world exploration". *arXiv preprint arXiv:2503.23631*.
- Lieder, F. and T. L. Griffiths (2020). "Resource-rational analysis: Understanding human cognition as the optimal use of limited computational resources". *Behavioral and brain sciences* 43, e1.
- Lindley, D. V. (1956). "On a measure of the information provided by an experiment". *The Annals of Mathematical Statistics* 27.4, pages 986–1005.
- Lindsey, R. V., J. D. Shroyer, H. Pashler, and M. C. Mozer (2014). "Improving students long-term knowledge retention through personalized review". *Psychological science* 25.3, pages 639–647.
- Lippe, P., T. Cohen, and E. Gavves (2021). "Efficient neural causal discovery without acyclicity constraints". *arXiv preprint arXiv:2107.10483*.
- Little, D. Y. and F. T. Sommer (2013). "Learning and exploration in action-perception loops". *Frontiers in neural circuits* 7, page 37.
- Liu, Z., Q. Liu, J. Chen, S. Huang, and W. Luo (2023). "simpleKT: a simple but tough-to-beat baseline for knowledge tracing". *arXiv preprint arXiv:2302.06881*.
- Locke, J. (1690). "An essay concerning human understanding."
- Loewenstein, G. (1994). "The psychology of curiosity: A review and reinterpretation." *Psychological bulletin* 116.1, page 75.
- Lombrozo, T. (2006). "The structure and function of explanations". *Trends in cognitive sciences* 10.10, pages 464–470.
- (2007). "Simplicity and probability in causal explanation". *Cognitive psychology* 55.3, pages 232–257.
- (2009). "Explanation and categorization: How "why?" informs "what?"". *Cognition* 110.2, pages 248–253.
- Long, T., Y. Liu, J. Shen, W. Zhang, and Y. Yu (2021). "Tracing knowledge state with individual cognition and acquisition estimation". *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 173–182.
- Loo, N., S. Swaroop, and R. E. Turner (2020). "Generalized variational continual learning". *arXiv preprint arXiv:2011.12328*.
- Lord, F. M. (2012a). "Applications of item response theory to practical testing problems". *Routledge*.
- (2012b). "Applications of item response theory to practical testing problems". *Routledge*.
- Loui, P., D. L. Wessel, and C. L. H. Kam (2010). "Humans rapidly learn grammatical structure in a new musical scale". *Music Perception* 27.5, pages 377–388.
- Lydon-Staley, D. M., D. Zhou, A. S. Blevins, P. Zurn, and D. S. Bassett (2021). "Hunters, busybodies and the knowledge network building associated with deprivation curiosity". *Nature human behaviour* 5.3, pages 327–336.
- Lynn, C. W. and D. S. Bassett (2020). "How humans learn and represent networks". *Proceedings of the National Academy of Sciences* 117.47, pages 29407–29415.
- MacWhinney, B. (2000). "The CHILDES project: The database". Volume 2. *Psychology Press*.
- Mantiuk*, F., H. Zhou*, and C. M. Wu (2025). "From Curiosity to Competence: How World Models Interact with the Dynamics of Exploration". *Proceedings of the 47th Annual Conference of the Cognitive Science Society*. Edited by A Ruggeri, D Barner, C Walker, and N Bramley. San Francisco, CA: *Cognitive Science Society*.
- Mao, J., C. Gan, P. Kohli, J. B. Tenenbaum, and J. Wu (2019). "The neuro-symbolic concept learner: Interpreting scenes, words, and sentences from natural supervision". *arXiv preprint arXiv:1904.12584*.
- Marcus, G. F. (2003). "The algebraic mind: Integrating connectionism and cognitive science". *MIT press*.
- Marcus, G. F., S. Pinker, M. Ullman, M. Hollander, T. J. Rosen, F. Xu, and H. Clahsen (1992). "Overregularization in language acquisition". *Monographs of the society for research in child development*, pages i–178.
- Marr, D. (2010). "Vision: A computational investigation into the human representation and processing of visual information". *MIT press*.
- Mathy, F. and J. Feldman (2012). "What's magic about magic numbers? Chunking and data compression in short-term memory". *Cognition* 122.3, pages 346–362.
- McClelland, J. L., M. M. Botvinick, D. C. Noelle, D. C. Plaut, T. T. Rogers, M. S. Seidenberg, and L. B. Smith (2010). "Letting structure emerge: connectionist and dynamical systems approaches to cognition". *Trends in cognitive sciences* 14.8, pages 348–356.

- McCloskey, M. (1991). "Networks and theories: The place of connectionism in cognitive science". *Psychological science* 2.6, pages 387–395.
- McCloskey, M. and N. J. Cohen (1989). "Catastrophic interference in connectionist networks: The sequential learning problem". *Psychology of learning and motivation*. Volume 24. Elsevier, pages 109–165.
- Meeter, M. and J. M. Murre (2004). "Consolidation of long-term memory: evidence and alternatives." *Psychological Bulletin* 130.6, page 843.
- Mehr, S. A., M. Singh, D. Knox, D. M. Ketter, D. Pickens-Jones, S. Atwood, C. Lucas, N. Jacoby, A. A. Egner, E. J. Hopkins, et al. (2019). "Universality and diversity in human song". *Science* 366.6468, eaax0868.
