Fostering Event-Predictive Encodings in Recurrent Neural Networks

PhD Thesis
2022

Author: Dania Humaidan

Supervisor: Martin V. Butz

EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

imprs-is

Fostering Event-Predictive Encodings in Recurrent Neural Networks

Dissertation

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

vorgelegt von
Dania Humaidan
aus Swaida/Syrien

Tübingen
2022

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

Tag der mündlichen Qualifikation:	04.05.2022
Dekan:	Prof. Dr. Thilo Stehle
1. Berichterstatter:	Prof. Dr. Martin V. Butz
2. Berichterstatter:	Prof. Dr.-Ing. Hendrik Lensch

The true scientist must dream.
For if they do not imagine the world and dream,
they will repeat what others have done,
and will add nothing.

Ahmad Zewail (1946-2016), Egyptian-American scientist, Nobel Laureate in Chemistry 1999

العالم الحقيقي لا بد أن يحلم.
أما إذا لم يتخيل العالم ويحلم، فسيكرر ما فعله الآخرون
ولن يضيف شيئا.

أحمد زويل (١٩٦٤-٢٠١٦)، العالم المصري الأميركي الحائز على جائزة نوبل في الكيمياء لعام ١٩٩٩

Acknowledgments

I would like to thank Prof. Martin V. Butz, for his supervision and unlimited support throughout my PhD. Not only to guide me to achieve this work, but also to teach me how to be a good researcher. I would also like to thank Prof. Hendrik Lensch and Prof. Sebastian Trimpe for the valuable feedback, and the International Max Planck Research School for Intelligent Systems for the support. A big thank you goes to my colleagues and friends, who provided the utmost support, scientifically and emotionally. Mum and my family, thank you for always being there for me. And to Tuebingen I say, you made it, I am staying for more adventures.

Publications

The work achieved in this thesis has been partially published in the following journal or conference manuscripts and workshop posters. This has also been declared in the respective footnotes.

Butz, M. V., Bilkey, D., Humaidan, D., Knott, A., Otte, S. (2018). Event Inference for Compaction, Imagination, and Control in Recurrent Forward Models. Poster at the 13th Women in Machine Learning workshop.

Butz, M.V., Bilkey, D., Humaidan, D., Knott, A., Otte, S. (2019) Learning, planning, and control in a monolithic neural event inference architecture. Neural Networks 117, 135–144

Butz, M.V., Menge, T., Humaidan, D., Otte, S. (2019) Inferring event-predictive goal- directed object manipulations in REPRISE. Artificial Neural Networks and Machine Learning – ICANN 2019 (11727), 639–653

Humaidan, D., Butz, M. V. (2019) Event Boundaries Detection Using Artificial Neural Networks. Poster at the Deep Learning and Reinforcement Learning Summer School (DLRLSS). 24.07-02.08.2019 Alberta Machine Intelligence Institute, Edmonton, Canada

Humaidan, D., Otte, S., Butz, M. V. (2020). Fostering event compression using gated surprise. In I. Farkas, P. Masulli, & S. Wermter (Eds.), International conference on artificial neural networks – icann 2020 (pp. 155–167). Springer.

Humaidan, D., Butz, M. V. (2020). SUGAR: a surprise-gated recurrent neural network model for event compression. Poster at the 15th Women in Machine Learning workshop co-located with NeurIPS 2020

Humaidan, D., Otte, S., Gumbsch, C., Charley, W., Butz, M. V. (2021) Latent Event-Predictive Encodings through Counterfactual Regularization. The 43rd annual meeting of the cognitive science society.

The work has also been presented in the following talks:

”Event boundaries detection using recurrent neural networks”. Talk at the Max Planck Institute for Intelligent Systems, Stuttgart Campus. 18-10-2018.

”Event boundaries detection using artificial neural networks”. Talk at the Bosch Center for Artificial Intelligence, Renningen, Germany. 05-07-2019.

”Using gated surprise to create and manage event compressions”. Talk at the ”10 Minutes” seminar organized at the Tuebingen AI building. 10-01-2020.

”Using gated surprise to create and manage event compressions”. 1st German Society for Cognitive Science (Gesellschaft für Kognitionswissenschaft) Doctoral Symposium on Cognitive Science. Tuebingen, 24-01-2020.

Contents

List of Figures

List of Tables

Abstract

Our interaction with the world is based on the information that we receive and actions that we perform. When we observe the surrounding environment, we obtain new information, which we use to update our beliefs and plan our next steps. It is suggested that we organize this information within separable units, called events. The time points at which an event ends and the next starts are called event boundaries. Understanding the ongoing events and the boundaries between them helps us to better plan the upcoming steps towards our goals deeper into the future. As a result, we can interact with the environment in more versatile, and longer-reaching goal-directed manner.

Alongside the use of sensory information to predict upcoming events, it appears that an additional inference process is implemented to infer the causes of our sensory experiences, as suggested by the Bayesian brain hypothesis. When our observations do not match our predictions, we get surprised and try to update our beliefs based on the new information. The update is based on the prediction error between the internal expectations and the sensory observations. In order to minimize the error, we need to be able to predict the upcoming states as accurately as possible, which means that we need to avoid being faced with unpredictable, surprising states. Modeling the temporal evolution of events, and the role of surprise in defining event boundaries, has attracted lots of attention. The structure within which such modeling can take place would represent a schema for the Bayesian brain model. Moreover, seeing the connections between different areas of the cortex, and the different temporal dynamics of them, evidence has accumulated that hierarchical, predictive structures develop in our brain, with lower sensory processing layers, which deal with concrete information, and deeper, or upper, layers that are responsible for abstraction, fitting the sensory information into event-characterizing, compact latent encodings.

This thesis explores how to construct a neural network-based, hierarchical architecture for surprise-based, event-predictive modeling. First, we explore retrospective gating by tackling the challenge of different moving vehicles as events. Then, we contrast this with prospective gating. In this case we use a set of temporal functions and show that events can be segmented when surprise signals are available. Moreover,

we show that event boundary anticipation is useful to enhance the compression of event-characterizing latent encodings. Finally, we extend the structure with a counterfactual regularization term, showing that the regularization enhances the stability and robustness of the latent event encodings in a symbolic sequence prediction task. With the achieved robust development of event-predictive encodings, the presented surprise-gated event segmentation structure could be employed in more complex systems to perform other decision-making tasks. Particularly the controlled opening of the surprise gate could be useful in other systems where the efficient and selective activation of contextual information is required.

Abstrakt

Unsere Interaktion mit der Welt basiert auf den Informationen, die wir erhalten, und den Handlungen, die wir ausführen. Wenn wir die Umgebung beobachten, erhalten wir neue Informationen, die wir verwenden, um unsere Überzeugungen zu aktualisieren und unsere nächsten Schritte zu planen. Es wird vorgeschlagen, dass wir diese Informationen in trennbare Einheiten, sogenannte Ereignisse, kodieren. Die Zeitpunkte, an denen ein Ereignis endet und das nächste beginnt, werden Ereignisgrenzen genannt. Das Verständnis der laufenden Ereignisse und der Grenzen zwischen ihnen hilft uns, die bevorstehenden Schritte in Richtung unserer Ziele besser zu planen. Dadurch können wir uns besser, zielorientiert verhalten.

Neben der Nutzung der sensorischen Informationen zur Vorhersage bevorstehender Ereignisse wird im Gehirn auch ein Inferenzprozess durchgeführt, der die Ursachen dieser sensorischen Erfahrungen ergründet. Dies wird auch von der Bayesschen Gehirnhypothese postuliert. Wenn unsere Beobachtungen nicht mit den Vorhersagen übereinstimmen, sind wir überrascht und versuchen, unsere Überzeugungen basierend auf den neuen Informationen zu aktualisieren. Die Aktualisierung erfolgt unter Verwendung des Vorhersagefehlers zwischen den internen Erwartungen und den sensorischen Beobachtungen. Um den Fehler zu minimieren, müssen wir die kommenden Zustände so genau wie möglich vorhersagen, das heißt, wir müssen vermeiden, mit unvorhersehbaren, überraschenden Zuständen konfrontiert zu werden. Die Modellierung der zeitlichen Entwicklung von Ereignissen und die Rolle von Überraschung bei der Definition der Ereignisgrenzen hat viel Aufmerksamkeit auf sich gezogen. Die Struktur, innerhalb derer eine solche Modellierung stattfinden kann, würde ein Schema für das Bayessches Gehirnmodell darstellen. Angesichts der Verbindungen zwischen verschiedenen Bereichen des Kortex und ihrer unterschiedlichen zeitlichen Entwicklungen wäre eine wahrscheinliche Struktur eine hierarchische Struktur mit einer unteren sensorischen Verarbeitungsschicht, die sich mit konkreten Informationen befasst, und einer oberen Schicht, die für die Abstraktion verantwortlich ist, in dem sie sensorische Informationen in spezifische Ereignisse und konzeptuelle Abstraktionen davon kodiert.

Diese Dissertation untersucht, wie eine hierarchische Architektur für überraschungsbasierte

ereignisprädiktive Modellierung konstruiert werden kann. Zuerst untersuchen wir retrospektives Gating, indem wir die Herausforderung unterschiedlicher sich bewegender Fahrzeuge als Ereignisse angehen. Dann stellen wir dies dem prospektiven Gating gegenüber. In diesem Fall verwenden wir eine Reihe von Zeitfunktionen und zeigen, dass Ereignisse segmentiert werden können, wenn Überraschungssignale verfügbar sind. Darüber hinaus zeigen wir, dass die Antizipation von Ereignisgrenzen nützlich ist, um die Komprimierung latenter Ereigniskodierungen zu verbessern. Schließlich erweitern wir die Struktur um einen kontrafaktischen Regularisierungsterm und zeigen, dass die Regularisierung die Stabilität und Robustheit der latenten Ereigniskodierungen in einer symbolischen Sequenzvorhersageaufgabe verbessert. Mit der erreichten robusten Förderung ereignisprädiktiver Kodierungen könnte die vorgestellte Ereignissegmentierungsstruktur in komplexeren Systemen weiter untersucht werden, um andere Entscheidungsaufgaben zu erfüllen. So kann erwartet werden, dass die zielgerichtete Kontrolle und Inferenz von Ereignissen und Ereignisgrenzen auch in anderen Systemen nützlich sein wird, insbesondere wenn effiziente und selektive Aktivierungen von Kontextinformationen hilfreich sind.

Chapter 1

Introduction

In this chapter, I will start with the big picture that represents the motivation behind the work in this thesis: achieving a step towards understanding human cognition. I will start by setting some definitions and shedding light on the relevant theories and models related to the study of human cognition. Then, I will zoom in to the aspect of how separable units are formed from the stream of information that we receive, called events. Event cognition represents the main focus of this work.

Next, since the tools used in this work come from the field of artificial intelligence, I will elaborate on the contributions of this field to the study of cognition. Specifically, the theories and models introduced to study cognitive functions, with a focus on event-predictive models.

I will end the chapter with a summary of the presented definitions and models, and explain the current state of event-cognitive modeling, which represents the starting point of my contributions in this work.

1.1 Human Cognition

1.1.1 The early investigation of cognition

The study of human cognition does not belong to one field only, but it requires combining aspects and knowledge from different fields, including but not limited to philosophy, psychology, linguistics and biology.

Although the philosophical questions about how the brain works have been long discussed, the movement of studying the mind and its intellectual abilities started with the cognitive revolution in the 1950's [78]. Long before this, the famous statement of Descart: "I think, therefore I am" indicated that our ideas are what define us. As a result, two movements were developed: rationalism and empiricism. While rationalism assumes that our realization of the surrounding world is based on reason and logic, empiricism drives learning back to experiences only. Interestingly, the empiricist philosopher Hume considered our brain to be a system that can use the previous experiences to anticipate what might happen next [9]. On the other hand, the Bayesian brain theory suggests that the brain actively infers the causes of its experiences, hence called an active inference system [26]. If the system, our brain in this case, receives feedback regarding its predictions, then the brain might actually be using a sort of a predictive feed-forward model. It receives an observation as an input, processes it and outputs a prediction of the observation expected in the next time step. Based on the received information, the brain decides on the suitable behavior and sends commands to perform a certain action. This action will then affect the surrounding environment such that we will receive new observations, and so on, as illustrated in Fig 1.1.

It is sensible to say that cognition development goes along with behavior development [9]. We benefit from the results and effects of our actions on the surrounding environment, by getting rewards for useful actions and negative feedback, in the form of wrong information or pain for example, on the actions that we better not repeat in similar situations. That being said, the use of past experiences and future predictions is not limited to the time around the current moment, but it serves the planning for future goals. In our daily activities, we subsequently plan for goals that might require many steps to be achieved. If the plan is to cook a certain meal, we will do many steps like shopping, preparing the required tools, etc., to achieve the final goal of getting the meal prepared. Thus, we move from basic sensorimotor control processes to decision making in a goal-directed manner. To this end, the sensorimotor dynamics are compacted into units, and understanding this abstraction remains one of the biggest challenge in cognitive science.

When we observe the surrounding environment, we use the collected information to map certain characteristics to the observed things or scenes, forming their "Gestalt". When we observe a cup of tea, its characteristics are mapped to its defini-

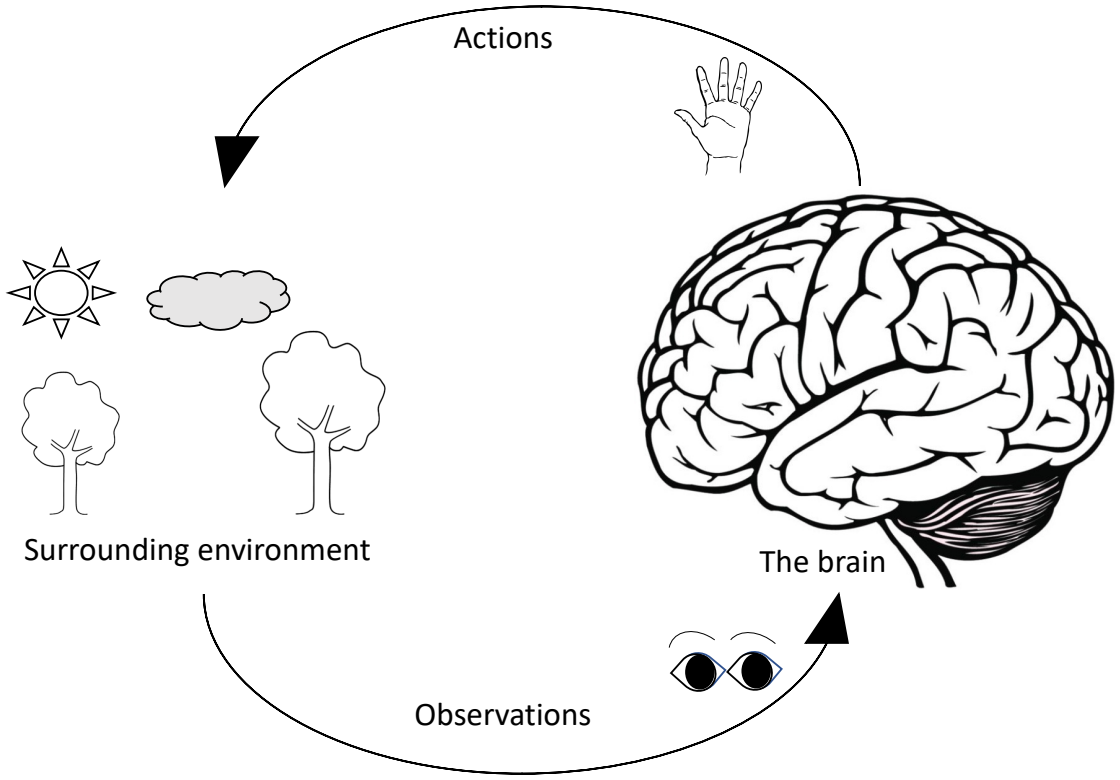


Figure 1.1: *The interaction between the brain and the surrounding environment. We receive information that could be pictures, sounds, etc, and process it to decide on what to do next. Our actions in turn affect the world around us.*

tion in the brains. Hearing the word ”Cup”, alone or within a sentence, would then trigger the representation of the cup that we built in our mind. However, when the observed ”Cup” is nothing like what we had in mind, this will result in a prediction error, which will be used to update our internal Gestalt of a cup.

Understanding human cognition also relies on biological science. This aspect includes studying the actual molecular and cellular neural units that constitute the brain, moving on to the sensorimotor neural systems, reaching the high-level cognitive processes and how to model these mathematically [9]. One useful method here is the study of the electrophysiological activity of the brain, called electroencephalography (EEG). By analyzing and filtering the signals, the activity of certain areas of the brain can be experimentally linked to certain cognitive functions. Another important tool is medical imaging of the brain. Varied types of imaging can be used, among which the functional magnetic resonance imaging (fMRI) attracted lots of attention for being a noninvasive tool to measure the brain activity and map it to the source [29].

Another important aspect that is closely related to cognitive science is language. Although learning how to speak and understand language requires collecting vocabu-

laries and learning grammars, it is proposed that the actual acquisition of language is also based on learned conceptualization and requires connecting the words to these concepts in our brain [80]. In fact, the study of language went on to be accompanied by cognitive studies, as the cognitive linguistics field appeared in the 1970's [60].

1.1.2 Common sense and awareness

One of the amazing characteristics of our mind is the ability to perform an overall comprehension and evaluation of different situations, even when no concrete situational background knowledge exists. This is known as having a common sense, which is still difficult to understand, and thus extremely difficult to simulate in artificial agents.

Even when we receive only very little sensory data, the mind is able to extract and deduce a significant amount of (hidden) information, forming internal behavior-relevant state encodings. We use these encodings to generate conclusions, decide on our next goals, and pursue them by means of inference-based planning [48] [7] [11].

Researchers aim to understand the mechanism behind the so-called efficient learning [37], i.e., learning from few examples, attempting to implement these abilities into artificial agents. This characteristic helps to avoid dangerous situations to a certain extent, even if not encountered before. Besides, it helps to make better decisions in partially novel situations.

1.2 Artificial Cognition

1.2.1 Simple neural network-based systems

The field of artificial intelligence, as the name reveals, started as a way to simulate the functions that constitute what we define as intelligence. Although we still do not have a full definition of intelligence, it can be described as "The ability to use memory, knowledge, experience, understanding, reasoning, imagination and judgement in order to solve problems and adapt to new situations" [49]. This definition gives us an insight into the aspects that scientists have been trying to artificially simulate using algorithms.

In order to build machines that can perform these tasks, it sounded reasonable to first understand how these functions are operated in the brain. The argument for this choice came from two main reasons. On the one hand, having these tasks functioning in the human brain is a great inspiration that having such complicated functions is possible. On the other hand, by studying animal cognition, we can develop new algorithms and verify them [37].