- Meltzoff, A. N., A. Waismeyer, and A. Gopnik (2012). "Learning about causes from people: observational causal learning in 24-month-old infants." *Developmental psychology* 48.5, page 1215.
- Messick, S. (1995). "Validity of psychological assessment: Validation of inferences from persons' responses and performances as scientific inquiry into score meaning." *American psychologist* 50.9, page 741.
- Meunier, D., R. Lambiotte, and E. T. Bullmore (2010). "Modular and hierarchically modular organization of brain networks". *Frontiers in neuroscience* 4, page 200.
- Mikolov, T., I. Sutskever, K. Chen, G. S. Corrado, and J. Dean (2013). "Distributed representations of words and phrases and their compositionality". *Advances in neural information processing systems* 26.
- Miller, G. A. (1956). "The magical number seven, plus or minus two: Some limits on our capacity for processing information." *Psychological review* 63.2, page 81.
- Minn, S., J.-J. Vie, K. Takeuchi, H. Kashima, and F. Zhu (2022). "Interpretable knowledge tracing: Simple and efficient student modeling with causal relations". *Proceedings of the AAAI Conference on Artificial Intelligence*. Volume 36. 11, pages 12810–12818.
- Minsky, M. et al. (1974). "A framework for representing knowledge".
- Mislevy, R. J., L. S. Steinberg, and R. G. Almond (2003). "Focus article: On the structure of educational assessments". *Measurement: Interdisciplinary research and perspectives* 1.1, pages 3–62.
- Mitchell, T. M., R. M. Keller, and S. T. Kedar-Cabelli (1986). "Explanation-based generalization: A unifying view". *Machine learning* 1.1, pages 47–80.
- Modirshanechi, A., J. Brea, and W. Gerstner (2022). "A taxonomy of surprise definitions". *Journal of Mathematical Psychology* 110, page 102712.
- Modirshanechi, A., P. Dayan, and E. Schulz (2025). "Testing an Integrative Framework for the Sense of Control". *8th Annual Conference on Cognitive Computational Neuroscience (CCN 2025)*.
- Molenaar, P. C. (2004). "A manifesto on psychology as idiographic science: Bringing the person back into scientific psychology, this time forever". *Measurement* 2.4, pages 201–218.
- Moore, A. W. and C. G. Atkeson (1993). "Prioritized sweeping: Reinforcement learning with less data and less time". *Machine learning* 13.1, pages 103–130.
- Muggleton, S. (1991). "Inductive logic programming". *New generation computing* 8.4, pages 295–318.
- (1994). "Inductive logic programming: derivations, successes and shortcomings". *ACM SIGART Bulletin* 5.1, pages 5–11.
- Nagatani, K., Q. Zhang, M. Sato, Y.-Y. Chen, F. Chen, and T. Ohkuma (2019). "Augmenting knowledge tracing by considering forgetting behavior". *The world wide web conference*, pages 3101–3107.
- Nagy, D. G., G. Orbán, and C. M. Wu (2025a). "Adaptive compression as a unifying framework for episodic and semantic memory". *Nature Reviews Psychology*, pages 1–15.
- Nagy, D. G., T. Shen, H. Zhou, C. M. Wu, and P. Dayan (2025b). "Analogy making as amortised model construction". *arXiv preprint arXiv:2507.16511*.
- Nagy, D. G., B. Török, and G. Orbán (2020). "Optimal forgetting: Semantic compression of episodic memories". *PLoS Computational Biology* 16.10, e1008367.
- Nakagawa, H., Y. Iwasawa, and Y. Matsuo (2019). "Graph-based knowledge tracing: modeling student proficiency using graph neural network". *IEEE/WIC/ACM International Conference on Web Intelligence*, pages 156–163.
- Nathan, M. J. and A. Petrosino (2003). "Expert blind spot among preservice teachers". *American educational research journal* 40.4, pages 905–928.
- Nelson, J. D. (2005). "Finding useful questions: on Bayesian diagnosticity, probability, impact, and information gain." *Psychological review* 112.4, page 979.
- Newell, A. and H. A. Simon (1972). "Human Problem Solving". Englewood Cliffs, NJ: Prentice-Hall.
- Newell, A. and H. A. Simon (2007). "Computer science as empirical inquiry: Symbols and search". *ACM Turing award lectures*, page 1975.
- Ng, A. Y., S. Russell, et al. (2000). "Algorithms for inverse reinforcement learning." *Icml*. Volume 1. 2, page 2.

- Nguyen, C. V., Y. Li, T. D. Bui, and R. E. Turner (2017). "Variational continual learning". *arXiv preprint arXiv:1710.10628*.
- Nosofsky, R. M. (1986). "Attention, similarity, and the identification–categorization relationship." *Journal of experimental psychology: General* 115.1, page 39.
- Nussenbaum, K. and C. A. Hartley (2019). "Reinforcement learning across development: What insights can we draw from a decade of research?" *Developmental cognitive neuroscience* 40, page 100733.
- Ocana, J. M. C., R. Capobianco, and D. Nardi (2023). "An overview of environmental features that impact deep reinforcement learning in sparse-reward domains". *Journal of Artificial Intelligence Research* 76, pages 1181–1218.
- O'Donnell, T. J. (2015). "Productivity and reuse in language: A theory of linguistic computation and storage". MIT Press.
- Olah, C., N. Cammarata, L. Schubert, G. Goh, M. Petrov, and S. Carter (2020). "Zoom in: An introduction to circuits". *Distill* 5.3, e00024–001.
- Ong, D. C., J. Zaki, and N. D. Goodman (2019). "Computational models of emotion inference in theory of mind: A review and roadmap". *Topics in cognitive science* 11.2, pages 338–357.
- O'regan, J. K. and A. Noë (2001). "A sensorimotor account of vision and visual consciousness". *Behavioral and brain sciences* 24.5, pages 939–973.
- Oudeyer, P.-Y. and F. Kaplan (2007). "What is intrinsic motivation? A typology of computational approaches". *Frontiers in neurorobotics* 1, page 108.
- Pan, A., K. Bhatia, and J. Steinhardt (2022). "The effects of reward misspecification: Mapping and mitigating misaligned models". *arXiv preprint arXiv:2201.03544*.
- Pandey, S. and G. Karypis (2019). "A self-attentive model for knowledge tracing". *arXiv preprint arXiv:1907.06837*.
- Pardos, Z. A. and N. T. Heffernan (2011). "KT-IDEM: Introducing item difficulty to the knowledge tracing model". *User Modeling, Adaption and Personalization: 19th International Conference, UMAP 2011, Girona, Spain, July 11-15, 2011. Proceedings* 19. Springer, pages 243–254.
- Patankar, S. P., D. Zhou, C. W. Lynn, J. Z. Kim, M. Ouellet, H. Ju, P. Zurn, D. M. Lydon-Staley, and D. S. Bassett (2022). "Curiosity as filling, compressing, and reconfiguring knowledge networks". *arXiv preprint arXiv:2204.01182*.
- Pathak, D., P. Agrawal, A. A. Efros, and T. Darrell (2017). "Curiosity-driven exploration by self-supervised prediction". *International conference on machine learning*. PMLR, pages 2778–2787.
- Pavlik, P. I. and J. R. Anderson (2008). "Using a model to compute the optimal schedule of practice." *Journal of Experimental Psychology: Applied* 14.2, page 101.
- Pavlik, P. I., L. G. Eglington, and L. M. Harrell-Williams (2021). "Logistic knowledge tracing: A constrained framework for learner modeling". *IEEE Transactions on Learning Technologies* 14.5, pages 624–639.
- Pearl, J. (1994). "A probabilistic calculus of actions". *Uncertainty in artificial intelligence*. Elsevier, pages 454–462.
- (2009). "Causality". Cambridge university press.
- Pellegrino, J. W., J. D. Bransford, and M. S. Donovan (1999). "How people learn: Bridging research and practice". National Academies Press.
- Penedo, G., Q. Malartic, D. Hesslow, R. Cojocaru, A. Cappelli, H. Alobeidli, B. Pannier, E. Almazrouei, and J. Launay (2023). "The refinedweb dataset for falcon llm: outperforming curated corpora with web data, and web data only". *arXiv preprint arXiv:2306.01116*.
- Perfors, A., J. B. Tenenbaum, and T. Regier (2011). "The learnability of abstract syntactic principles". *Cognition* 118.3, pages 306–338.
- Piaget, J. (1970). "Science of education and the psychology of the child. Trans. D. Coltman."
- Piantadosi, S. T., N. Goodman, B. A. Ellis, and J. Tenenbaum (2008). "A Bayesian model of the acquisition of compositional semantics". *Proceedings of the thirtieth annual conference of the cognitive science society*. Cognitive Science Society Washington DC, pages 1620–1625.
- Piantadosi, S. T. and R. A. Jacobs (2016). "Four problems solved by the probabilistic language of thought". *Current Directions in Psychological Science* 25.1, pages 54–59.
- Piantadosi, S. T., J. B. Tenenbaum, and N. D. Goodman (2012). "Bootstrapping in a language of thought: A formal model of numerical concept learning". *Cognition* 123.2, pages 199–217.
- Piech, C., J. Bassen, J. Huang, S. Ganguli, M. Sahami, L. J. Guibas, and J. Sohl-Dickstein (2015). "Deep knowledge tracing". *Advances in neural information processing systems* 28.
- Planton, S., T. van Kerkoerle, L. Abbi, M. Maheu, F. Meyniel, M. Sigman, L. Wang, S. Figueira, S. Romano, and S. Dehaene (2021). "A theory of memory for binary sequences: Evidence for a mental compression algorithm in humans". *PLoS computational biology* 17.1, e1008598.

- Poli, F, G Serino, R. Mars, and S Hunnius (2020). "Infants tailor their attention to maximize learning". *Science advances* 6.39, eabb5053.
- Pothos, E. M. (2007). "Theories of artificial grammar learning." *Psychological bulletin* 133.2, page 227.
- Pressey, S. L. (1926). "A simple apparatus which gives tests and scores-and teaches". *Sch. & Soc.* 23, pages 373–376.
- Quilty-Dunn, J., N. Porot, and E. Mandelbaum (2023). "The best game in town: The reemergence of the language-of-thought hypothesis across the cognitive sciences". *Behavioral and Brain Sciences* 46, e261.
- Raffin, A., A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann (2021). "Stable-baselines3: Reliable reinforcement learning implementations". *Journal of machine learning research* 22.268, pages 1–8.