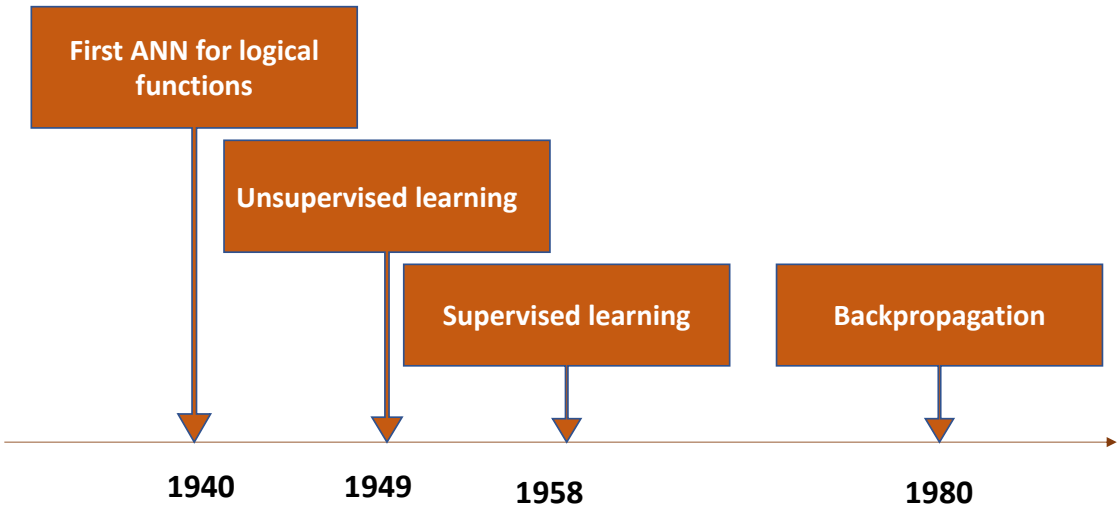


Figure 1.2: *A simple timeline marking the appearance of important algorithms in AI.*

This mission has been pursued by neuroscientists. The field of neuroscience has had many accomplishments in the last century, and it has inspired the field of artificial intelligence.

The most obvious inspiration from neuroscience are artificial neural networks. As their name reveals, these structures are composed of a set of units called neurons. The neurons are computational units that receive a certain input and applies a mathematical operation on it to produce the output. Since the first feed-forward artificial neural networks were presented in the early 1940s to perform simple logic operations, huge improvements have been proposed; some of the important steps are shown in a simple timeline in Fig. 1.2. Inspired by the proposed structure of neural computations in the brain, many layers were used in the neural networks to form deep networks. The layers serve the simulation of the extensively parallel proposed model of the peripheral sensorimotor stages of the human neural system [71] [57]. In addition, parallel layers can act on different aspects to achieve a final result by combining the individual results. Such multilayer structures are also helpful to investigate the hierarchical structure of the brain computations [23].

The other aspect in artificial intelligence highly inspired by neuroscience is the concept of using a certain reward or punishment to the performed action. This feedback signal can be used to encourage or limit a certain behavior to enhance the learning process; a term called reinforcement learning (RL) [77].

1.2.2 Incorporating more cognitive aspects

In order to improve the performance of AI techniques, the use of many cognitive functions within the AI algorithms and structures is being exploited nowadays.

Among the most important additions is the inclusion of the attention mechanism. As the name reveals, the attention mechanism means focusing on part of the input based with the highest priority at each level. Selective attention has been studied in the human visual cortex using the functional magnetic resonance imaging (fMRI), as we focus on certain parts of the pictures, the parts which include important features [51] [2]. The question of how does attention focus on what is important remains an important question in cognition.

From the episodic memory, which represents an instance-based mechanism of memory, to working memory, in which information are stored and updated using an active store, different forms of memory mechanisms have been exploited in AI structures [37].

Understanding and simulating the learning process also represented a research hotspot in AI models. By integrating deep learning with reinforcement learning, researcher have implemented the so called experience replay, i.e., storing some training data that had a positive effect on the learning process and play it offline to learn from it. In addition, similar to what we do as humans when we learn new skills without forgetting old ones, thus has been implemented in AI models under the name of continual learning. In the future, researchers are aiming to develop models that are able to generalize and transfer the experience learned from a certain task to other tasks.

1.2.3 Towards human-level AI

The recent advances in deep learning have shown enhanced performance in many tasks, sometimes even superior to that of humans [30]. The intended goals of constructed and always enhanced artificial intelligence (AI) systems converge to achieve better and better accuracy in certain tasks, like mastering a certain game [72], or diverge to propose novel overall structures.

With the ongoing development in artificial intelligence, the argument is rising of whether this development is going in the intended direction of Strong AI, the AI which resembles the abilities of the human mind [18]. The argument proposed and discussed [59] is that there is a need to expand current model-blind machine learning research to include counterfactuals, which represent alternative scenarios of what would happen if different choices were made. Counterfactual reasoning has been used in plenty of ML models to decide on the best policy in decision-making [88], but its practical integration within ML-based event cognition models is still not well investigated.

Human-level (Strong) AI can reason about the surrounding world and adapt its behavior accordingly to achieve the most benefit and avoid danger. Achieving such AI requires us to dive more deeply into the actual behavior of the human brain. Hierarchical. extensively parallel structures should be used, and the development

of event-predictive encodings from sensorimotor experiences should be investigated and implemented [13, 37].

1.3 Event Cognition

1.3.1 Quantitative and descriptive theories on event cognition

Surrounded by a lot that is happening and changing throughout time, we humans can not only act based on what we are receiving at the moment. In fact, we need to use our previous observations to learn what is the best option to consider for the current situation. This does not mean going through all the saved information, but rather using the accumulated information to develop an internal model representing the outside world. This model summarizes our experiences and it is always being updated using new observations.

Before introducing event cognition, it is necessary to distinguish between two terms often interchangeably used: event and context. An ”event” is a segment of time in a given location that an observer perceives as having a beginning and an end [84]. The ”context” is the environment in which a particular event occurs [82].

For example, driving a car is an event, driving a bus is another event. While both events represent moving from one location to another using a vehicle, the car and bus represent two different contexts. The difference here is that the car and bus have different characteristics, like speed, weights, etc. Thus, they represent two different circumstances for the event of moving or driving to happen within. In the next chapters, the information provided to the network represents contextual information, while the encodings compressed in the network represent event encodings.

The way we perceive the information and organize it has been an interesting research question since 1920’s, with many studies about the understanding and perception of events. The proposed Event Segmentation Theory (EST) [63] [86] suggests that we use sensory information to develop representations of separable events. We use this structure of events and subevents to represent the current event and also to predict the next thing to happen [21]. The practical implication of this is that we use these units to reach a final goal, starting from the current state. Thinking about reading a book in the library in another room, we will soon realize that we need to be close to the library to reach the book, and then we need to be in the other room where the library is, and then need to leave the current room, and then we need to walk, so we need to get up first. Hence, we are able to construct many intermediate steps to reach the final goals, which we may also call intermediate goals.

The intuitive principle behind context switch recognition is that we notice a difference in the received information. When a new person appears through the door, the visual and audio information we are receiving are different, this is how we re-

alize the new event of someone entering the room. The reason this shift causes us to believe a new event started is because it does not match our predictions, we were expecting to see a door, but instead we see a person passing through the door, this difference - measured mainly by prediction error - is used to update our model and to identify an event boundary. When we want to simulate this, we should look for similar signals that can inform our active internal models about the context switch.

1.3.2 Why are events important

Recognizing events helps to adopt an enhanced behavior that can result in better outcomes of our actions. Many studies from the fields of neuroscience and psychology shed light on the importance of events in our cognitive functions [62].

The ability of humans to distinguish different contexts have been previously investigated in many studies [85] [87] [69]. When the predictions do not match the actual observations, then we realize that a new event has begun. An example of a continuous set of events is a film with many scenes. Loschky and colleagues [50] reported that when a group of participants were shown selected parts of a film, they had more systematic eye movements when the seen part could be put within a larger context. In the work of Baldassano and colleagues [5], participants were shown different events, like flying from the airport or eating at a restaurant. By observing the brain activity, they showed that they had consistently different brain activity patterns when different events were going on. In fact, having more boundaries between the events positively affects the memory, as reported by Pettijohn and colleagues [61].

The development of event-predictive encodings plays an important role in the creation of versatile and beneficial goal-directed behavior. Such encodings emerge via the implementation of inductive biases, which mean the assumptions made by the learner to perform predictions based on new input. In fact, in has been suggested that the development of such encodings is one of the important aspects that make the human cognitive abilities superior to what has so far been achieved using AI techniques [13].

1.4 Artificial Event Cognition

1.4.1 Theoretical background

A main characteristic of the human mind that AI systems are still striving to reach is the ability to maintain a robust perception of the surrounding world. This robust perception is the result of a complex inference process in the brain, by which the brain tries to infer the hidden causes of the ongoing experiences in a goal-directed manner. This planning happens in a hierarchical fashion [4]. In short, our mind tends

Table 1.1: *A brief comparison between the top-down and bottom-up information flow.*

	Content	Role	Update speed
Bottom-up signals	Observations, details	Used to update the contextual encodings and thus react to unexpected stimuli	Fast, whenever a new observation is received
Top-down signals	Expectations, beliefs, contexts	Enhances the prediction when the sensory data are noisy or ambiguous	Slow, when the beliefs are updated

to process the stream of sensorimotor experiences and segment them to form generative event-predictive encodings [10]. The details involve two suggested directions of information flow. On the one hand, the prediction error from lower level sensory layers—mainly the divergence of the predictions from the actual observations—travels in a bottom-up (feedback) direction towards deeper layers. Fig. 1.3 shows an illustration of the two-way information flow inspired by the suggested hierarchical levels of events [44]. Meanwhile, event-predictive encodings flow from the deeper layers, where the planning happens towards the sensory level in a top-down (feedforward) direction, to enable better prediction [22]. A brief comparison [55] between the two directions is listed in Table 1.1. It is suggested that our brain constantly tries to predict what might happen next around us. It tries to limit its predictions within a set of predicted states, so that it avoids any surprising signals [22].

1.4.2 Implemented models for event identification

Identifying when a certain event ends and the next one starts has been experimentally studied. Examples included asking the participants to identify the points in a certain video at which they expect the current scene to end or monitoring the brain activity via functional MRI and identifying neural activity patterns indicating transitions between events [50] [5].

Radvansky, Zacks and colleagues [63] interpreted these indicators of inter-event switches, called event boundaries, as to correspond to an increase in prediction error, i.e., what actually happened is different from what was expected. Thus, it is about being able to predict what would happen next, if the reality does not match our predictions, we realize a new event has started.

Seamless transition between events relies on the effective transfer of event-encodings from deep prediction layers to sensorimotor processing layers, which requires more than a bidirectional transmission of the information. Mittal and colleagues [55] went to combine the four aspects of hierarchy, attention, bidirectionality and modularity

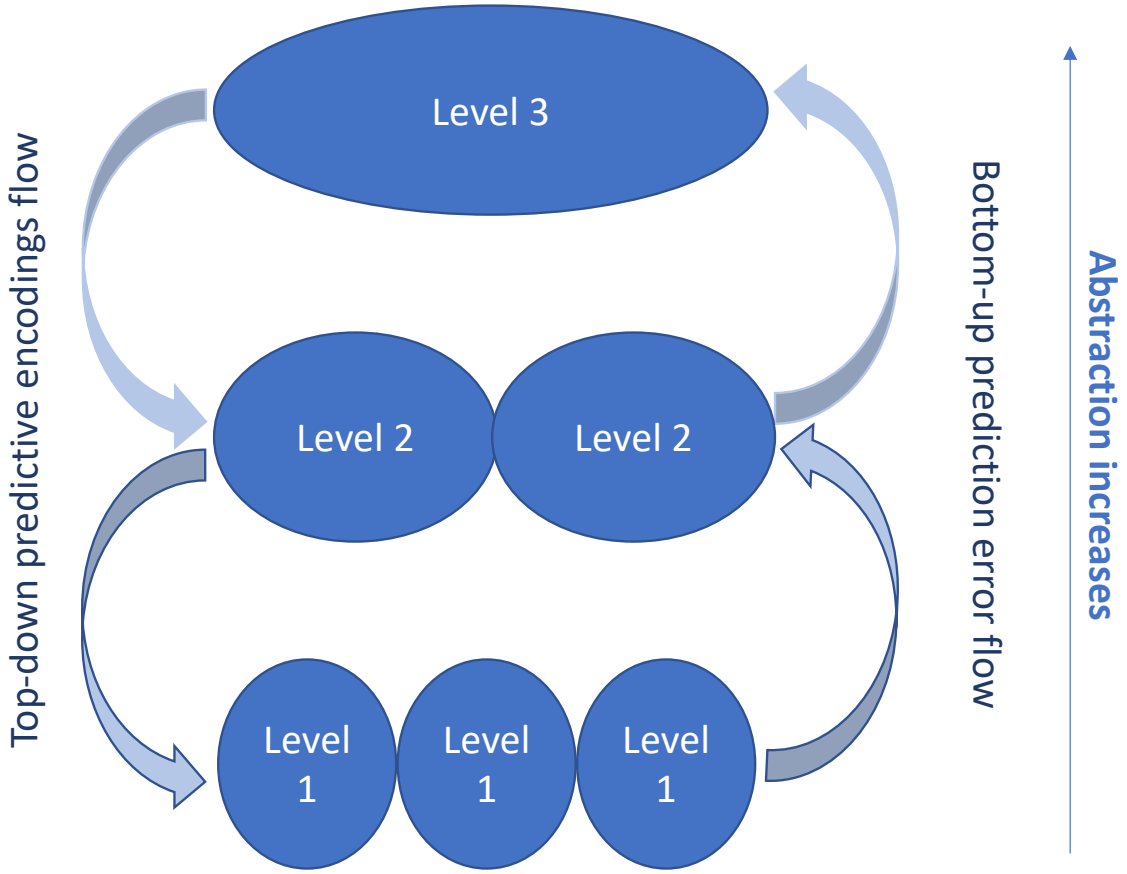


Figure 1.3: *The suggested generative hierarchical event-predictive model in the brain. The information gets more abstract as we move towards deeper (upper) levels. The prediction error flows in the bottom-up direction to update the beliefs and contexts, while event-predictive encodings travel in the top-down direction to enhance the predictions based on sensory data.*

in their model to combine top-down and bottom-up signals. When it comes to identifying useful experiences, one of the most commonly used methods in reinforcement learning is experience replay, which allows determining the experiences that were beneficial to the learning and training process, and replaying them offline to enhance the performance. On the other hand, reasoning about alternative experiences, which were not originally experienced, in the form of counterfactual reasoning has also been investigated. In reinforcement learning, Hart and Knoll [36] used counterfactual reasoning to decide if the chosen policy in an autonomous driving task is safe enough before applying it. In dialogue generation, Zhu and colleagues [88] applied counterfactual off-policy training to infer the results of alternative policies.

However, as we move from a certain event to the next one, e.g. finish reading the newspaper and start to eat something, this switch between the two different contexts happens pretty smoothly. It is as if we predict the switch to the next event, or the

event boundary, and adjust our predictions based on that. Our experience helps us to enhance our predictability of when different events can be expected to finish. Distinct event-predictive encodings of different events can help us to interpret the received observations within the current event, since they work as an inductive bias that affects our understanding of what we observe.

From the practical aspect, some models were presented to simulate the switch between different events, such that contextual information is saved in a submodule of the structure. Reynolds and colleagues [64] proposed a model to process a series of events and distinguish the boundaries between them. Event boundaries were identified as an increase in the prediction error. The model included a simple recurrent neural network along with a set of LSTM units, as shown in Fig. 1.4. They performed a set of simulations and pointed out extremely important features concerning how prediction error is related to event cognition:

- Inter-event prediction error is larger than intra-event error.
- The prediction of event-like series can be enhanced when stable context is provided.
- The events can have internal representations; these are learned and updated at event boundaries by a gating mechanism, which uses prediction error.

As we perform more tasks, the internal representations of the events play the role of developing event encodings, which then foster their further consolidation as well as the learning of further and more abstracted events and event boundaries.

The model of Reynolds et al. was later extended by a reinforcement learning (RL) agent in the work of Metcalf et al. [54], as shown in Fig. 1.5. The point was to let the RL agent control the gating mechanism, hereby getting rid of externally set thresholds. Interestingly, this work used unsupervised control of the gate, which goes in line with what was proposed in the event segmentation theory [63, 86].

Keeping updated information, which identifies or hints at the currently ongoing context, in mind while processing sensorimotor prediction tasks or other learning tasks reduces the event-specific prediction error. However, this does not mean that we have to keep updating the contextual information all the time. In fact, this will be impractical. The coordination between the learning process and requesting updated contextual information has been presented in the work of Solowjow and colleagues [74]. They proposed an event-triggered learning model, which was implemented in a control system that requests new information to update the model only when there is an actual apparent need for this update.

Although these models shed light on important aspects of event cognition and simulated some points practically, a smooth, seamless transition between the events has not yet been properly implemented. Given the proposed theories and presented

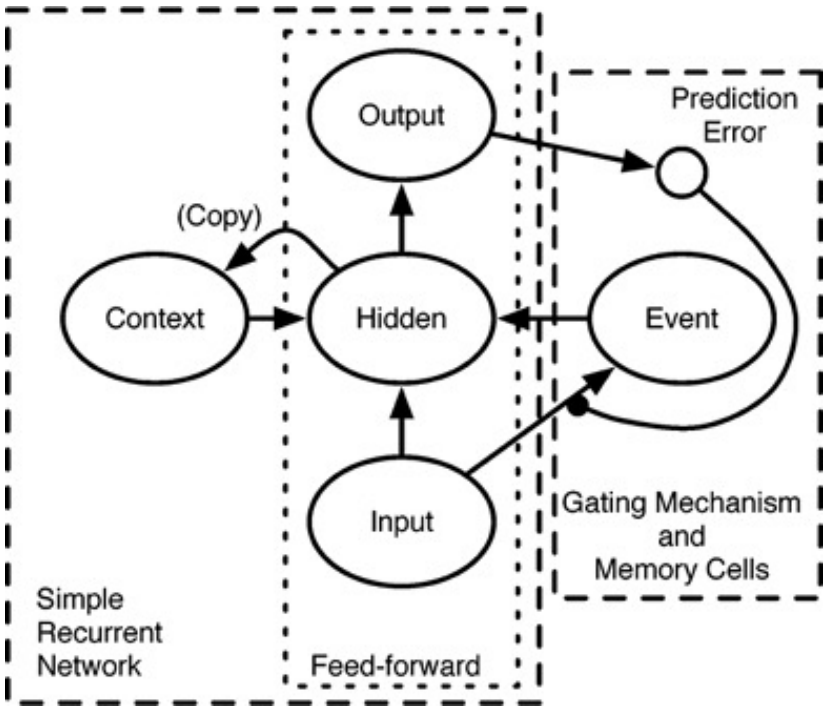


Figure 1.4: *An illustration of the event-predictive model proposed by Raynolds et al. [64]. The contextual information are used to enhance the prediction of the model.*

models on event cognition and to overcome their limitations, the next step would be to construct a hierarchical model that fosters event-predictive encodings to achieve seamless event segmentation using surprise-based gating of the information flow, which is the core of this thesis. After this introduction in Chapter 1, Chapter 2 presents a technical background to get familiar with the used algorithms and structures of machine learning. Then, Chapter 3 investigates the role of contextual information and how to infer them retrospectively when the information is not provided. It also discusses how event-predictive encodings are formed, and how to separate the connections within one structure to achieve good performance on simultaneous tasks. In Chapter 4, the gating mechanism to control the information flow is closely investigated, and different behaviors of the gate are discussed. Chapter 5 discusses 3 presented models that differ in the way surprise signals are obtained and introduces the idea of evaluating the outcomes of different decisions of the gate. Finally, Chapter 6 presents a summary of the work and discusses some open questions, of which some might represent potential future work.