- Rao, R. P. and D. H. Ballard (1999). "Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects". *Nature neuroscience* 2.1, pages 79–87.
- Reali, F. and T. L. Griffiths (2009). "The evolution of frequency distributions: Relating regularization to inductive biases through iterated learning". *Cognition* 111.3, pages 317–328.
- Reber, A. S. (1967). "Implicit learning of artificial grammars". *Journal of verbal learning and verbal behavior* 6.6, pages 855–863.
- Rescorla, M. (2015). "The computational theory of mind".
- Rescorla, R. (2014). "Pavlovian second-order conditioning (psychology revivals): studies in associative learning". *Psychology Press*.
- Rescorla, R. A. (1972). "Informational variables in Pavlovian conditioning". *Psychology of learning and motivation*. Volume 6. Elsevier, pages 1–46.
- Rice, H. G. (1953). "Classes of recursively enumerable sets and their decision problems". *Transactions of the American Mathematical society* 74.2, pages 358–366.
- Rigoux, L., K. E. Stephan, K. J. Friston, and J. Daunizeau (2014). "Bayesian model selection for group studies—revisited". *Neuroimage* 84, pages 971–985.
- Rissanen, J. (1986). "Stochastic complexity and modeling". *The annals of statistics*, pages 1080–1100.
- Ritter, F. E., J. Nerb, E. Lehtinen, and T. M. O'Shea (2007). "In order to learn: How the sequence of topics influences learning". *Oxford University Press*.
- Rittle-Johnson, B., A. M. Loehr, and K. Durkin (2017). "Promoting self-explanation to improve mathematics learning: A meta-analysis and instructional design principles". *ZDM* 49.4, pages 599–611.
- Rolnick, D., A. Ahuja, J. Schwarz, T. Lillicrap, and G. Wayne (2019). "Experience replay for continual learning". *Advances in neural information processing systems* 32.
- Rosch, E., C. B. Mervis, W. D. Gray, D. M. Johnson, and P. Boyes-Braem (1976). "Basic objects in natural categories". *Cognitive psychology* 8.3, pages 382–439.
- Rosenbaum, C., I. Cases, M. Riemer, and T. Klinger (2019). "Routing networks and the challenges of modular and compositional computation". *arXiv preprint arXiv:1904.12774*.
- Rosenbaum, D. A., R. G. Cohen, S. A. Jax, D. J. Weiss, and R. Van Der Wel (2007). "The problem of serial order in behavior: Lashley's legacy". *Human movement science* 26.4, pages 525–554.
- Rosenblatt, F. (1958). "The perceptron: a probabilistic model for information storage and organization in the brain." *Psychological review* 65.6, page 386.
- Rozenblit, L. and F. Keil (2002). "The misunderstood limits of folk science: An illusion of explanatory depth". *Cognitive science* 26.5, pages 521–562.
- Rubino, V., P. Dayan, and C. M. Wu (submitted). "Simplicity guides the discovery and use of compositionality". *PsyArXiv*.
- Rule, J. S., S. T. Piantadosi, A. Cropper, K. Ellis, M. Nye, and J. B. Tenenbaum (2024). "Symbolic metaprogram search improves learning efficiency and explains rule learning in humans". *Nature Communications* 15.1, page 6847.
- Rule, J. S., E. Schulz, S. T. Piantadosi, and J. B. Tenenbaum (2018). "Learning list concepts through program induction". *BioRxiv*, page 321505.
- Rule, J. S., J. B. Tenenbaum, and S. T. Piantadosi (2020). "The child as hacker". *Trends in Cognitive Sciences* 24.11, pages 900–915.
- Rumelhart, D. E. (2017). "Schemata: The building blocks of cognition". *Theoretical issues in reading comprehension*. Routledge, pages 33–58.
- Rumelhart, D. E., G. E. Hinton, and R. J. Williams (1986). "Learning representations by back-propagating errors". *nature* 323.6088, pages 533–536.
- Rumelhart, D. E. and A. Ortony (2017). "The representation of knowledge in memory 1". *Schooling and the acquisition of knowledge*. Routledge, pages 99–135.
- Russell, S. and E. Wefald (1991). "Principles of metareasoning". *Artificial intelligence* 49.1-3, pages 361–395.

- Russell, S. J. and D. Subramanian (1994). "Provably bounded-optimal agents". *Journal of Artificial Intelligence Research* 2, pages 575–609.
- Rusu, A. A., N. C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick, K. Kavukcuoglu, R. Pascanu, and R. Hadsell (2016). "Progressive neural networks". *arXiv preprint arXiv:1606.04671*.
- Ryan, R. M. and E. L. Deci (2000). "Intrinsic and extrinsic motivations: Classic definitions and new directions". *Contemporary educational psychology* 25.1, pages 54–67.
- Sablé-Meyer, M., K. Ellis, J. Tenenbaum, and S. Dehaene (2022). "A language of thought for the mental representation of geometric shapes". *Cognitive Psychology* 139, page 101527.
- Saffran, J. R., R. N. Aslin, and E. L. Newport (1996). "Statistical learning by 8-month-old infants". *science* 274.5294, pages 1926–1928.
- Saffran, J. R., E. K. Johnson, R. N. Aslin, and E. L. Newport (1999). "Statistical learning of tone sequences by human infants and adults". *Cognition* 70.1, pages 27–52.
- Sakai, K., K. Kitaguchi, and O. Hikosaka (2003). "Chunking during human visuomotor sequence learning". *Experimental brain research* 152.2, pages 229–242.
- Salge, C., C. Glackin, and D. Polani (2014). "Empowerment—an introduction". *Guided Self-Organization: Inception*, pages 67–114.
- Salmon, W. C. (2020). "Scientific explanation and the causal structure of the world".
- Sanborn, A. N. (2017). "Types of approximation for probabilistic cognition: Sampling and variational". *Brain and cognition* 112, pages 98–101.
- Sancaktar, C., C. Gumbsch, A. Zadaianchuk, P. Kolev, and G. Martius (2025). "SENSEI: Semantic exploration guided by foundation models to learn versatile world models". *arXiv preprint arXiv:2503.01584*.
- Sao Pedro, M., R. Baker, and J. Gobert (2013). "Incorporating scaffolding and tutor context into bayesian knowledge tracing to predict inquiry skill acquisition". *Educational Data Mining 2013*. Citeseer.
- Särkkä, S. and A. Solin (2019). "Applied stochastic differential equations". Volume 10. *Cambridge University Press*.
- Saxe, R. and N. Kanwisher (2013). "People thinking about thinking people: the role of the temporo-parietal junction in "theory of mind"". *Social neuroscience. Psychology Press*, pages 171–182.
- Schank, R. C. and R. P. Abelson (2013). "Scripts, plans, goals, and understanding: An inquiry into human knowledge structures". *Psychology press*.
- Schmidhuber, J. (1991). "Curious model-building control systems". *Proc. international joint conference on neural networks*, pages 1458–1463.
- (2010). "Formal theory of creativity, fun, and intrinsic motivation (1990–2010)". *IEEE transactions on autonomous mental development* 2.3, pages 230–247.
- Schulman, J., F. Wolski, P. Dhariwal, A. Radford, and O. Klimov (2017). "Proximal policy optimization algorithms". *arXiv preprint arXiv:1707.06347*.
- Schulz, L. E. and E. B. Bonawitz (2007). "Serious fun: preschoolers engage in more exploratory play when evidence is confounded." *Developmental psychology* 43.4, page 1045.
- Schwartenbeck, P., T. FitzGerald, R. J. Dolan, and K. Friston (2013). "Exploration, novelty, surprise, and free energy minimization". *Frontiers in psychology* 4, page 710.
- Schwarzer, M., N. Rajkumar, M. Noukhovitch, A. Anand, L. Charlin, R. D. Hjelm, P. Bachman, and A. C. Courville (2021). "Pretraining representations for data-efficient reinforcement learning". *Advances in Neural Information Processing Systems* 34, pages 12686–12699.
- Schönfinkel, M. (1924). "Über die Bausteine der mathematischen Logik". *de. Mathematische Annalen* 92.3, pages 305–316. (Visited on 01/31/2024).
- Seitzer, M., B. Schölkopf, and G. Martius (2021). "Causal Influence Detection for Improving Efficiency in Reinforcement Learning". *Advances in Neural Information Processing Systems*. Edited by A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan.
- Sekar, R., O. Rybkin, K. Daniilidis, P. Abbeel, D. Hafner, and D. Pathak (2020). "Planning to explore via self-supervised world models". *International conference on machine learning*. PMLR, pages 8583–8592.
- Selent, D., T. Patikorn, and N. Heffernan (2016). "Assistments dataset from multiple randomized controlled experiments". *Proceedings of the Third (2016) ACM Conference on Learning@ Scale*, pages 181–184.
- Serra, J., D. Suris, M. Miron, and A. Karatzoglou (2018). "Overcoming catastrophic forgetting with hard attention to the task". *International conference on machine learning*. PMLR, pages 4548–4557.
- Settles, B. and B. Meeder (2016). "A trainable spaced repetition model for language learning". *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 1: long papers)*, pages 1848–1858.
- Shafro, P., N. D. Goodman, and T. L. Griffiths (2014). "A rational account of pedagogical reasoning: Teaching by, and learning from, examples". *Cognitive psychology* 71, pages 55–89.

- Shanks, D. R. (1995). "The psychology of associative learning". Volume 13. *Cambridge University Press Cambridge*.
- Shannon, C. E. (1959). "Coding theorems for a discrete source with a fidelity criterion". *IRE Nat. Conv. Rec* 4.142-163, page 1.
- Shannon, C. E. (1948). "A mathematical theory of communication". *The Bell system technical journal* 27.3, pages 379–423.
- Shannon, C. E. and W. Weaver (1949). "The Mathematical Theory of Communication". *University of Illinois Press*.
- Sharma, S., A. Curtis, M. Kryven, J. Tenenbaum, and I. Fiete (2021). "Map induction: Compositional spatial submap learning for efficient exploration in novel environments". *arXiv preprint arXiv:2110.12301*.
- Shen, S., Q. Liu, E. Chen, Z. Huang, W. Huang, Y. Yin, Y. Su, and S. Wang (2021). "Learning process-consistent knowledge tracing". *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pages 1452–1460.