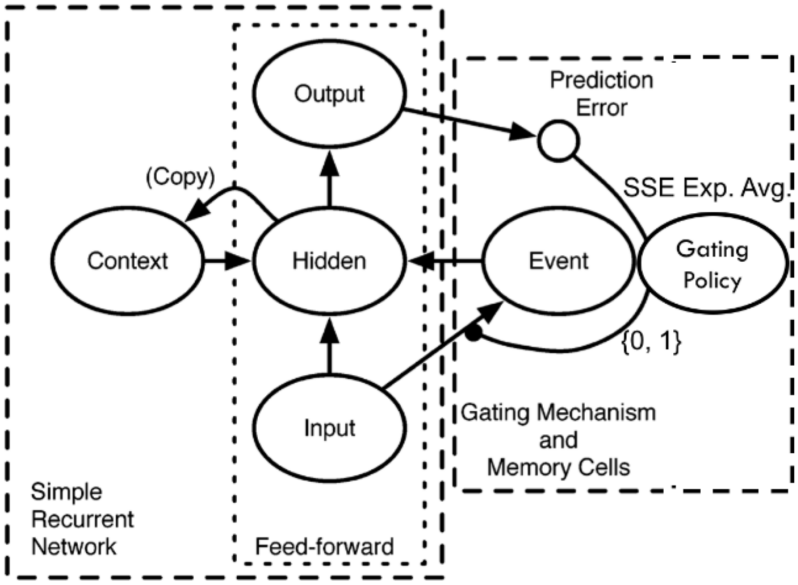


Figure 1.5: *An illustration of the extended unsupervised event-predictive model proposed by Metcalf et al. [54]. Note that the gating mechanism is now controlled by a reinforcement learning agent.*

Chapter 2

Technical background

In this chapter I will provide a summarized introduction to the machine learning structures used in this work. I will start with a brief introduction to artificial neural networks (ANNs), statistical models that process the input through multiplying it by certain weights and then applying a certain function on the result. Then, I will talk about a subtype of ANNs called recurrent neural networks (RNNs), in which the output of the previous time step is also included in the input at the current time step.

In this work, two special types of RNNs have been used, mainly Long-Short Term Memory (LSTM) and Gated Recurrent Unit (GRU) networks. Thus, this chapter provides an overview of the two types and the reasons behind the choice to use them. In the subsequent chapters, the individual event-predictive neural network architectures developed in this thesis will be presented. The order will follow the adjustments applied on the subsequent structures to answer the raised limitations.

2.1 Artificial neural networks

Just like the brain is made up of a large amount of interconnected neurons, artificial neural networks (ANNs), as the name reveals, are a set of interconnected artificial neurons. And as the brain processes the data and makes informative conclusions, ANNs receive a certain input and process it to obtain the corresponding output. They represent an important class of structures in the field of machine learning.

The main goal of ANNs is to map the input into a desired output. For example, ANNs can be used in image classification. Consider having a set of cat images and another of dog images. We can use the ANN to classify each image as to belong to the class of cat images or that of dog images. In order to understand how this might be achieved, we can have a look at what is actually happening inside an ANN.

Fig. 2.1 shows the main unit in an ANN, the artificial neuron, called the perceptron. The perceptron receives an input, multiplies it by certain weights, and then applies a function, called the activation function, to obtain the output. Usually, a constant term is added to the input to adjust the output, called the bias. If we have many of these neurons connected to each other, then we have an artificial neural network. Fig. 2.2 shows the overall schema of a simple ANN, where the neurons are grouped in an input layer, processing or hidden layer and an output layer.

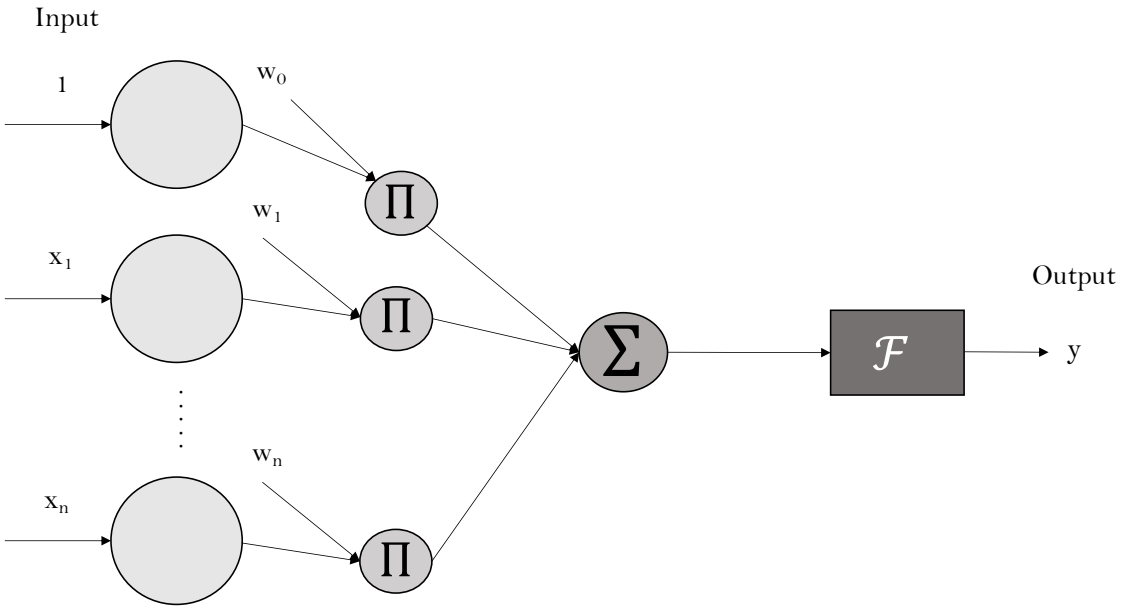


Figure 2.1: *An illustration of the perceptron. The input values (x) are multiplied by the corresponding weights (w) and summed up. Then, an activation function (F) is applied to obtain the output.*

The weights in the network are updated by comparing the output of the network with the perfect output, e.g., the predicted class of the image with the actual class

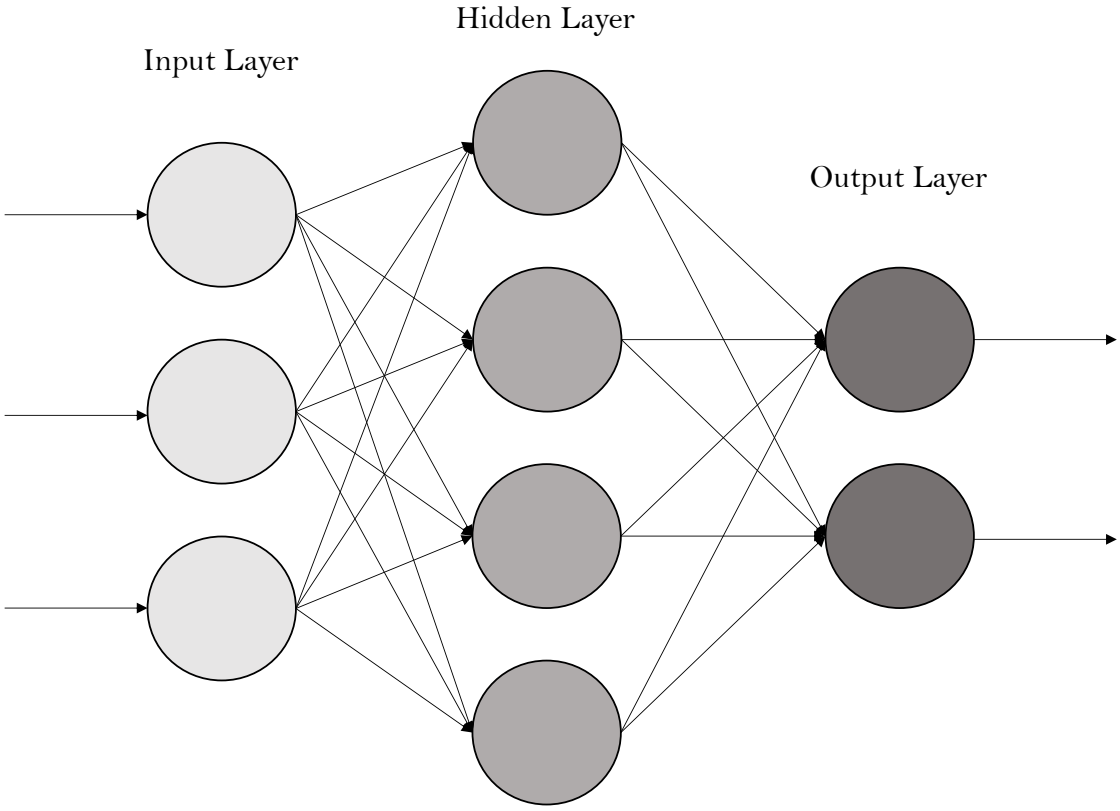


Figure 2.2: *An illustration of an artificial neural network. The neurons are grouped in three groups: input, hidden and output. Note that there can be many hidden layers.*

of the image. Then, the difference between the two values, i.e., the prediction error, is calculated. The goal here is to minimize this error as much as possible, which is the objective function of the network. One common way is to use a method called gradient descent, by calculating the gradient of the objective function and trying to find the values that minimize the function as much as possible.

The hidden layer, which does not deal with the direct input or output, has a distinct value for ANNs. As the name reveals, it contains hidden information within the ANN, which can be included as an additional input at the next time step as we will see.

2.2 Recurrent neural networks

When dealing with a series of data, like a stream of sensory data representing the movement of a vehicle, these data points are then related to each other. If we want to predict the location of the vehicle at time t+1 given where it is at time t, then it is useful to also use the information on where it was at time t-1.

Neural networks that keep the output of the previous time step to be used as

as additional input at the current time step are called recurrent neural networks (RNNs). They represent powerful structures when analyzing time series data and have achieved amazing results. Fig. 2.3 shows an example illustration of a recurrent neural network.

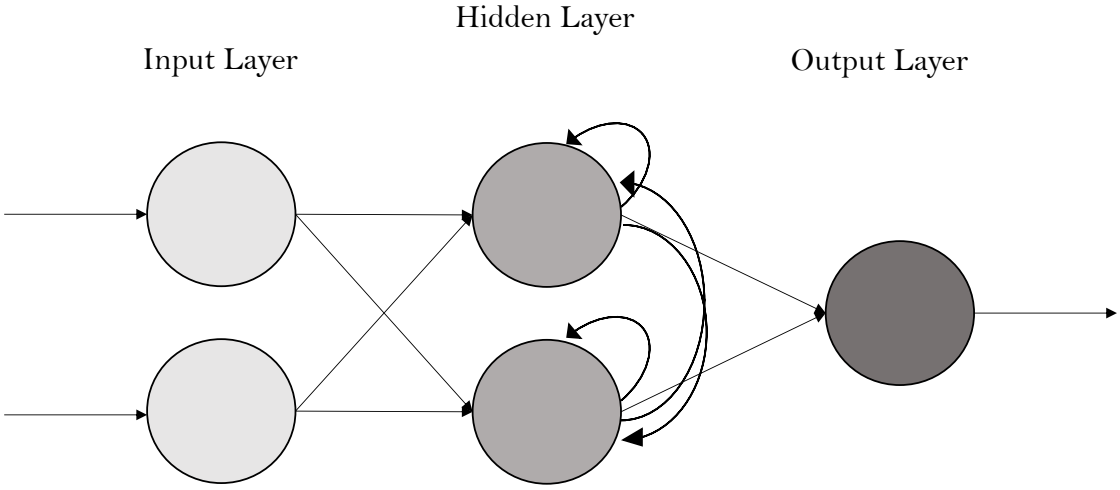


Figure 2.3: *An illustration of a recurrent artificial neural network.*

However, as the data series gets longer, it becomes hard to keep the gradient significant to cause an update. This problem is known as the vanishing gradient problem. As a solution to this problem, a new RNN structure was presented, called the long-short term memory.

2.2.1 Long-short term memory units

So far, the data flow in neural network is composed of the input, hidden and output data. The key idea in long-short term memory (LSTM) units, presented by Hochreiter and Schmidhuber in 1997 [38], is the extra line of data called the cell state. This state is updated every time step by dropping or adding data through the connections between this data line and the input at the current time step. The connections are controlled with gates, which open or close to decide whether the data can pass or not. The LSTM unit has three gates to edit and update the cell state: the forget gate, input date and update gate. Due to this structure, LSTMs are able to retain useful data and forget those that should be replaced. A simple illustration of an LSTM unit is shown in Fig. 2.4.

The idea of having gated data flow is very important when the update of the post-gate data should only happen based on certain conditions and not always or never. LSTM units have widely spread and became the state-of-the-art in RNNs for the processing of long time series data. However, the LSTM does not have control over

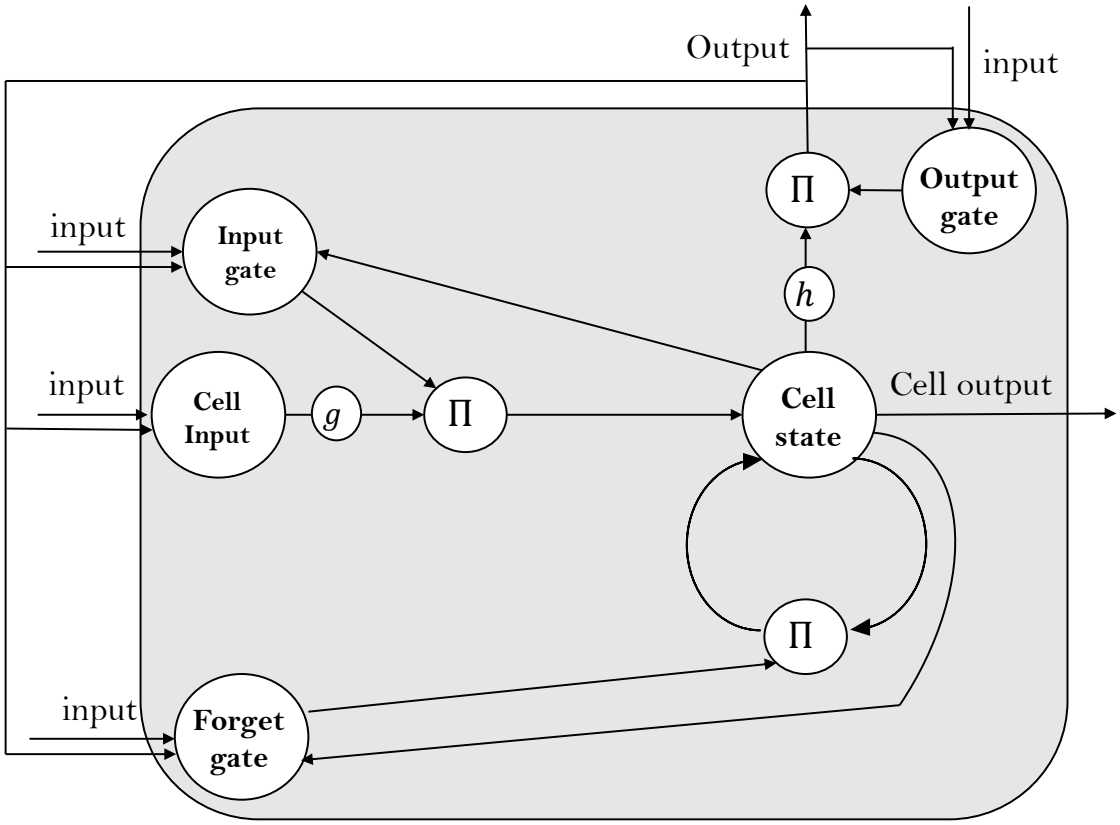


Figure 2.4: *An illustration of a long-short term memory unit.*

how much of the information from the previous time step is used. All the information from the previous time step is passed into the current time step and then updated using the input at the current time step. To add more control on the flow of old Information, Gated Recurrent Units were presented.

2.2.2 Gated recurrent units

In 2014, a new structure was proposed by Cho et al. [14] [15], called the Gated Recurrent Unit (GRU). This structure allowed direct control of the information flow from the previous time step. There are two main gates in the GRU: the update gate and the reset gate. The GRU also contains a sort of cell state, which represents the information saved within the GRU unit from previous time steps. At every time step, the reset gate decides how much of the old information will be added to the input at the current time step. Then, according to the update gate, the cell state is updated as to contain some of the new information and some of the old cell state information. A simple illustration of a GRU is shown in Fig. 4.4.

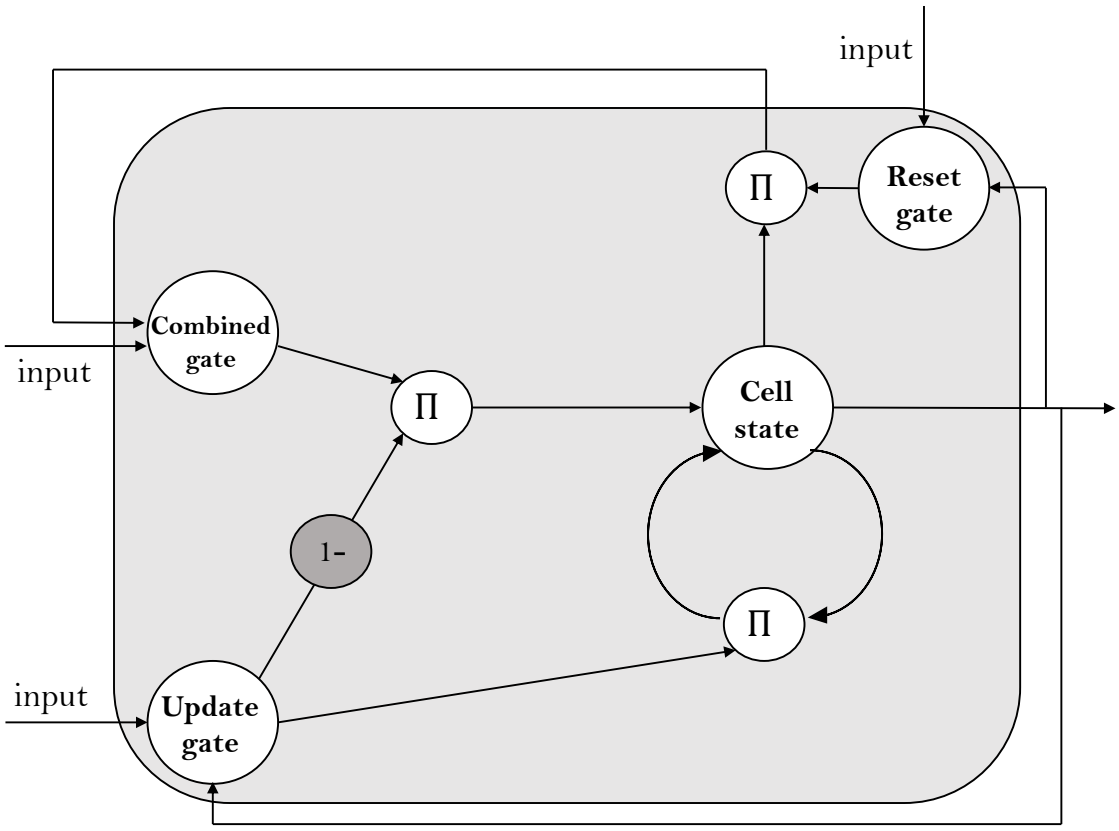


Figure 2.5: *An illustration of a gated recurrent unit.*

2.3 Conclusion

ANN structures that have shown superiority in processing time series data can be very useful to construct models that can simulate how the brain processes the stream of sensorimotor information. The idea of using previous information to decide on how to process the current data is similar to how the brain needs to have some contextual information to decide on the current task. For example, to continue the sentence ”thus, I can speak .....”, we need to know which information was mentioned before. If the previous sentence was: ”I was born in Germany”, then we can predict that the sentence will continue as ”thus, I can speak German”. This information, once entered, is saved within the cell state. Then, it is update according to the gates. Now, how to control this update such that it is only performed when an update of this contextual information is needed? This is one aspect of the work performed in this thesis.