- Shi, Y., B. Paige, P. Torr, et al. (2019). "Variational mixture-of-experts autoencoders for multi-modal deep generative models". *Advances in Neural Information Processing Systems* 32.
- Shin, D., Y. Shim, H. Yu, S. Lee, B. Kim, and Y. Choi (2021). "Saint+: Integrating temporal features for ednet correctness prediction". *LAK21: 11th International Learning Analytics and Knowledge Conference*, pages 490–496.
- Siegler, R. S. (2004). "U-shaped interest in u-shaped development-and what it means". *Journal of Cognition and Development* 5.1, pages 1–10.
- Sievers, B., L. Polansky, M. Casey, and T. Wheatley (2013). "Music and movement share a dynamic structure that supports universal expressions of emotion". *Proceedings of the national academy of sciences* 110.1, pages 70–75.
- Silver, D., J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, et al. (2017). "Mastering the game of go without human knowledge". *nature* 550.7676, pages 354–359.
- Silver, D. and J. Veness (2010). "Monte-Carlo planning in large POMDPs". *Advances in neural information processing systems* 23.
- Simon, H. A. (1955). "A behavioral model of rational choice". *The quarterly journal of economics*, pages 99–118.
- (2014). "Cognitive architectures and rational analysis: Comment". *Architectures for intelligence. Psychology Press*, pages 25–39.
- Sims, C. R. (2016). "Rate–distortion theory and human perception". *Cognition* 152, pages 181–198.
- Smallwood, R. D. and E. J. Sondik (1973). "The optimal control of partially observable Markov processes over a finite horizon". *Operations Research* 21.5, pages 1071–1088.
- Smith, A. C., L. M. Frank, S. Wirth, M. Yanike, D. Hu, Y. Kubota, A. M. Graybiel, W. A. Suzuki, and E. N. Brown (2004). "Dynamic analysis of learning in behavioral experiments". *Journal of Neuroscience* 24.2, pages 447–461.
- Smith, J. E. and R. L. Winkler (2006). "The optimizer's curse: Skepticism and postdecision surprise in decision analysis". *Management Science* 52.3, pages 311–322.
- Smolensky, P. (1988). "On the proper treatment of connectionism". *Behavioral and brain sciences* 11.1, pages 1–23.
- Soldaini, L., R. Kinney, A. Bhagia, D. Schwenk, D. Atkinson, R. Authur, B. Bogin, K. Chandu, J. Dumas, Y. Elazar, et al. (2024). "Dolma: An open corpus of three trillion tokens for language model pretraining research". *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15725–15788.
- Solomonoff, R. J. (1964). "A formal theory of inductive inference. Part I". *Information and Control* 7.1, pages 1–22.
- Sow, D. M. and A. Eleftheriadis (2003). "Complexity distortion theory". *IEEE Transactions on Information Theory* 49.3, pages 604–608.
- Spirtes, P., C. N. Glymour, and R. Scheines (2000). "Causation, prediction, and search". *MIT press*.
- Still, S. and D. Precup (2012). "An information-theoretic approach to curiosity-driven reinforcement learning". *Theory in Biosciences* 131.3, pages 139–148.
- Sucholutsky, I., K. M. Collins, M. Malaviya, N. Jacoby, W. Liu, T. Sumers, M. Korakakis, U. Bhatt, M. K. Ho, J. B. Tenenbaum, et al. (2025). "Representational Alignment Supports Effective Teaching". *ICLR 2025 Workshop on Bidirectional Human-AI Alignment*.
- Sun, R., H. Zang, X. Li, and R. Islam (2024). "Learning latent dynamic robust representations for world models". *arXiv preprint arXiv:2405.06263*.
- Sutton, R. S. (1991). "Dyna, an integrated architecture for learning, planning, and reacting". *ACM Sigart Bulletin* 2.4, pages 160–163.
- Sutton, R. S., A. G. Barto, et al. (1998). "Introduction to reinforcement learning". Volume 135. *MIT press Cambridge*.

- Sutton, R. S., D. Precup, and S. Singh (1999). "Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning". *Artificial intelligence* 112.1-2, pages 181–211.
- Tang, H., D. Key, and K. Ellis (2024). "Worldcoder, a model-based llm agent: Building world models by writing code and interacting with the environment". *Advances in Neural Information Processing Systems* 37, pages 70148–70212.
- Tang, H., R. Houthoofd, D. Foote, A. Stooke, O. Xi Chen, Y. Duan, J. Schulman, F. DeTurck, and P. Abbeel (2017). "# exploration: A study of count-based exploration for deep reinforcement learning". *Advances in neural information processing systems* 30.
- Teh, Y. W. (2006). "A hierarchical Bayesian language model based on Pitman-Yor processes". *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 985–992.
- Tellex, S., T. Kollar, S. Dickerson, M. Walter, A. Banerjee, S. Teller, and N. Roy (2011). "Understanding natural language commands for robotic navigation and mobile manipulation". *Proceedings of the AAAI conference on artificial intelligence*. Volume 25. 1, pages 1507–1514.
- Ten, A., P.-Y. Oudeyer, M. Sakaki, and K. Murayama (2024). "The curious U: Integrating theories linking knowledge and information-seeking behavior". *PsyArXiv*.
- (2025). "The curious U: Integrating theories linking knowledge and information-seeking behavior". *Open Mind* 9, pages 1763–1785.
- Tenenbaum, J. B., T. L. Griffiths, and C. Kemp (2006). "Theory-based Bayesian models of inductive learning and reasoning". *Trends in cognitive sciences* 10.7, pages 309–318.
- Tenenbaum, J. B., C. Kemp, T. L. Griffiths, and N. D. Goodman (2011). "How to grow a mind: Statistics, structure, and abstraction". *science* 331.6022, pages 1279–1285.
- Tishby, N., F. C. Pereira, and W. Bialek (2000). "The information bottleneck method". *arXiv preprint physics/0004057*.
- Tobin, J., R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel (2017). "Domain randomization for transferring deep neural networks from simulation to the real world". *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, pages 23–30.
- Todorov, E. and M. I. Jordan (2002). "Optimal feedback control as a theory of motor coordination". *Nature neuroscience* 5.11, pages 1226–1235.
- Tolman, E. C. (1948). "Cognitive maps in rats and men." *Psychological review* 55.4, page 189.
- Tomasello, M., A. C. Kruger, and H. H. Ratner (1999). "Cultural learning". *Lev Vygotsky: critical assessments* 4, pages 101–143.
- Tong, S., Q. Liu, W. Huang, Z. Hunag, E. Chen, C. Liu, H. Ma, and S. Wang (2020). "Structure-based knowledge tracing: An influence propagation view". *2020 IEEE international conference on data mining (ICDM)*. IEEE, pages 541–550.
- Tsividis, P. A., J. Loula, J. Burga, N. Foss, A. Campero, T. Pouncy, S. J. Gershman, and J. B. Tenenbaum (2021). "Human-level reinforcement learning through theory-based modeling, exploration, and planning". *arXiv preprint arXiv:2107.12544*.
- Turing, A. M. (1987). "Computing machinery and intelligence (1950)". *Mind* 96.386, pages 33–60.
- Turing, A. M. et al. (1936). "On computable numbers, with an application to the Entscheidungsproblem". *J. of Math* 58.345-363, page 5.
- Turner, A. M., L. Smith, R. Shah, A. Critch, and P. Tadepalli (2019). "Optimal policies tend to seek power". *arXiv preprint arXiv:1912.01683*.
- Tversky, B. (1996). "Spatial perspective in descriptions."
- Underwood, B. J. and L. Postman (1960). "Extraexperimental sources of interference in forgetting." *Psychological Review* 67.2, page 73.
- Valkov, L., D. Chaudhari, A. Srivastava, C. Sutton, and S. Chaudhuri (2018). "Houdini: Lifelong learning as program synthesis". *Advances in neural information processing systems* 31.
- VanLehn, K. (2011). "The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems". *Educational psychologist* 46.4, pages 197–221.
- Varela, F. J., E. Rosch, and E. Thompson (1991). "The embodied mind". *The embodied mind: Cognitive science and human experience*.
- Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin (2017a). "Attention is all you need". *Advances in neural information processing systems* 30.
- (2017b). "Attention is all you need". *Advances in neural information processing systems* 30.
- Vereshchagin, N. and P. Vítányi (2005). "On lossy compression".

- Verwey, W. B. and Y. Dronkert (1996). "Practicing a structured continuous key-pressing task: Motor chunking or rhythm consolidation?" *Journal of motor behavior* 28.1, pages 71–79.
- Voorhees, E. M. et al. (1999). "The TREC-8 question answering track report". *Trec*. Volume 99, pages 77–82.
- Vygotsky, L. S. (1978). "Mind in society: The development of higher psychological processes". Volume 86. *Harvard university press*.
- Walker, C. M., T. Lombrozo, C. H. Legare, and A. Gopnik (2014). "Explaining prompts children to privilege inductively rich properties". *Cognition* 133.2, pages 343–357.
- Wallace, C. S., D. M. Boulton, et al. (1968). "An information measure for classification." *Comput. J.* 11.2, pages 185–194.
- Walsh, M. M., K. A. Gluck, G. Gunzelmann, T. Jastrzembski, and M. Krusmark (2018). "Evaluating the theoretic adequacy and applied potential of computational models of the spacing effect". *Cognitive science* 42, pages 644–691.
- Wang, C., W. Ma, M. Zhang, C. Lv, F. Wan, H. Lin, T. Tang, Y. Liu, and S. Ma (2021). "Temporal cross-effects in knowledge tracing". *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pages 517–525.
- Wang, G., Y. Xie, Y. Jiang, A. Mandlekar, C. Xiao, Y. Zhu, L. Fan, and A. Anandkumar (2023). "Voyager: An open-ended embodied agent with large language models". *arXiv preprint arXiv:2305.16291*.
- Wang, H., A. Bhattacharya, D. Pati, and Y. Yang (2022). "Structured variational inference in Bayesian state-space models". *International Conference on Artificial Intelligence and Statistics*. PMLR, pages 8884–8905.
- Wenzek, G., M.-A. Lachaux, A. Conneau, V. Chaudhary, F. Guzmán, A. Joulin, and E. Grave (2020). "CCNet: Extracting high quality monolingual datasets from web crawl data". *Proceedings of the twelfth language resources and evaluation conference*, pages 4003–4012.
- White, R. W. (1959). "Motivation reconsidered: the concept of competence." *Psychological review* 66.5, page 297.