Chapter 3

Retrospective event inference using artificial neural networks

When we decide to perform a certain action, like lifting a glass of water, we usually have a reason for doing this. The overall context might be the desire to drink water, and we might perform several tasks while keeping this context in mind, like getting up and walking towards the place of the glass and filling it with water, etc. Hence, our actions are influenced by the larger contexts they are being performed within.

In this chapter I will talk about the role of including information that indicate the current context, within which the tasks are being performed. I will describe the structure used to investigate the role of contextual information, and the process of inferring these information in case they are not provided to the system.

Then, I will talk about the actual representation that encodes these contexts, which helps to predict what might happen next given the current context. These representations are hence called predictive-encodings, and here we will talk about their development and role in event-predictive processing.

Finally, I will describe the trial to separate the tasks of inferring the contextual information and the other predictive task that the system might be performing through modularizing the structure. I will show the effects of this modularization and provide some insights into the further usage of such modularized structure [1].

[1] This chapter is based on our published papers:
Butz, M.V., Bilkey, D., Humaidan, D., Knott, A., Otte, S. (2019) Learning, planning, and control in a monolithic neural event inference architecture. Neural Networks 117, 135–144
Butz, M.V., Menge, T., Humaidan, D., Otte, S. (2019) Inferring event-predictive goal- directed object manipulations in REPRISE. Artificial Neural Networks and Machine Learning – ICANN 2019 (11727), 639–653
Most of the figures and tables are taken from these papers.

3.1 Context Inference

In our daily life, we are always analyzing the observations we make on the surrounding environment. As we perform certain actions, we usually have a goal in mind we are working to achieve. We can head to the kitchen with the goal of preparing a sandwich. Then, all the steps that we do would be directed towards making that sandwich. Thus, our interpretation of the actions we do is influenced by the overall goal we have. While opening the fridge, we can take out different things, but we take out cheese because we know we need it for the sandwich. This interpretation of the actions in light of the contextual information leads to achieve our goals more efficiently. To analyze the contextual information accompanying the execution of a mission to reach a certain target, we have investigated a neural network-based system that in addition to the sensorimotor information it receives and processes, also includes contextual information as an input, which is used while a predictive task is being performed.

3.1.1 REPRISE

The REtrospective and PRospective Inference SchEme (REPRISE) was proposed by [12] to infer from previous experiences and also probe into the future. This structure is a long short-term memory (LSTM)-based temporal predictive sensorimotor forward model [38], an illustration is shown in Fig. 3.1

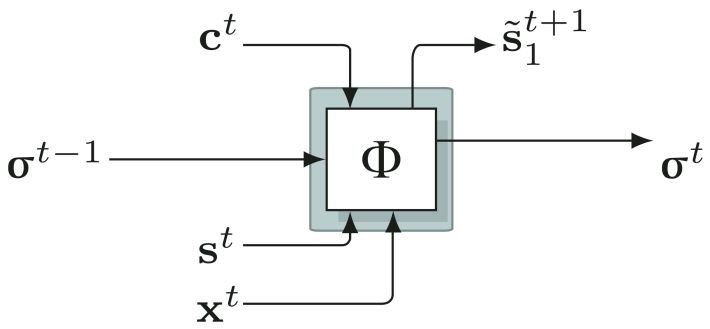


Figure 3.1: *An illustration of the investigated REPRISE. The structure uses the sensory (s^t) and motor (x^t) information it receives to predict the next state of the active system (σ^t).*

In this recurrent neural network structure, the performed task includes a group

of flying vehicles with different characteristics, thus different behaviors, trying to reach a certain target. The structure tries to determine the best trajectory to reach the target, given the type of the vehicle. There are three vehicles, representing three events, iterated at certain time intervals: $\Phi 1$ is a multi-copter-like vehicle, called the rocket, $\Phi 2$ is a static omnidirectional vehicle, called the stepper and $\Phi 3$ is a dynamical, omnidirectional gliding vehicle, called the glider. The properties of these vehicles are listed in Table 3.1.

Table 3.1: *Different systems used in REPRISE evaluation*

Vehicle	Number of motors	Gravity	Inertia
Rocket	2	Yes	Yes
Stepper	4	Yes	Yes
Glider	4	Yes	No

The structure receives the following input: actual sensory information (the location of the vehicle), an imagined sequence of motor commands and information denoting the currently active vehicle. The network can predict the sensory information at the next time step that would result from executing the motor commands. To do this, the forward model considers the current system to be dynamic, with time-dependent states, such that the next system state is determined based on the previous state and the mapping Φ according to the following equation:

$$(s^t, \sigma^t, x^t) \xrightarrow{\Phi} (s^{t+1}, \sigma^{t+1}) \tag{3.1}$$

where x represents the system control, s denotes the perceivable state, and σ represents the hidden state, given that a partially observable Markov decision process is assumed.

In order to reach the target, the motor commands, which are random so far, should be adapted according to the vehicle in use. By performing prospective inference into the future, REPRISE uses the resulting prediction error to update the motor commands, such that this error can be reduced. For this, it uses the backpropagation through time (BPTT) algorithm.

This task becomes more difficult when the identity of the current vehicle is not provided, and REPRISE needs then to infer this information as well, this time retrospectively using information from the previous steps. As a result, REPRISE performs two-fold inference: prospectively to infer the motor commands, and retrospectively to infer the current vehicle in use.

3.1.2 Goal location reaching behavior

We started by providing the network with the following:

1. Sensory information, represented in the Cartesian coordinates (x,y) of the vehicle.

2. Motor commands, which are represented by a randomly generated vector of 4 values.

3. Contextual information, a three-bit one-hot vector representing the currently used vehicle.

First, a random vehicle is picked to be used, and the network is provided with the contextual information, current location of the vehicle and random motor commands. During training, the network performs prospective motor active inference to update the random motor commands with suitable commands to reach the goal. At every 150 time steps, a new goal location is randomly generated and the controlled vehicle is switched. We trained the network for a total of 2000 epochs, each with 1000 time steps. We tried to train the network with and without providing the contextual information, and using different learning rates. Then, we tested the network. The evaluation of the method showed that the REPRISE architecture can control the vehicle to reach the goal location by executing the motor commands updated using active motor inference. Besides, it achieved the smallest average distance to the goal location when the contextual information was provided, as shown in Table 3.2.

Table 3.2: *Average accumulated distance to target in REPRISE using different learning rates*

Average	η_σ=0	η_σ=1e-4	η_σ=.001	η_σ=.01	η_σ=.1
$\vec{c}$ set	0.061	0.060	0.060	0.061	0.109
η_c=1e-4	0.157	0.139	0.109		
η_c=.001	0.106	0.100	0.079	0.082	
η_c=.01	0.090		0.072	0.077	0.127
η_c=.1	0.092			0.077	0.115

3.1.3 Developing the contextual encodings

When the information that determines the currently active vehicle is not provided to REPRISE, it needs to infer this information from the behavior produced by the vehicle. In order to have a closer look on the inferred contextual values, we have

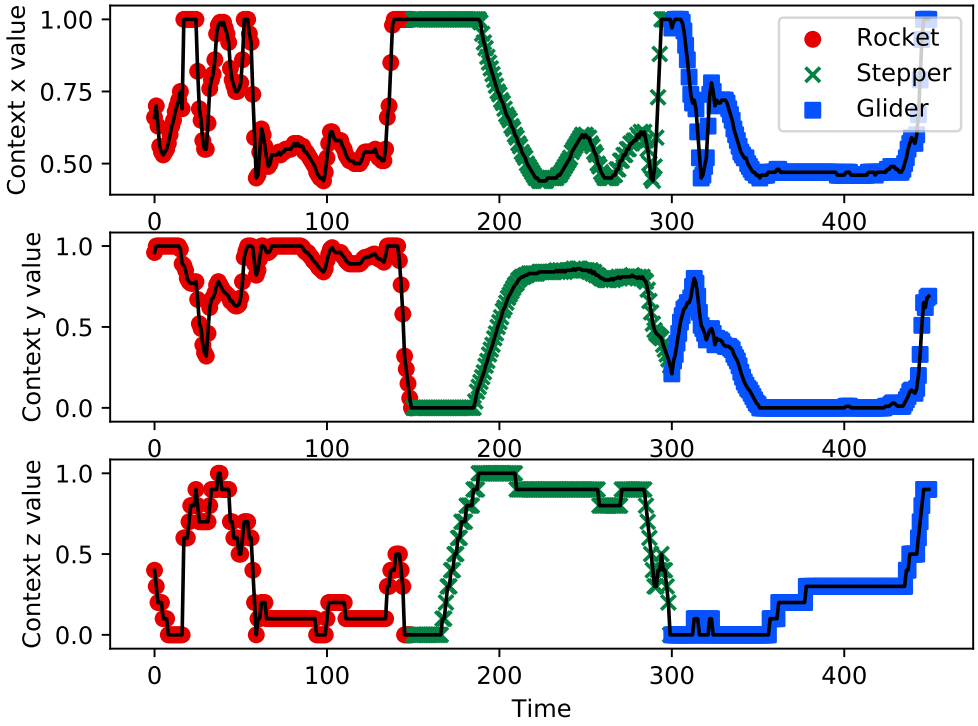


Figure 3.2: *The development of three context guess neural values while controlling the three vehicles, switching every 150 steps. The context is adapted via retrospective inference. While we can see good separation of the context guesses for the three vehicles, we can observe some uncertainty in the estimates.*

recorded the inferred values that represent the current vehicle and visualized them, as shown in Fig. 3.2

It can be observed that different clusters can indeed be distinguished, representing the different vehicles (events). One can see that since the stepper and glider both have 4 functioning motors, their clusters are closer to each other, and even overlapping. The rocket, on the other hand, is more distinguishable by its two motors; thus, its representing cluster is more clearly separated from the others. As shown in Fig 3.3, the development of the three contextual encodings takes some time. In the beginning of the event, the network still cannot decide which event is going on. Then, based on the error signals backpropagated through the network and the retrospectively-performed inference procedure, the values tend to be within a small range. The final values before switching to the next context are the values shown in Fig. 3.4

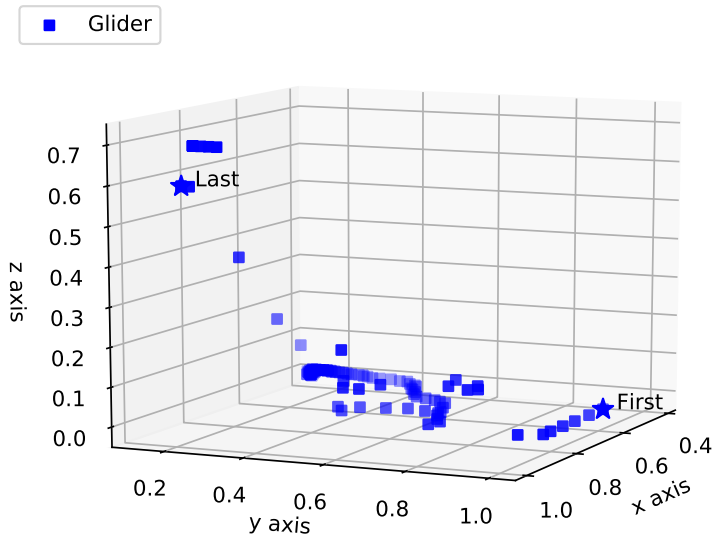


Figure 3.3: *The development of the context guess representing the glider vehicle. The star symbol with "First" indicates the start from the initial values, then the values get close to each other within the same cluster. Finally, the values drift towards the values of the next context, where the last value is located at the star symbol with "Last".*

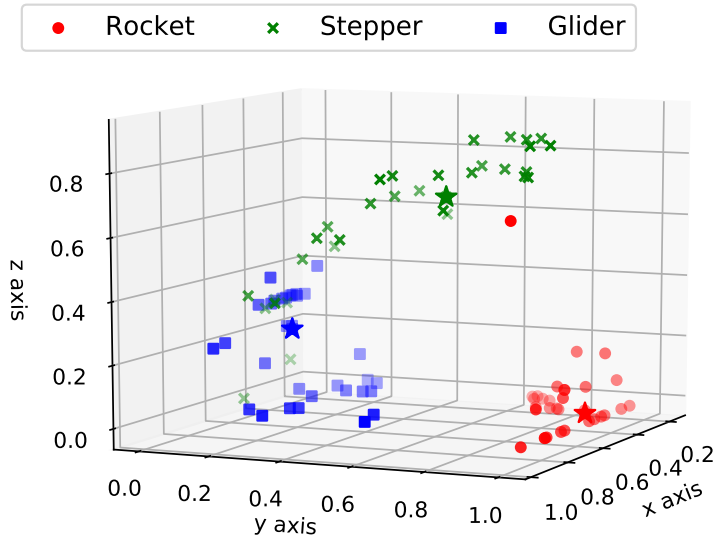


Figure 3.4: *Context guess values plotted when the vehicle reaches the goal location. The center of the cluster is marked by the star symbol.*

In order to better visualize the context values, we used the Principle Component Analysis (PCA) algorithm to reduce the dimensionality and plotted the context values in 2D space. The results are shown in Fig. 3.5

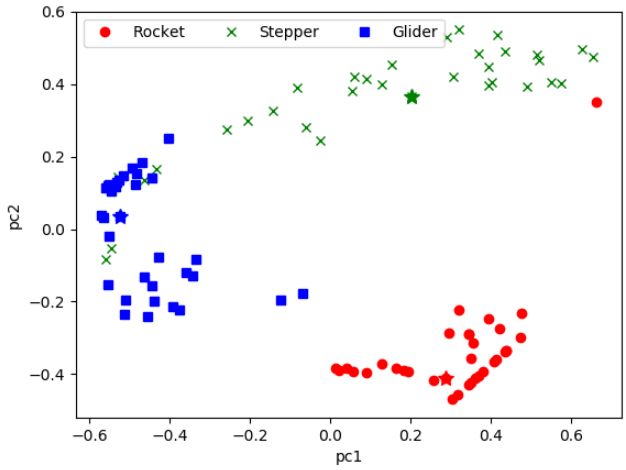


Figure 3.5: *2D plot of the context guess values when the vehicle reaches the goal location, obtained by applying the PCA algorithm on the 3D values. The center of the cluster is marked by the star symbol.*

3.1.4 Prediction error, Euclidean distance

At every time step, the vehicle moves according to the motor command trying to get closer to the target. Two values change along: the distance between the vehicle and the target and the prediction error between the vehicle location predicted by the network and the actual location after executing the command.

In order to inspect the dynamics around the event boundaries, we recorded the changes in the Euclidean distance between the vehicle and target and the prediction error during the testing phase. In addition, we plotted the corresponding step-wise change in the context guess, as shown in Fig. 3.6.

We can observe that after a new target is set, it takes some time for the distance to start decreasing, during which the prediction error also fluctuates. This is because at every new randomly generated target, some time steps are needed for the motor commands to be appropriately adapted to drive the vehicle towards the target.

Besides and more importantly, a vehicle switch happens at the same time as the target switch (every 150 time steps here). Since the vehicles have different characteristics, representing different contexts, time is needed after the switch to adapt the network dynamics to the new vehicle. This can also be seen in the visualization of the context guess development in the bottom part of Fig. 3.6, such that the context guess gradually changes after the switch until it hovers within a cluster of values.

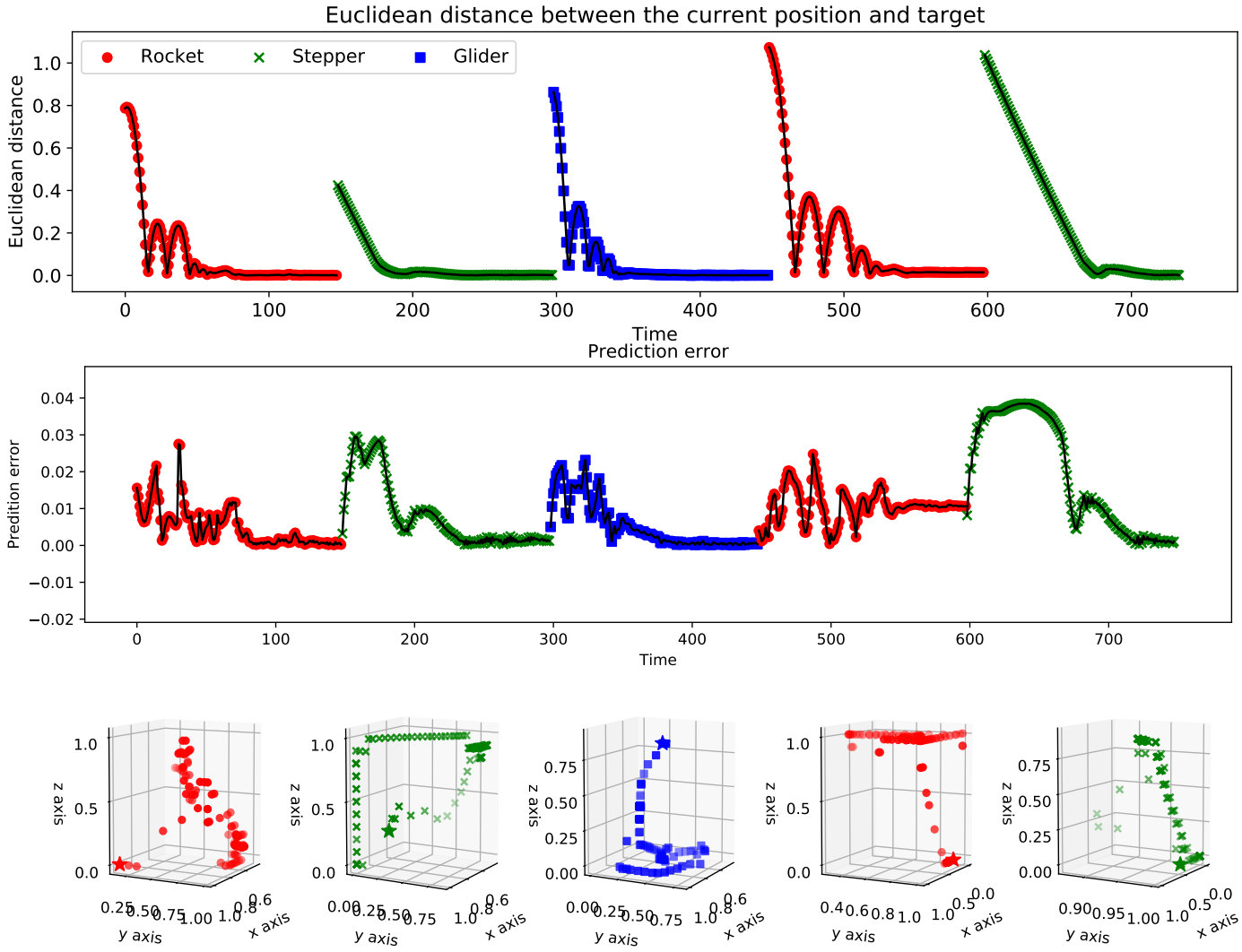


Figure 3.6: *Top: the Euclidean distance between the vehicle and the target at each time step. Middle: the prediction error of the network at each time step. Bottom: the changes in the context guess at every time step.*

3.1.5 Asynchronous vehicle and target switch

So far, the switch of the vehicle and target happened simultaneously. However, we wanted to explore how the prediction error would change if only one of them was changed at a time.

We performed an experiment in which the vehicle and target were asynchronously switched (vehicle switch every 100 time steps, target switch every 150 time steps). Then, we plotted the prediction error during the testing phase of the trained network, as shown in Fig. 3.7.

It can be observed that we can still see increases in the prediction error at the vehicle switches, although the increases are less than when this switch was accompanied by a target switch.

It can also be seen that the amount of these sudden increases depends on whether the vehicle switch happens when the prediction error starts to stabilize (ex. time point 200) or while the prediction error is still largely fluctuating (ex. time point 500).

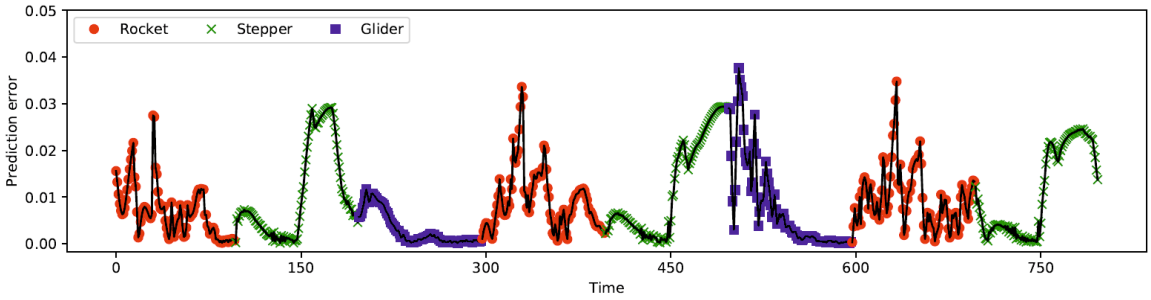


Figure 3.7: *The prediction error of the network when the vehicle and the goal are being asynchronously switched (vehicle every $V = 100$ time steps, goal every $G = 150$ time steps). We can still observe sudden increases in the prediction error at vehicles switches, although these changes are better observed when the goal and vehicle are switched simultaneously.*

3.2 Gradient separation and modularization

The actively inferred motor commands in REPRISE led to an enhanced performance in reaching the target, achieving the best behavior when the contextual information was provided. However, in the goal-directed planning in the brain, multitasking can happen. In this case, many tasks are being processed at the same time, and organizing these tasks within the corresponding events is important to perform them efficiently. Besides, it is important to still be able to use the separate errors result-

ing from different tasks to update the corresponding believes and event-predictive encodings.

In the next experiment, we investigated the behavior of REPRISE when there are competing goals to be achieved. In addition to the goal location that the vehicle needs to reach, we added a target object that the vehicle needs to pick up and attach before heading to the goal location. An example is shown in Fig. 3.8.

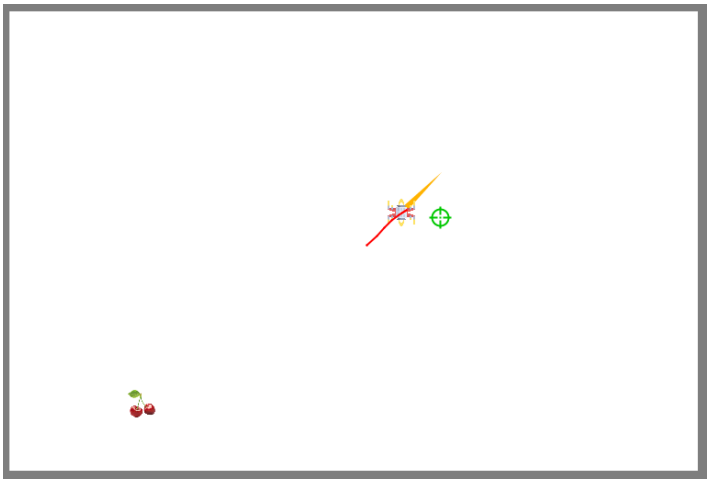


Figure 3.8: *The updated experiment with two goals: picking up the object (the cherries) and flying to the target location (the green sign).*

In this case, the network receives the following input:

1. Sensory information, represented in the Cartesian coordinates (x,y) of the vehicle and a sensor for the attachment of the object [0,1].

2. Motor commands, which are represented by a randomly generated vector of 4 values representing the 4 thrusts and one signal to attached the target [0,1].

3. Contextual information, a three-bit one-hot vector representing the currently used vehicle.

We trained the network in three different modes:

1. Free Flight: the vehicle moves in a random fashion across the environment.

2. Seek: the vehicle moves towards the object to be transported and attach it when the distance between the vehicle and the object is below a predefined threshold.

3. Transport: the vehicle moves randomly, like in the free flight mode, but with the object attached.

The switch from one mode to another happened when attaching or detaching the object, or every 50 time steps.

We first started by evaluating the ability of REPRISE to reach the goal location, sequentially with a learning rate of $1e$-3 and $1e$-4 and switching the vehicle and target every 150 steps. Then, we evaluated the ability of the structure to reach the object location and attach it. Finally, the two tasks were combined to evaluated the ability of the structure to attach the object and transport it to the goal location.

The results showed that the vehicle could reach the goal location while keeping an average Euclidean distance of 0.042 units within the last third of the overall running period. However, when the task was to reach the target object, although the object was finally reached, the average Euclidean distance between the vehicle and the object was 0.535. This increased distance from the object made it hard to pick it up and transport it. The results are shown in Fig. 3.9 as the average over seven trained and evaluated networks.

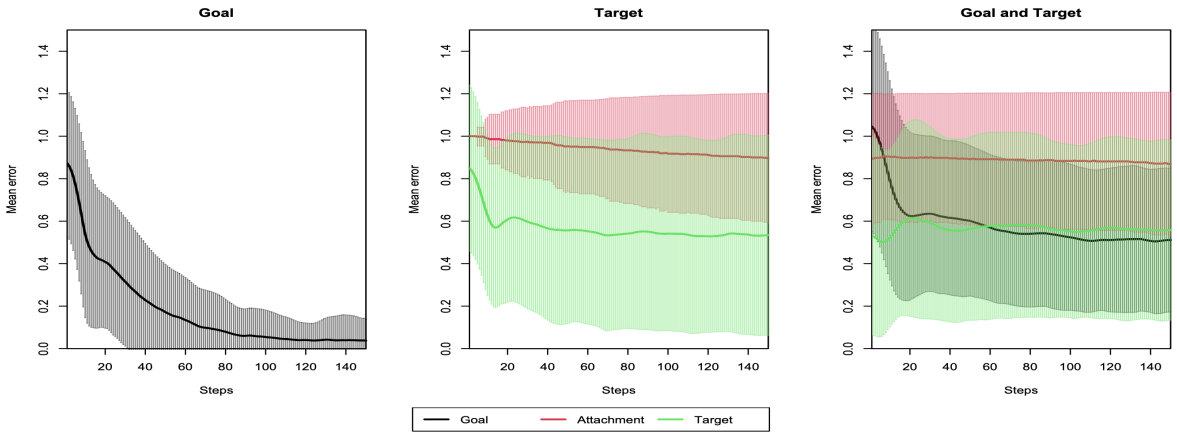


Figure 3.9: *The results of evaluating the network on the three tasks of reaching the goal location (left), reaching the target object location (middle) and transporting the object to the goal location (right). The black color denotes the Euclidean distance to the goal location, the green color indicates the Euclidean distance to the target object, and the red color denotes the error in the current attachment signal. Each graph also includes the respective standard deviation.*

Although the two tasks were similar, the second one included attaching the object in addition to reaching it. In order to check if the discrepancy is a result of the additional error cased by the object attachment, we ran the object reaching experiment without performing control inference on the attachment part. The results showed that the performance was significantly improved, as shown in Fig. 3.10. The table shows that the goal location could be reached with an average Euclidean distance of 0.080 units during the last 50 time steps of the trial. On the other hand, the target object could be reached with the average distance of 0.146 units during the last 50

time steps.

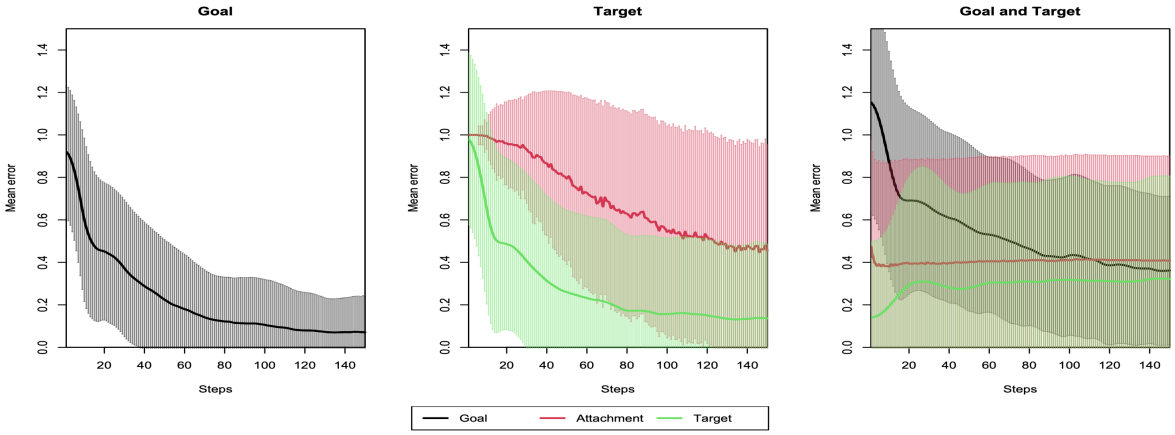


Figure 3.10: *The evaluation results of the network in the three modes without performing control inference on the object attachment. The color legend is similar to that in Fig. 3.9.*

The observed results could be explained by two points. On the one hand, the training statistics in the scenario of object attachment can be biased. This is because in order to be able to attach the object, the vehicle needs to be close enough to the object, so the attachment task is affecting the flying behavior. On the other hand, having more than one task on which active control inference is applied means that many gradients would be competing and interfering with each other, resulting in performance drop.

In order to handle these issues, we first considered to avoid using the information of the vehicle location as the sensory input to the network. This is because when the vehicle is often close to the object, the absolute locations will be non-uniformly distributed. Hence, we used the velocity of the vehicle instead of the location, and changed the desired goal reaching objective into a velocity objective. The results, illustrated in Fig. 3.11, show a significant performance improvement. The target location could be reached with an average Euclidean distance of 0.0006 units during the last 50 time steps, while the goal object could be reached with an average Euclidean distance of 0.045 units within the last 50 time steps.

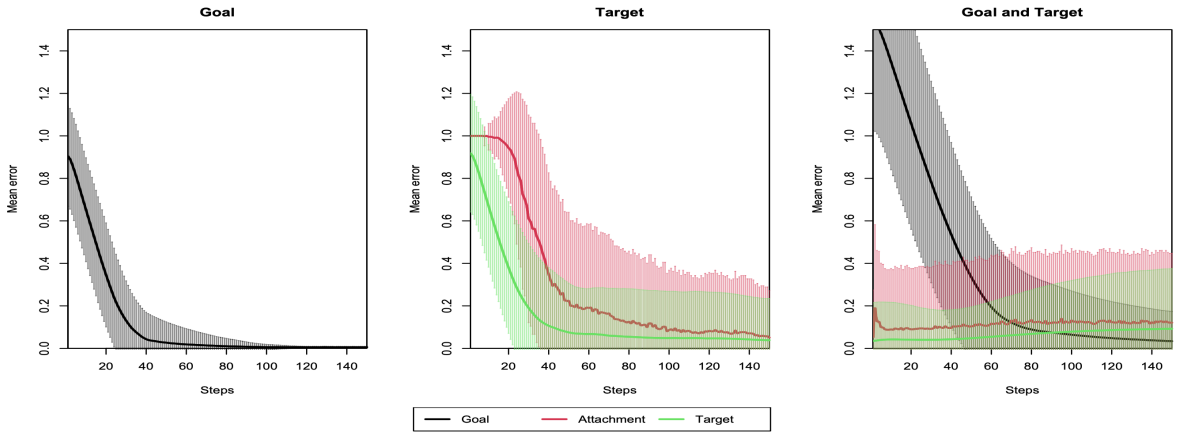


Figure 3.11: *The evaluation results of the network in the three modes when the velocity-based encoding is used. The color legend is similar to that in Fig. 3.9.*

However, during transportation of the object to the target location, there still exists the problem of target objects being dropped. Hence, although the objectives of reaching the location or objects were achieved more efficiently with switching to the velocity objective, the attached object was frequently lost. We speculate that this behavior might be driven back to the problem of interferring gradients.

In order to overcome this problem, we updated the used REPRISE structure into a modularized scheme. In this modularized architecture, one sub-RNN is used for the task of the sensory prediction, and another sub-RNN is used for the task of predicting the attachment status. The original and modularized architectures are shown in Fig. 3.12.

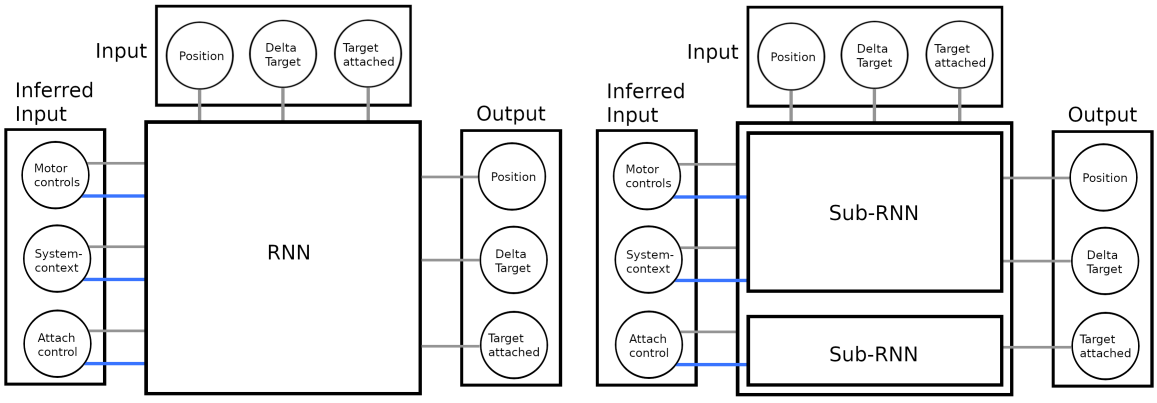


Figure 3.12: *The default (left) and modularized (right) RNN. Grey connections indicate forward-passes, while blue connections denote inference. In the modularized RNN, we connect the output and inferred input to one of the sub-RNNs each; thus, the back-propagated error signals are modularized.*

We performed two experiments using the modularized network. First, the velocity-based encoding was used. The results show that the average Euclidean distance when reaching the goal location was 0.018, and that when reaching the target object was 0.028 during the last 50 time steps. During the objective of reaching the location while transporting the object, the average Euclidean distance was 0.036, and the object was attached in 98% of the cases. The results are shown in Fig. 3.13.

In the second experiment, we used the spatial-encoding. The goal location could be reached with an average Euclidean distance of 0.014 units, and the target object could be reached with an average Euclidean distance of 0.037 units during the last 50 steps. The average Euclidean distance when reaching the goal location while transporting the object was 0.086 units. The results are shown in Fig. 3.14.

In summary, the experimental results show that this constellation succeeds in preventing the gradient interference problem, not only when the velocity is used as the sensory input to the network, but also when the location is used.

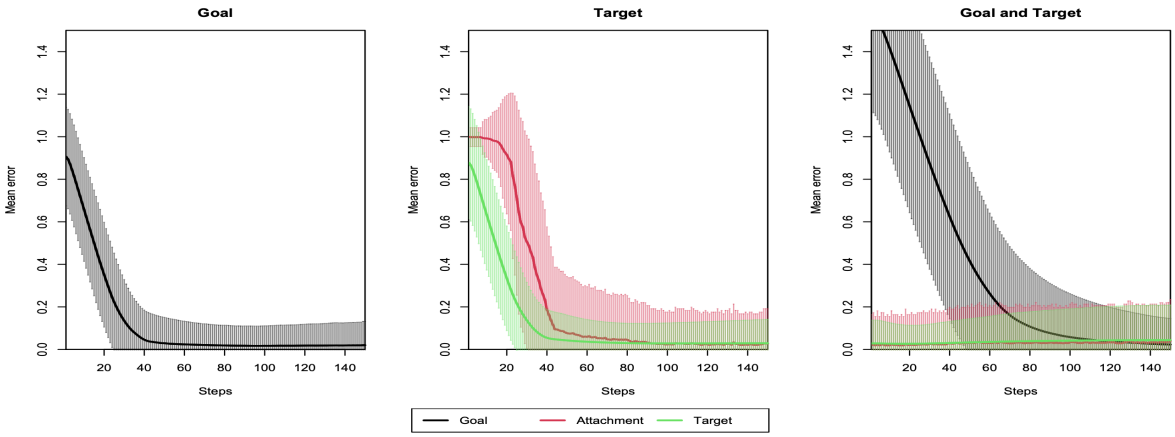


Figure 3.13: *Evaluation results of the modularized RNN architecture when the velocity-based encoding is used. The color legend is similar to Fig. 3.12.*

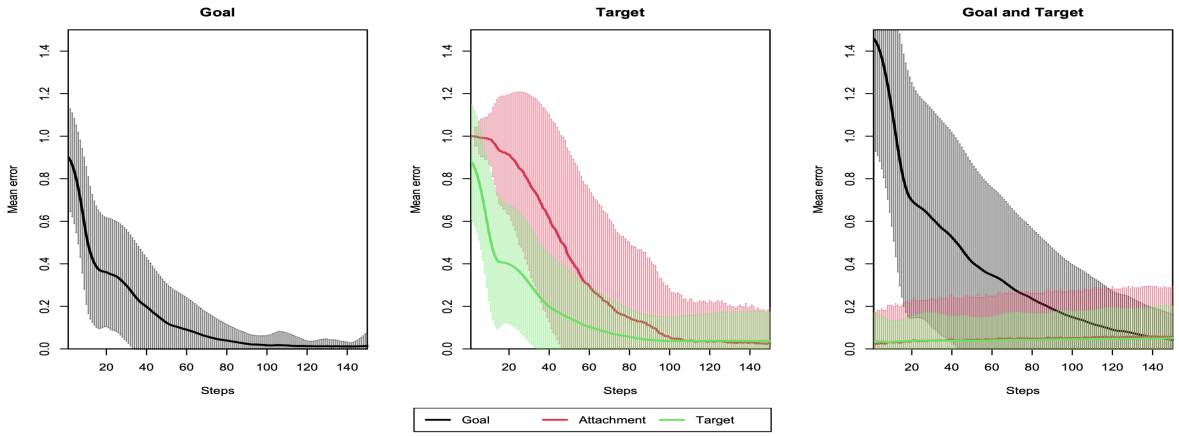


Figure 3.14: *Evaluation results of the modularized RNN architecture when the spatial-based encoding is used. The color legend is similar to Fig. 3.8.*

3.3 Conclusion

So far we found that events stay stable over time, as REPRISE managed to infer context values that form distinguishable clusters. However, there seems to be some issues that need to be handled. Although the context values lie within distinguishable clusters, the different context clusters overlapped in some places, and the values were not stable. Hence, we need to achieve a better segmentation of the events. In order to tackle these issues, we investigated the segmentation of events using RNNs, while maintaining a stable event representation within the same event. In the next section, we show how the edits we performed helped the structure to segment the events by adding inductive bias. Through the different experiments, we show how our proposed generative structure can develop hidden states that encode events.

Chapter 4

Providing event switches

It has long been suggested that the switches between subsequent events, called event boundaries, are signaled by an increase in the prediction error [63]. This increase is caused by predicting what would happen next within the current context, while the context has already changed. Thus, the next thing to happen could be correctly predicted only if we start to predict within the new context. If we try to plot the values representing how surprised the model would be by what is happening with respect to time, these event boundaries will be represented by spikes in the surprise signal. On the other hand, no such spikes would be observed as long as the same event is going on.