- Wiggins, J. (2007). "Compositional process in music". *International handbook of research in arts education*. Springer, pages 453–476.
- Williams, P. L. and R. D. Beer (2010). "Nonnegative decomposition of multivariate information". *arXiv preprint arXiv:1004.2515*.
- Wilson, R. C. and A. G. Collins (2019). "Ten simple rules for the computational modeling of behavioral data". *elife* 8, e49547.
- Wilson, R. C., A. Geana, J. M. White, E. A. Ludvig, and J. D. Cohen (2014). "Humans use directed and random exploration to solve the explore–exploit dilemma." *Journal of experimental psychology: General* 143.6, page 2074.
- Wittgenstein, L. (1953). "Philosophical investigations. Philosophische untersuchungen."
- Wixted, J. T. and E. B. Ebbesen (1997). "Genuine power curves in forgetting: A quantitative analysis of individual subject forgetting functions". *Memory & cognition* 25, pages 731–739.
- Wolpert, D. M., J. Diedrichsen, and J. R. Flanagan (2011). "Principles of sensorimotor learning". *Nature reviews neuroscience* 12.12, pages 739–751.
- Wolpert, D. H. and W. G. Macready (2002). "No free lunch theorems for optimization". *IEEE transactions on evolutionary computation* 1.1, pages 67–82.
- Wong, L., K. M. Collins, L. Ying, C. E. Zhang, A. Weller, T. Gerstenberg, T. O'Donnell, A. K. Lew, J. D. Andreas, J. B. Tenenbaum, et al. (2025). "Modeling open-world cognition as on-demand synthesis of probabilistic models". *arXiv preprint arXiv:2507.12547*.
- Woodward, J. (2005). "Making things happen: A theory of causal explanation". *Oxford university press*.
- Wu, C. M., R. Dale, and R. D. Hawkins (2024). "Group coordination catalyzes individual and cultural intelligence". *Open Mind* 8, pages 1037–1057.
- Wu, S., N. Éltető, I. Dasgupta, and E. Schulz (2023). "Chunking as a rational solution to the speed–accuracy trade-off in a serial reaction time task". *Scientific reports* 13.1, page 7680.
- Xia, L. and A. G. Collins (2021). "Temporal and state abstractions for efficient learning, transfer, and composition in humans." *Psychological review* 128.4, page 643.
- Yang, B., W.-t. Yih, X. He, J. Gao, and L. Deng (2014). "Embedding entities and relations for learning and inference in knowledge bases". *arXiv preprint arXiv:1412.6575*.
- Yudelson, M. V., K. R. Koedinger, and G. J. Gordon (2013). "Individualized Bayesian knowledge tracing models". *Artificial Intelligence in Education: 16th International Conference, AIED 2013, Memphis, TN, USA, July 9-13, 2013. Proceedings* 16. Springer, pages 171–180.
- Zenon, A., O. Solopchuk, and G. Pezzulo (2019). "An information-theoretic perspective on the costs of cognition". *Neuropsychologia* 123, pages 5–18.

- Zhang, C., J. B  tepage, H. Kjellstr  m, and S. Mandt (2018). "Advances in variational inference". *IEEE transactions on pattern analysis and machine intelligence* 41.8, pages 2008–2026.
- Zhao, B., C. G. Lucas, and N. R. Bramley (2023). "A model of conceptual bootstrapping in human cognition". *Nature Human Behaviour*, pages 1–12.
- Zheng, X., B. Aragam, P. K. Ravikumar, and E. P. Xing (2018). "Dags with no tears: Continuous optimization for structure learning". *Advances in neural information processing systems* 31.
- Zhou, D., D. M. Lydon-Staley, P. Zurn, and D. S. Bassett (2020). "The growth and form of knowledge networks by kinesthetic curiosity". *Current opinion in behavioral sciences* 35, pages 125–134.
- Zhou, H., R. Bamler, C. M. Wu, and A. Tejero-Cantero (2024a). "Predictive, scalable and interpretable knowledge tracing on structured domains". *The Twelfth International Conference on Learning Representations*.
- Zhou, H., B. Besch, C. M. Wu, and T. Gerstenberg (2026a). "Effective Explanations Support Planning Under Uncertainty". *Proceedings of the 48th Annual Conference of the Cognitive Science Society*.
- Zhou, H., F. Mantiuk, D. G. Nagy, and C. M. Wu (2025). "Agent-centric learning: from external reward maximization to internal knowledge curation". *Reinforcement Learning Conference: Finding the Frame Workshop*.
- Zhou, H., D. G. Nagy, P. Dayan, and C. M. Wu (2026b). "Path-dependent program induction under resource constraints explains human sequence learning". *arXiv preprint arXiv:2606.20623*.
- Zhou, H., D. G. Nagy, and C. M. Wu (2024b). "Harmonizing Program Induction with Rate-Distortion Theory". *Proceedings of the 46th Annual Conference of the Cognitive Science Society*. Edited by S. Frank, M. Toneva, A. Mackey, and E. Hazeltine. *Cognitive Science Society*, pages 2511–2518.
- Zipf, G. K. (2016). "Human behavior and the principle of least effort: An introduction to human ecology". *Ravenio books*.
- Zurn, P. and D. S. Bassett (2022). "Curious minds: The power of connection". *Mit Press*.