We constructed a model that can develop event-predictive encodings based on provided signals that indicate switches between events; these encodings are stable within the same event. Evaluating the model showed that the use of such mechanism leads to a lower prediction error, both due to interpreting the sensorimotor changes in light of the ongoing event, and having the encodings of these events stable as long as we are still within the same event [1].

[1] This chapter is based on our published paper:
Humaidan, D., Otte, S., Butz, M. V. (2020). Fostering event compression using gated surprise. In I. Farkas, P. Masulli, & S. Wermter (Eds.), International conference on artificial neural networks – icann 2020 (pp. 155–167). Springer.
Most of the figures and tables are taken from this paper.

4.1 Surprise signals in event-predictive models

Despite the important role of surprise signals in identifying event transitions, only a few event-predictive models implemented surprise signals. Surprise signals were indirectly used in form of unexpected prediction error to predict event transitions in the work of Reynolds and colleagues [65]. In this work, a feed forward network was combined with a recurrent neural network module, memory cells and a gating mechanism. An RL agent was later added to this model to control the gating mechanism with a learned policy [54]. Furthermore, reservoir computing was also used to successfully segment the information stream into understandable units [1]. More recently, the SUBMODES architecture [33] used surprise signals to detect switches between different behavioural primitives. In this model, surprise signals represented unexpected prediction error signals. The model presented in this chapter uses surprise signals to control the flow of information though a top-down gate in a hierarchical multi-level structure.

4.2 Gated surprise model

In order to simulate the event processing procedure and the changes in the prediction error within the same event and between different events, we considered extending the single-level network used in the context inference experiment (see Chapter 3), into a multi-level hierarchical structure. Hence, we included two layers: one responsible for generating the contextual encodings, and another responsible for performing the predictive task using the contextual information provided by the contextual layer. Fig.4.2 shows the extended structure compared with the single-level structure in Fig.4.1.

However, there was still one element missing, which is a way to organise the flow of information between the two layers. To add such element, we further extended the structure by adding a middle gating layer.

The final event prediction structure is then composed of three layers. The first layer is a deep neural network layer, implemented using an LSTM network (LSTM-context = LSTMc); this layer is responsible of generating the context value that will then be fed into the middle layer, the gated recurrent unit (GRU)-like layer. At the middle layer, it is decided whether the old context, which is saved in the recurrent hidden information in the layer, is to be forwarded to the third layer, the sensorimotor layer (LSTMfunction = LSTMf), or if the new context, received from the context layer, is to be used instead. The chosen context will last until a new context value representing a new event is provided by the context layer LSTMc. The sensorimotor layer performs a prediction task of the next sensory state of the system in use.

The decision about the current context is taken at the GRU layer based on a

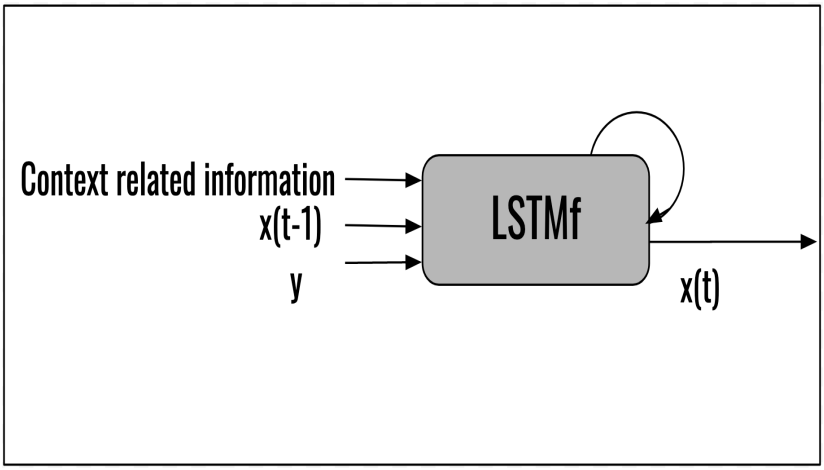


Figure 4.1: *A single-layer network processing both contextual and sensorimotor information.*

signal that determines the end of one event and the beginning of another one. When a new event begins, the sensorimotor layer that was predicting the sensory state of the system will now start to produce erroneous predictions as the system has been switched. This will lead to an increase in the prediction error, and this error can be used to generate the event boundary signal, i.e., the Surprise signal, or the predictability of the current context. If the predictability is large, then the network realizes that the same event is still going on. Once the predictability decreases, this indicates a switch in the context and a new event beginning. This will open the switch at the GRU layer allowing the context prediction from the context layer to be forwarded to the sensorimotor layer and saved in the GRU layer to be used until a new event begins.

We have also added the option to include input preprocessing layers to each of LSTMc and LSTMf. The structure is shown in Fig. 4.3.

The switch GRU The used GRU structure was edited to act as a switch to decide when to use the already saved context from the previous time step, and when to let the new context generated by LSTMc flow. To perform this task, the update gate at the GRU was modified to be unweighted, with its input being the surprise received as input from the Surprise gate. The combined gate now gets the new context from LSTMc and the hidden cell state (context of the previous time step) as inputs. The reset gate was removed as it has no role here. The following figure shows the used GRU structure. Note that the dotted lines denote unweighted inputs. Fig.4.4 shows the detailed structure of the GRU-like layer.

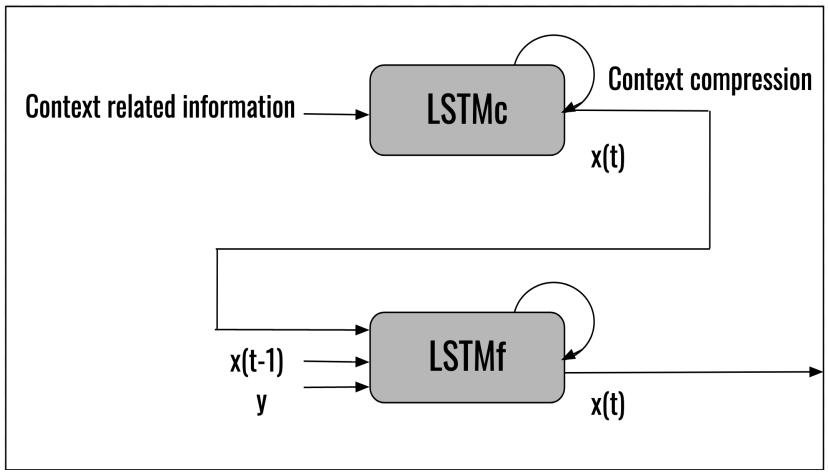


Figure 4.2: *A hierarchical structure composed of two layers: upper layer for contextual information processing and generation of context compressions, and a lower layer for sensorimotor information processing and to perform the predictive task.*

4.3 Experiments

The performed experiments included having a prediction task performed by the sensorimotor layer, for which the useful contextual information are being provided by the top-down layer after being received from the context layer. For our experiment, we used a simple example of a time series that includes four functions representing four different contexts or events: an addition function (Add) that adds the two inputs, a subtraction function (Sub) that subtracts one input from another, a Sine function (Sin) that applies the Sine function on one of the inputs and adds the result to the other one, and a constant function (Const) that outputs one of the inputs without any change and ignores the other one. The switching between the different contexts happened at random frequencies.

4.3.1 Single network experiments

As a proof of concept, we first used a single LSTM layer, as the one showed in Fig. 4.1, which takes the function value at the previous time step (x) and a randomly generated value (y) between -1 and 1 and outputs the prediction about the next function value that would be the result of applying one of the four functions on the x and y inputs. Then, we added the input of context related information represented in a one-hot vector denoting the ongoing event. Next, we tried to include the preparation part on a single-layer level, such that the provided contextual information were switched at random frequencies earlier that the actual context switch to simulate the idea of preparing for the next event before it actually happens.

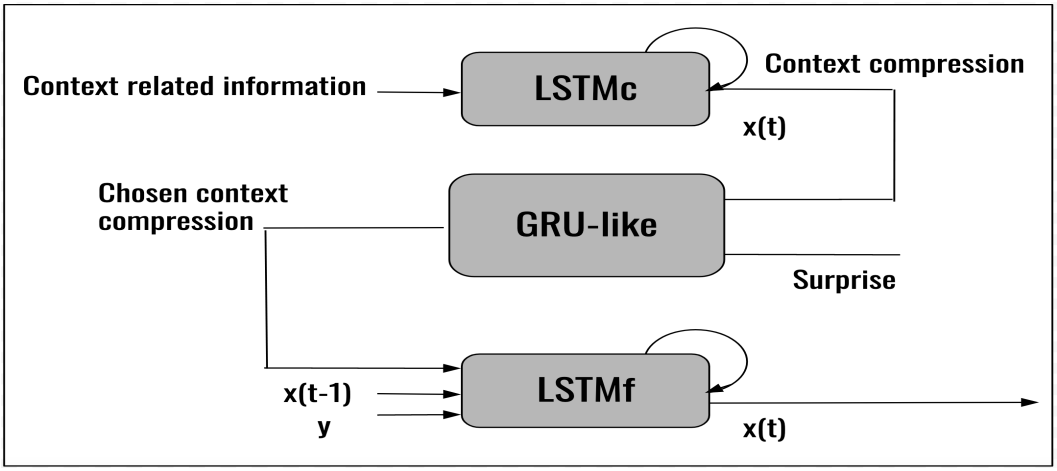


Figure 4.3: *The hierarchical structure composed of a deep contextual layer (LSTMc), a GRU-like gating layer and a low-level function processing layer (LSTMf). An MLP to preprocess the function input of LSTMf and a similar preprocessing unit with LSTMc were also tested (not shown).*

The idea behind using an LSTM layer was for the good performance it shows when processing time series data. This would be helpful when no contextual information is provided, so reflecting on previous steps can be helpful. We repeated the experiments using an MLP layer instead of the LSTM layer to compare the performance of the two networks.

The results are shown in Table 5.1 and Fig. 4.5, which show that using only one layer for the sensorimotor and contextual processing provides the best result when no planning is included, such that the network receives the exact contextual information on the ongoing event, better than when planning for the next event is included. The worst performance is obtained when the network does not receive any context related information.

In addition, so far the switches between the different contexts happened at random frequencies, but the consecutive functions were in the same order; thus, we ran an experiment in which the next function was randomly chosen. The results show that the network keeps the same good performance.

Regarding the LSTM vs. MLP, the results show that when no contextual information is provided, or when the information is provided but earlier that the actual function switch, the LSTM has a better performance than the MLP. On the other hand, the MLP performs better when the contextual information is provided, both with a fixed or a random order of the functions. This is because providing the helpful information makes it less advantageous to have the recurrency characteristic; on the contrary, it causes a drop in the performance.

In Fig. 4.6 and Fig. 4.6, the effect of including the contextual information while

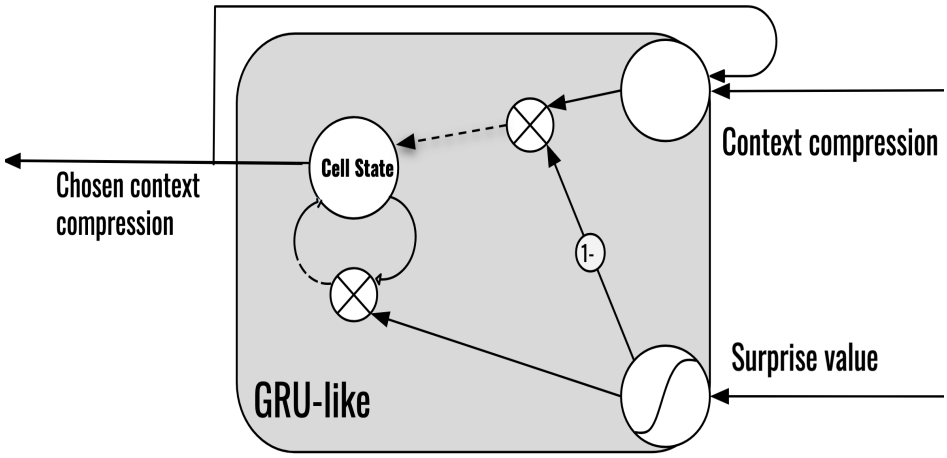


Figure 4.4: *The edited GRU layer used as a gating layer between the contextual layer and sensorimotor (functional) layer.*

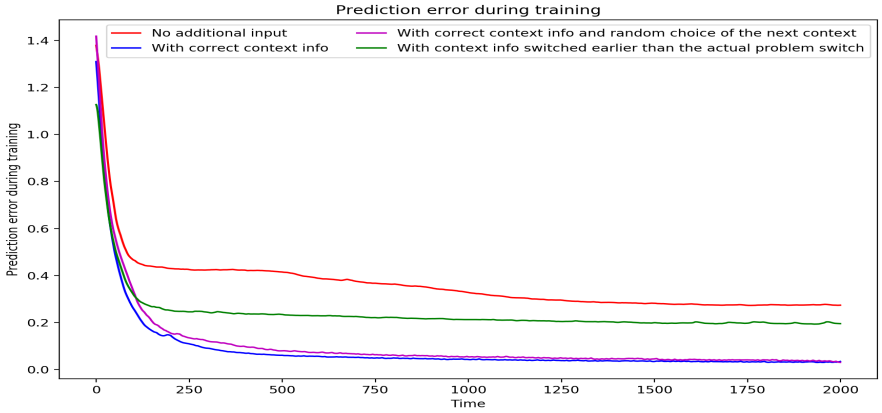


Figure 4.5: *The average training error in single LSTM layer experiments. We can see the effect of providing contextual information on decreasing the prediction error.*

Table 4.1: *Average training error in single LSTM and MLP layer experiments.*

Experiment	LSTM		MLP	
	avg. error	stdev.	avg. error	stdev.
No CI provided	0.2670	0.0272	0.4180	0.0016
CI provided with fixed function order	0.0533	0.0292	0.0098	0.0011
CI provided with random function order	0.0551	0.0215	0.0139	0.0022
CI provided but switched earlier	0.1947	0.0134	0.3180	0.0012

performing a prediction task can be clearly seen in the training phase and also in the testing phase, respectively.

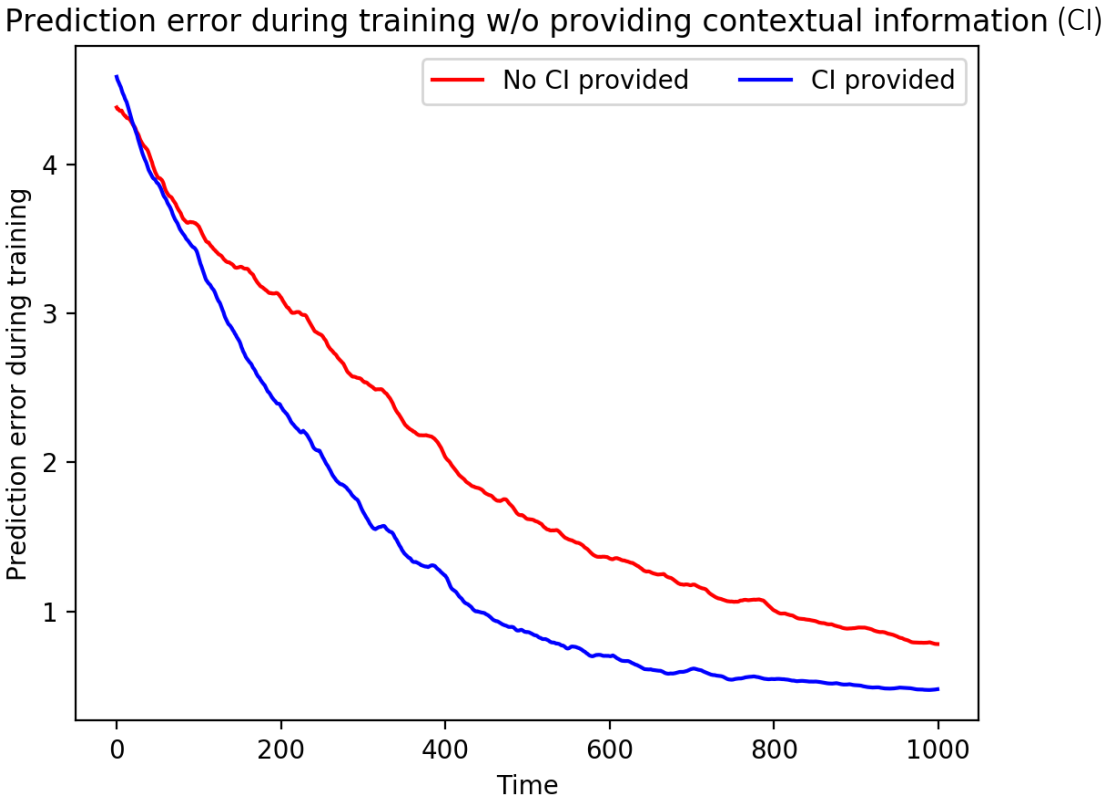


Figure 4.6: *The prediction error during training with and without providing the contextual information (CI).*

4.3.2 Full network experiments

Next, we ran the full structure composed of a deep LSTM context layer (LSTMc), a top down GRU-like layer and a sensorimotor functional LSTM layer (LSTMf). LSTMc receives context-related information as an input represented in a one-hot vector denoting the ongoing event, and outputs a compression of this event that is passed on to the gating GRU-like layer, which receives the context compression and decides on whether to pass on the new context compression or to keep using the same one. This decision is based on whether a ”Surprise” was encountered. This ”Surprise” happens when actual observations do not match the predictions, because the context has switched. In this experiment, and in order to closely investigate the role that these surprise signals play in event processing, we provided different values of the surprise signals and observed the outcomes. After the gate status has been decided on, the output of the gating layer is provided to LSTMf, which takes it as an input alongside the sensorimotor input, i.e., the two values on which the function will be performed in this example. We evaluated the structure by testing four cases:

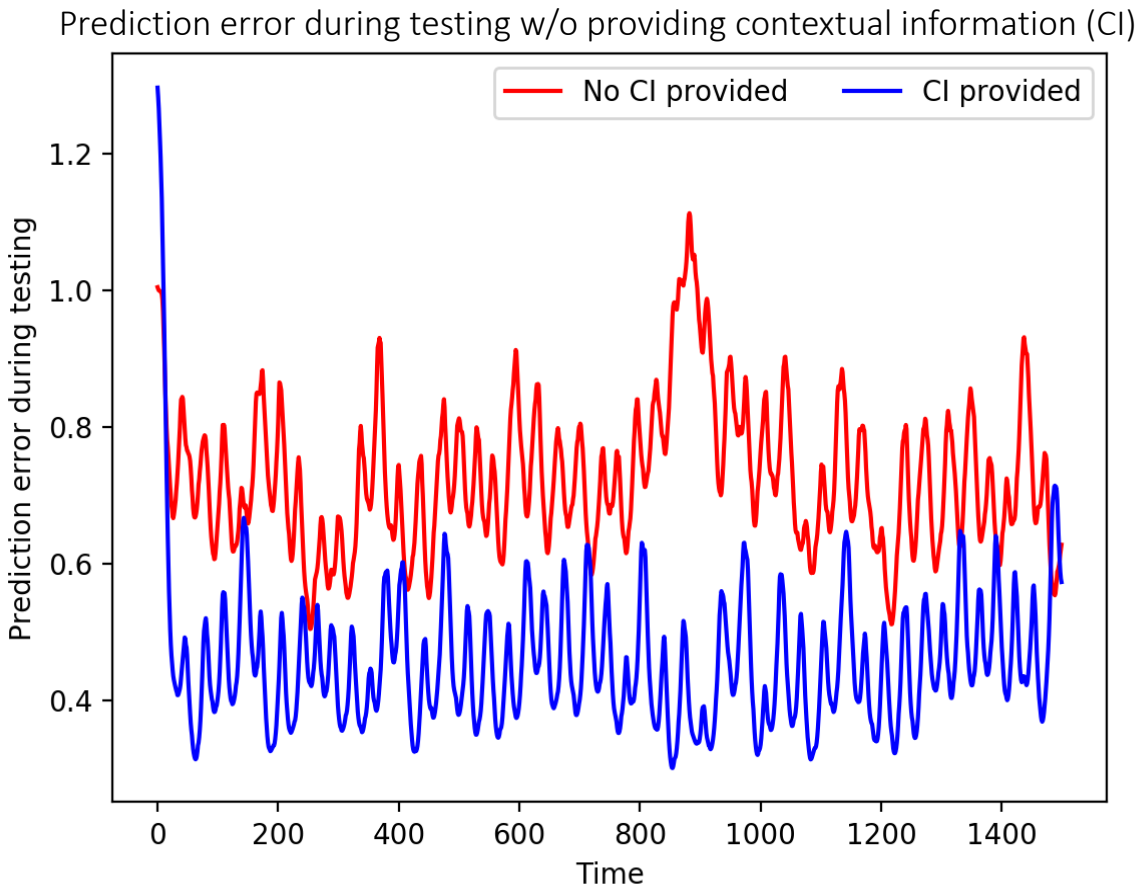


Figure 4.7: *The prediction error during testing with and without providing the contextual information (CI).*

1. The gate is always open: In this case we keep providing large surprise signals. The GRU output is mainly contributed by the new context from LSTMc.

2. The gate is always closed: In this case we keep providing small surprise signals. The GRU output is mainly contributed by the old context.

3. The gate is half open: The provided surprise signals are large enough to keep the gate half open, i.e., the old and new context contribute equally to the GRU output.

4. The gate is only open at context switches: This case simulates providing the perfect surprise signals only at the switch of the event. The gate is closed otherwise.

We ran the four scenarios on the structure and computed the average overall pre-

Table 4.2: *Average training prediction error and average distance between the centers of the clusters formed by the values of the context compressions in different gate states. The lowest average error and largest average distances are marked in bold.*

Gate status	avg. error	stdev error	Compared clusters	avg. distance	stdev distance
Always closed	0.280	0.059	Any	0.0	0.0
Always open	0.206	0.014	Add Sin	0.28	0.17
			Add Sub	1.22	0.42
			Add Const	0.64	0.19
			Sin Sub	1.27	0.34
			Sin Const	0.70	0.24
			Sub Const	0.7	0.26
Only open at switch	**0.059**	0.017	Add Sin	**0.69**	0.15
			Add Sub	**3.12**	0.42
			Add Const	**1.46**	0.27
			Sin Sub	**2.59**	0.47
			Sin Const	0.92	0.17
			Sub Const	**1.72**	0.4
Gradually opened	0.083	0.030	Add Sin	0.61	0.17
			Add Sub	2.17	0.69
			Add Const	1.35	0.56
			Sin Sub	1.81	0.39
			Sin Const	**1.00**	0.20
			Sub Const	0.82	0.31

diction error during training and the average distance between the clusters formed by the values of the context compressions. Then, we computed the average over 10 differently initialized networks. The results are shown in Table 4.2.

The results show that the best performance can be obtained by keeping the gate closed while the same context is going, and only opening it when a new event starts to pass the new event compression to the GRU to provide it to LSTMf. The worst case is when the gate is always closed, so no context information is provided from LSTMc. Similar results can be obtained using a random order of consecutive functions. The training performance average over 10 different network is shown in Fig. 4.8.

So far, we have been using a fixed weight update frequency of 20 time steps, such that we backpropagate the error along the 20 previous time steps. Next, we tried with different weight update frequencies to show the effect of this parameter, the results are shown in Table 4.3. Good results can be obtained when a random weight update frequency between 10 and 30 is used. We can observe that the best result can be achieved when the provided surprise signal is gradually changed. This might be because this gradual open/close of the gate increase the likelihood of convergence. This shows the importance of further investigating the gradual surprise signal.

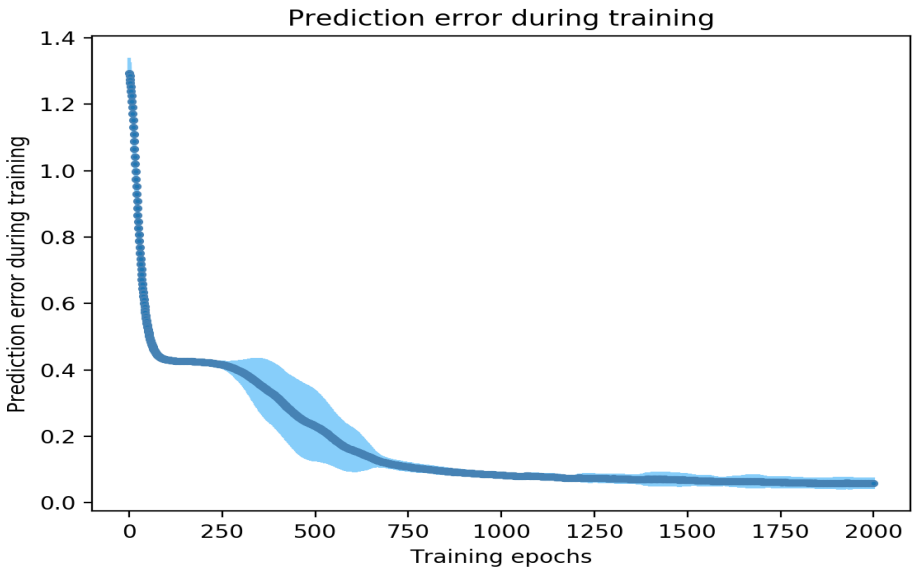


Figure 4.8: *The prediction error during training with the standard deviation averaged over 10 different networks.*

Table 4.3: *Average training prediction error while using different weight update frequency settings.*

Weight update frequency	Fixed at 35		Random 20-50		Random 10-30	
	avg. error	stdev error	avg. error	stdev error	avg. error	stdev error
Always closed	0.365	0.070	0.428	0.083	0.345	0.078
Always open	0.270	0.071	0.224	0.022	0.206	0.018
Open at context switch	0.200	0.142	0.318	0.122	0.166	0.149
Gradually opened	**0.106**	0.077	**0.103**	0.041	**0.070**	0.013

In order to show the importance of the hierarchical flow of information, we tried to provide the surprise signal directly to LSTMc, which led to worse results. This points out the role the surprise signal plays in gating the information flow. Then, we removed the preparation part regarding the early switch of the contextual information input and provided in-tune contextual information to LSTMc. The results showed that the network still performs equally well and achieves the best results when the gate is only open at the event switches. This shows that although the information is now tuned in with the current event, having the gate always open with contextual compressions flowing all the time to LSTMf will negatively affect the performance as it leads to unstable and varying compressions, even if slightly only. Finally, we tried the other alternative for the network type that we tried before with the single-level network, the MLP network. The results show that although the

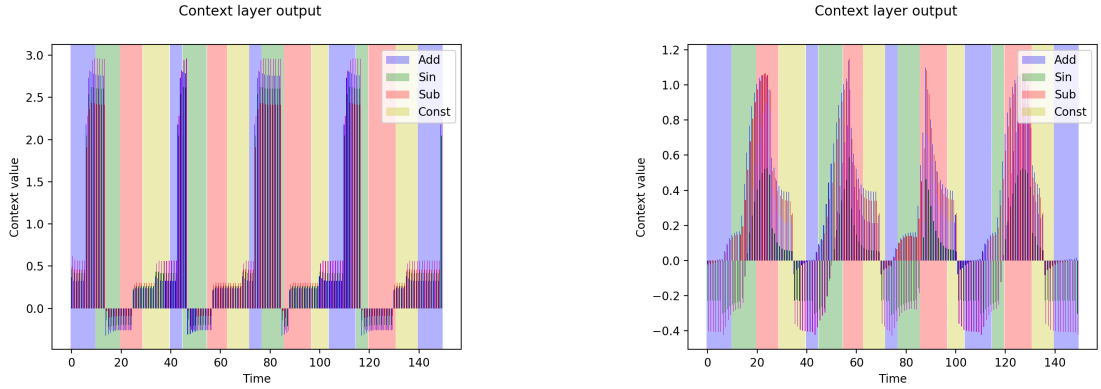
MLP achieved a good performance in the simple case of a single-level network with contextual information directly provided, it performed badly when used within the hierarchical multi-level structure. The results are shown in Table 4.4.

Table 4.4: *Average prediction error when (i) the surprise signal is fed to LSTMc, whereby the GRU-like gate is always open (Surp. to LSTMc), (ii) the context information is provided to LSTMc exactly in tune with the function event (In-tune CI to LSTMc), and when an MLPf is used instead of an LSTMf (MLPf).*

Input to LSTMc /	Surp. to LSTMc		In-tune CI to LSTMc		MLPf	
Gate status	avg. error	stdev	avg. error	stdev	avg. error	stdev
0 / Always closed	0.2515	0.0678	0.310	0.080	0.4213	0.00164
1 / Always open	0.2280	0.0198	0.066	0.040	0.4215	0.00123
1 at c.s. / open at c.s.	0.1031	0.0555	0.055	0.019	0.4211	0.00165

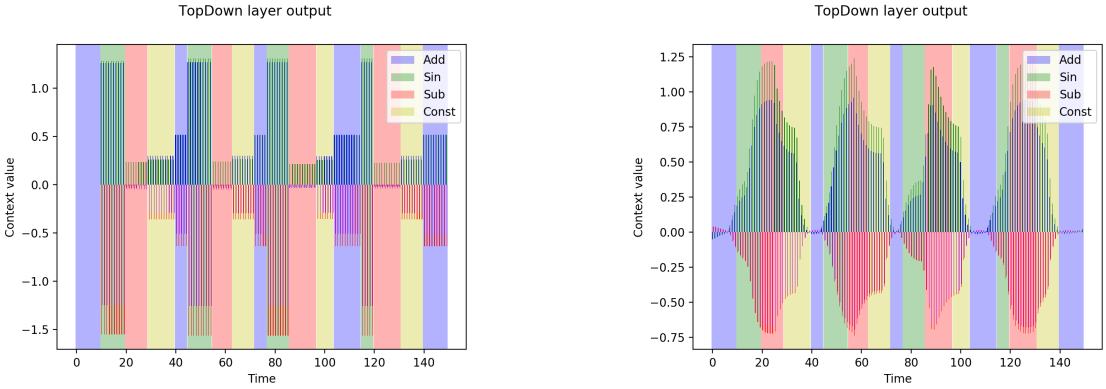
It is worth mentioning here, that the context related information that is provided as input to the LSTMc layer corresponds to the currently ongoing context for some time, then switches to denoting the next event that is about to take place (Event boundary signal or EB), as this layer starts to plan for the next event. Here comes the role of the GRU-like gating layer in keeping the preparation part going in the LSTMc while blocking the new context compression from being passed to LSTMf before the actual switch happens. When the preparation part was excluded, and the provided contextual information was equivalent to the actual ongoing event, opening the gate only at the switches yielded the same performance of when the gate was always open or half-open; thus, triggering the call for new contextual information only at the switches is sufficient for the best performance, which can have significant resource-efficient effect when permanent triggering of the contextual information can be wasteful regarding the resources (computational power, memory, etc.).

In order to have a closer look on the event compressions generated by the network, we plotted their values during testing. Fig 4.9 shows these context compression produced by the deep context layer structure, while Fig 4.10 shows the values that are decided on by the GRU-like top down layer. We can see the stable compressions when the gate is only open at the switches, which resulted in the best performance. Fig 4.11 shows the context compressions for nine differently initialized networks. We can notice that the increasing function has a context-encoding that is always close to zero. This might be because the network activities are alway reset to zero before starting with the first function of increasing at each epoch. Besides, the encoding for the constant function is between the increasing and decreasing encodings. Furthermore, the sine function has a distinct compression.



(a) *Gate open at switch* **(b)** *Gate always open*

Figure 4.9: *Context compressions produced by the context layer. Background colors indicate the different contexts.*



(a) *Gate open at switch* **(b)** *Gate always open*

Figure 4.10: *Context compressions produced by the GRU-like gating layer. Background colors indicate the different contexts. Note that as the gate was still closed during the first event in (a), context values are still on zero.*

4.4 Retrospective inference

The previous experiments showed that prospective context inference using surprise signals can be applied to segment the continuous sensorimotor stream of data into separable events. Reflecting on the retrospective context inference performed in REPRISE (see Chapter 3), we went on to investigate if a retrospective context inference process can be added to the current structure. The aim here was to infer the output of the gating layer, i.e., the contextual input to LSTMf.

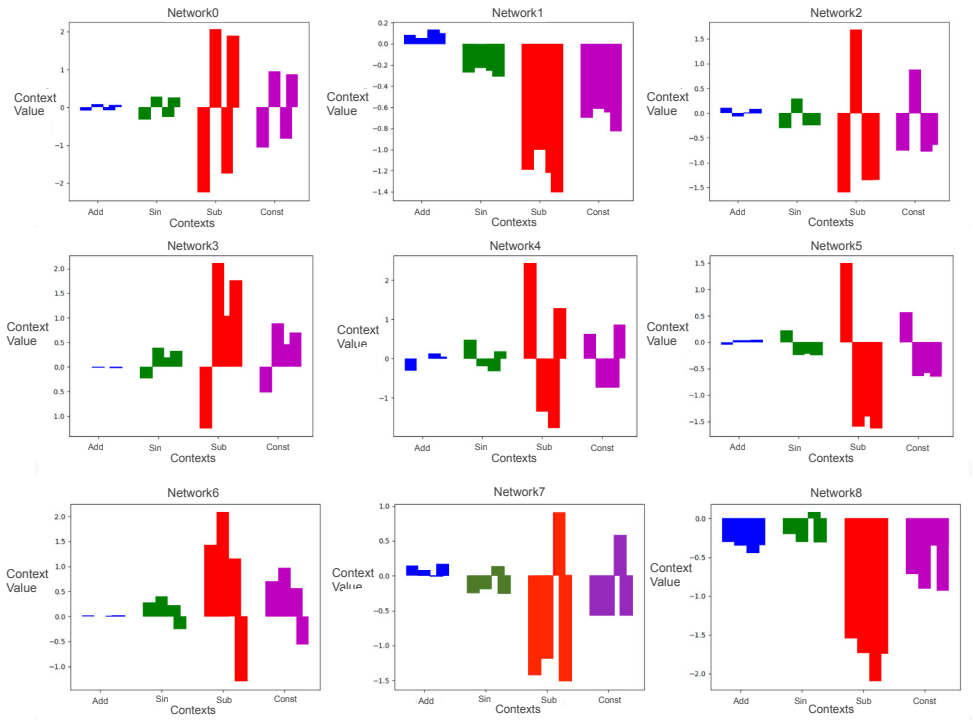


Figure 4.11: *Context compressions in nine differently initialized networks.*

First, the predictive task was performed for the first two events. Then, at the end of the second event, retrospective inference was performed, which benefited from the resulting prediction error during the event to infer the event encoding that should have been provided by the gating layer. An illustration of the execution timeline is shown in Fig. 4.12.

After evaluating the results of retrospective context inference, the inferred contextual encodings were not informative. One reason that might explain why the retrospective inference, which proved useful in REPRISE, did not result in well-adapted context compressions here might be the effect of providing contextual information at the deep layer. In this case, two different directions of information flow in the structure are cancelling each other's effect: the provided contextual information flowing from the upper layer to the lower one, and also the retrospective inference performed at the lower layer inferring the contextual values to be set as the input from the previous layer. Hence, we did not include this component in later experiments. It will be interesting though to further investigate a possible way to benefit from both adaptive sources to shape the best contextual compressions.

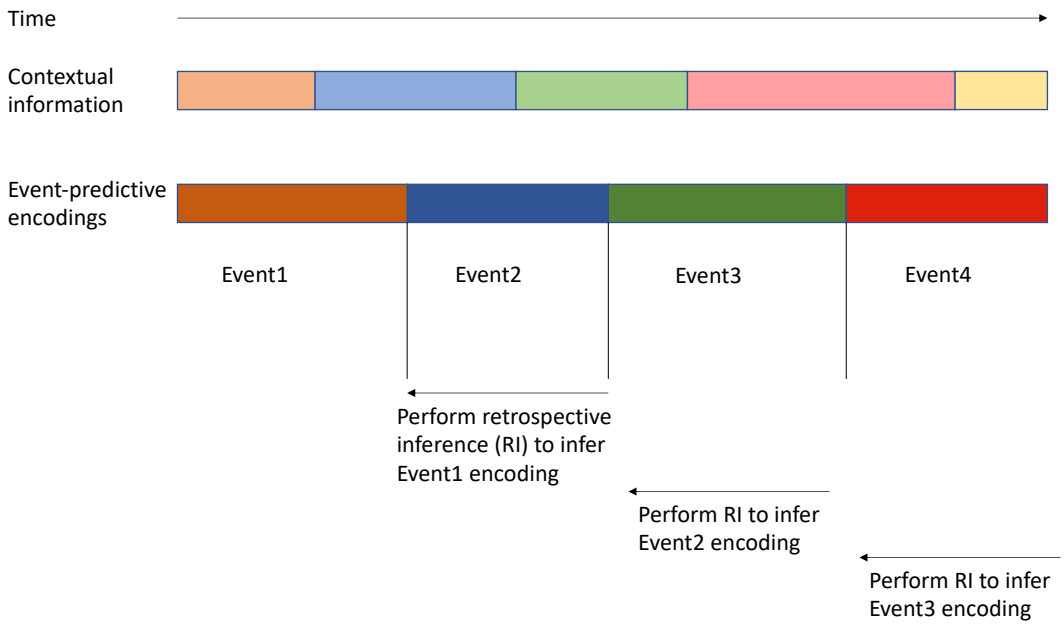


Figure 4.12: *An illustration showing the timeline of performing retrospective inference. At the end of each event, RI is performed to infer the suitable event encoding of the current event.*

4.5 Conclusion

The computational models of event cognition are a hot research topic. These models aim to validate neuroscience theories about how our mind works. While implementing stand-alone units for specific tasks can contribute to the overall understanding, simulating the integrated functionalities of interrelated brain areas within models that are structurally inspired by the information processing structure in the brain is an important step towards an overall understanding of the brain functionality and more flexible robotic agents. In this chapter, we presented a hierarchical neural network structure that contains a deep preparation layer, processing contextual information and generating suitable event compressions, which pass through a top-down layer to decide between this newly arrived compression and the one from the previous time step. If the surprise is low, then the same old compression is used, as this indicates that the same event is still ongoing; otherwise, the new context compression is passed to the sensorimotor processing layer, which performs a certain predictive task.

Our structure shows that the best predictive performance can be obtained when the gate is only open as a new event starts, compared with the cases of being always close, open or half-open. The deep context layer does not only consider the currently ongoing event, but starts after a while to prepare for the next one. Thus, it is important that the gating top-down layer passes on a new context compression only when a new event actually starts.

Event-triggered learning was proposed for control, such that the system only requests new information and the model is updated when learning is actually needed [74]. To this end, our suggested structure shows that even when the part of preparing for the next event is not included in the structure, and the context layer always receives the information regarding the actual ongoing event, the performance of the network when the gate is only open at the switch is no less than that when the gate is always open or half-open; thus, the same prediction accuracy is achieved in a significantly more resource-efficient manner.

Chapter 5

Implementing predicted unpredictability

After showing that providing perfect surprise signals helps to efficiently segment the stream of data into distinct events in Chapter 4, this chapter introduces the enhancement of the gated surprise model with 1) a module to calculate surprise signals directly from the prediction error, 2) a module to anticipate surprise using masked information that indicates an upcoming event switch, 3) a counterfactual regularization process to evaluate alternative decisions regarding the gate control, such that the best option is picked to achieve more stable event encodings [1].

[1] This chapter is based on our published paper:
Humaidan, D., Otte, S., Gumbsch, C., Charley, W., Butz, M. V. (2020) Latent Event-Predictive Encodings through Counterfactual Regularization. The 43rd annual meeting of the cognitive science society. Most of the figures and tables are taken from this paper.

5.1 Surprise prediction

The use of prediction error-based surprise signals allowed to use the appropriate event encoding. It also enhanced the results of the prediction task compared with when no event information is used. However, the fact is that waiting for the prediction error to increase to realize the change in the context only happens in really surprising events. For example, if it suddenly started to rain after we left our place, we will update our plan and probably go back to pick up an umbrella. In normal times, we would already estimate the end of the current event. Baldwin and Kosie [6] had an interesting way to form this idea as: predicting the unpredictable. While we do not always know what would happen next, we always have clues that help to expect a certain event to end. In the proposed "Free Energy Principle" [22] [23], Friston expressed that being able to avoid surprise means that we need to take into account every possible thing to happen next, which is impossible; otherwise, we intend to minimize our surprise as much as possible, by trying to limit the possibilities to a set of well-realized states. An example of such prediction of the event boundaries might be when we predict that the lecture is about to end because the time is almost over, or because the lecturer said "I will stop here for today .. ".

To implement a smooth surprise-free transition between the events, we extended the model presented in Chapter 4 with a surprise prediction module. The role of this module is to receive event boundary information indicating when the current event might end, such that an appropriate surprise signal can be predicted in a timely manner at the switch, which reduces the increase in the resulting prediction error at the switches between different events.

Overall, we implement three models in this chapter, which differ according to the source of the surprise signal and how it is decided upon the use of old or new information to construct event encodings.

5.2 Event Switching Layer

The event switching layer is implemented using a state switching gate, which is also used in gated recurrent unit RNNs [14]. The switching gate controls when the next top-down guidance signal from the event anticipation layer will be passed down to the event processing layer. The input to the event switching layer comes from the event anticipation layer ($\mathbf{x}_a$) and the event boundary layer ($\mathbf{x}_b$). We denote its input layer with η and its update gate with ζ.

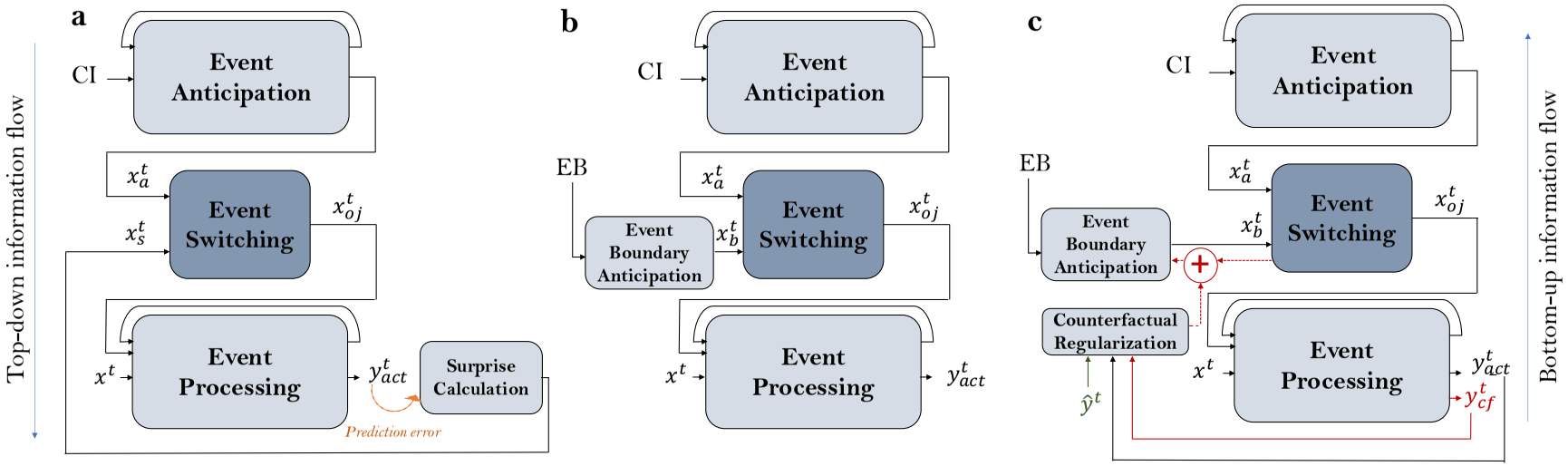


Figure 5.1: ***a)*** *$SUGAR_a$: the surprise signal is calculated online and used to control the update gate in the event switching module.* ***b)*** *$SUGAR_b$: the surprise signal is anticipated in a dedicated event boundary anticipation module, which receives fuzzy event boundary information as input.* ***c)*** *$SUGAR_c$: similar to $SUGAR_b$ regarding surprise anticipation, but counterfactual regularization is added to foster more precise gate opening and more compact latent event-predictive encodings. CI: contextual information input, EB: event boundary information input.*

5.2.1 Forward pass

The switch between maintaining the previous latent event encoding or updating it with new contextual information is controlled through the scalar activity x_ζ of the update gate, which is defined as follows:

$$\begin{aligned} net_\zeta^t &= x_s^t \\ x_\zeta^t &= \varphi_\zeta(net_\zeta^t), \end{aligned} \tag{5.1}$$

where x_s^t is the surprise signal input at the current time step t, and φ_η represents the sigmoid activation function of the cell.

The output of the event anticipation layer $\mathbf{x}_a$ combined with the previous hidden state of the event switching layer $\mathbf{x}_h$ constitute the activity $\mathbf{x}_\eta$ of the input layer as follows:

$$\begin{aligned} net_{\eta j}^t &= \sum_i w_{ij}^a x_{ai}^t + \sum_{j'} w_{j'j}^\eta x_{hj'}^{t-1} \\ x_{\eta j}^t &= \varphi_\eta(net_{\eta j}^t), \end{aligned} \tag{5.2}$$

where the activation function φ_η is linear.

Then, the input layer activities are fused with the previous hidden layer activities, dependent on the current activity of the update gate, to give the hidden cell state $\mathbf{x}_h$ as follows:

$$x_{hj}^t = x_\zeta^t x_{hj}^{t-1} + (1 - x_\zeta^t) x_{\eta j}^t \tag{5.3}$$

Lastly, a regular feed-forward pass is used to define the output of the event switching layer, which is fed into the event processing layer, as follows :

$$\begin{aligned} net_{oj}^t &= \sum_i w_{ij}^o x_{hi}^t \\ x_{oj}^t &= \varphi_o(net_{oj}^t), \end{aligned} \tag{5.4}$$

where the respective activation function φ_o is linear.

5.2.2 Backward pass

We used standard backpropagation through time to update the parameters of the network throughout the model. The ADAM optimizer was used for the weight update. The error term was defined as the squared reconstruction loss, which is calculated as the squared difference between the prediction output of the event processing layer and the label. Hence, the full model was trained end-to-end.

The following equations were used to compute the gradient.

The cell output:

$$\delta_{hj}^{t} = \epsilon_{hj}^{t} = \Sigma_k w_{jk}^{t} \delta_k^{t} + \Sigma_{j'} w_{jj'}^{\eta} \delta_{\eta j'}^{t} + x_{\zeta j}^{t+1} \delta_{hj}^{t+1} \tag{5.5}$$

The update gate:

$$\delta_{\zeta j}^{t} = \Phi' \zeta j^{t}(net_{\zeta j}^{t})(-x_{\eta j}^{t} \delta_{hj}^{t} + x_{hj}^{t-1} \delta_{hj}^{t}) \tag{5.6}$$

The input gate:

$$\delta_{\eta j}^{t} = \Phi' \eta j^{t}((1 - \Phi_{\zeta j}(net_{\zeta j}^{t}))\delta_{hj}^{t}) \tag{5.7}$$

5.3 Calculated Surprise

We adopted the example used in Section 3.2 to test the use of a generated surprise signal, which is calculated based on the prediction error.

The mechanism is based on keeping a moving average of the prediction error, such that an error value that is significantly higher than the average is considered as a surprising signal. First, we calculated the difference between the current error value and the average value. Then, the surprise signal was calculated by dividing the difference by the current standard deviation of the surprise signal.

The results show that using the prediction error to calculate a corresponding surprise signal resulted in opening the gate right after the switch, in addition to a few openings during the events. The resulting prediction error was comparable with that obtained when the perfect surprise signal was provided.

The average prediction error of the calculated surprise compared with the provided surprise during training is shown in Table 5.1.

Table 5.1: *Average training error for the functions example in Chapter 3 including the calculated surprise.*

Surprise Signal	Gate	Average Prediction Error
Provided	Always closed	0.280
Provided	Always open	0.230
Provided	Open at switch	0.059
Provided	Gradual open/close	0.083
Calculated	According to the surprise	0.080

In addition, Fig.5.2 shows the calculated surprise signal, with spiking signals indicating a high surprise mainly after the switches, and a few times within events.

Next, we checked the difference between the calculated surprise scenario and the previously used perfect surprise signals, both when the contextual compressions were never updated, i.e., the gating layer was always stopping the information flow from

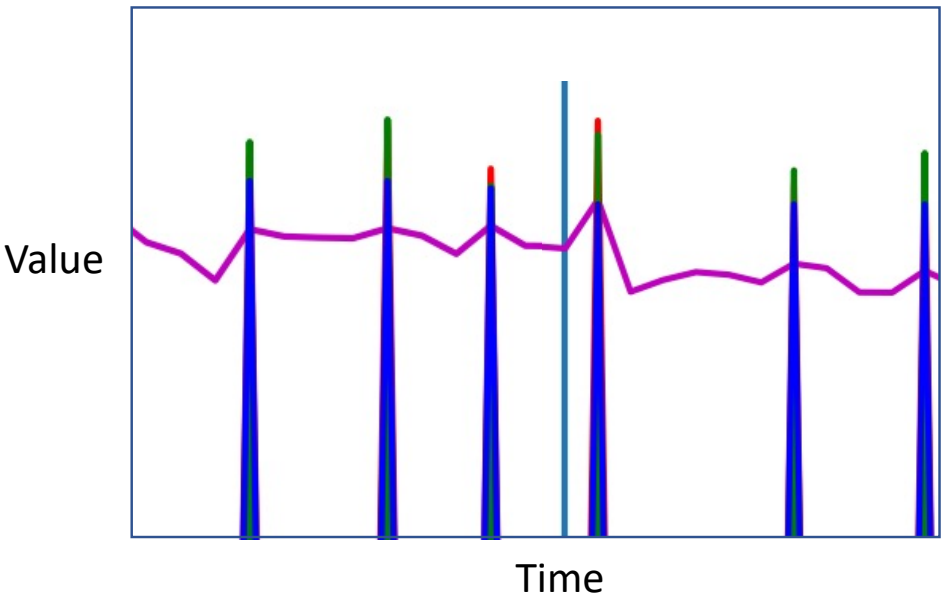


Figure 5.2: *The prediction error value (in purple) and surprise signals (in blue) along time steps. The vertical blue bar indicates an event switch. It can be observed that the prediction error significantly increases after the event switch. We can also note some time points within the event with a prediction error large enough to make the gate open.*

the contextual layer to the predictive layer, and when the gate was perfectly opened at the context switch. Fig. 5.3 shows the actual event switches and the surprise values during the testing phase of the network. We can notice that the performance of the calculated surprise was better that when the gate was always closed. In both cases, we can see the spikes in the surprise signal at the event switches, which are less in the calculated surprise case. The figure also shows the perfect case, such that the surprise signal does not spike across switches anymore because the gate is perfectly opened at the switches and the corresponding event-predictive encodings are passed to the predictive layer.

5.4 Predicted Surprise

We have seen that when calculated surprise is used, the signal spikes at event switches. This is because when the switch happens to a new event, the predictive task is still being performed in light of the previous event predictive-encoding. Then, since the event has changed, the prediction error is high, and it is translated into a spiking surprise signal that will lead to opening the gate and passing on a new event predictive-encoding. In order to achieve a seamless transition between events, we extended the structure by adding a surprise prediction module. This module receives surprise prediction information in the form of a high signal when a transition is predicted to

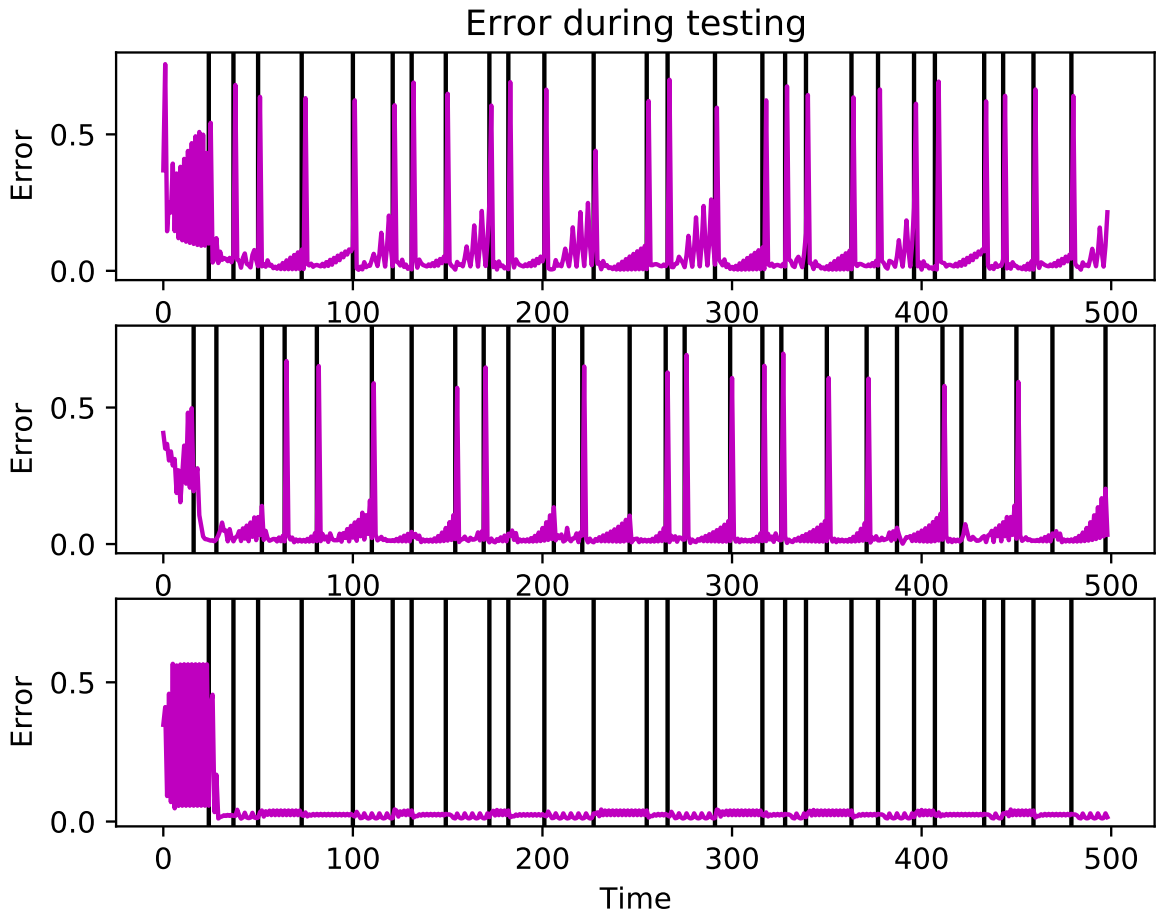


Figure 5.3: *Prediction error during the testing phase. Top: the update gate is always closed, middle: the update gate is controlled based on calculated surprise, bottom: the update gate is precisely opened only at event switches. Black lines denote event transition boundaries.*

happen, and it predicts the suitable surprise signal to be provided to the gating layer.

We used an example of a hierarchical sequence prediction task. The different problems consisted of different sequences of a set of symbols: $[A, B]^*$, $[B, C]^*$ and $[A, B, C, B]^*$. The switching time between different problems was randomly chosen from a uniform standard distribution U(10,30). The symbols of the sequence were provided as input to the event processing layer as a one-hot vector. The used example is shown in Fig. 5.4, and the general structure of the problems is shown in Fig. 5.4.

We evaluated the structure on this task and the results showed that using the surprise prediction module to predict surprise signals leads to a similarly accurate prediction as when the perfect surprise signal is provided. As shown in the example in Fig. 5.5, the prediction error values during testing show a smooth transition between events. Plotting the predicted surprise signals by the added module shows how spiking signals are predicted at the context switches. This allows to pass the con-

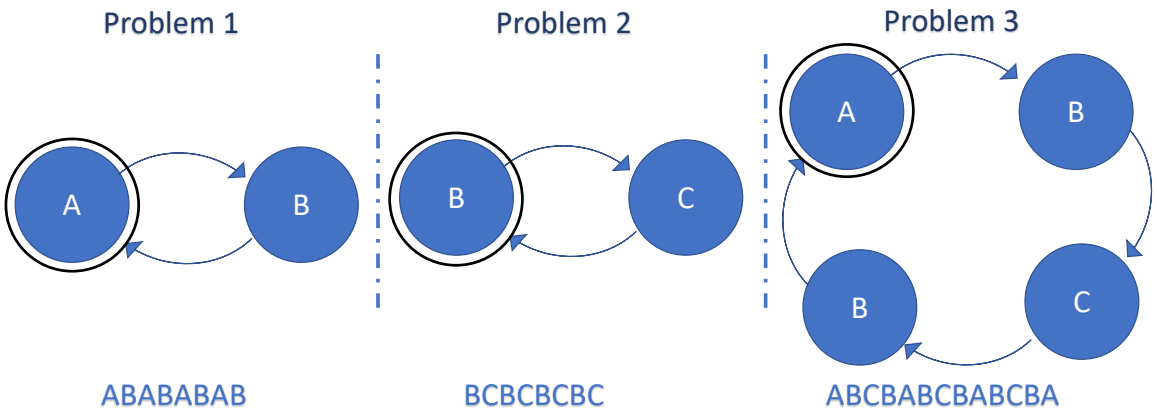


Figure 5.4: *The used symbolic sequence problems with examples of the sequences generated by each problem.*

textual encoding of the event that has just started, which leads to better prediction at the low-level layer.

To show the robustness of this prediction, we added a random input to the surprise prediction module. The results showed that even when mixed signals are being provided, the network learns to recognize those indicating the context switch and responds by predicting high surprise values at the switches.

5.5 Counterfactual Regularization

Although having a definitive clue that the current event has ended is essential to start acting within the context of the next one, we usually observe various indicators that the current event is about to end. We mimicked this idea by having the input to the surprise prediction module change gradually a few time steps before the switch and then at the switch step. The results shown in Fig. 5.6 showed that this would lead to the gate being open also before the actual switch. This resulted in the context compressions being unstable during the same event, leading to a slightly increased surprise signal between the events.

To tackle this problem, we investigated the possible effects of including counterfactual reasoning. We updated the loss function of the surprise prediction module by an additional error term. This term refers to the difference between the prediction error obtained using the current settings, and the error that could have been obtained should the settings be different, termed as the counterfactual error (CFR) term.

For every time step, after running the network using the surprise signal predicted by the surprise prediction module, we re-run the lower-level layer using the opposite status of the gate. If the gate is open at time t, then we re-run with the gate closed,

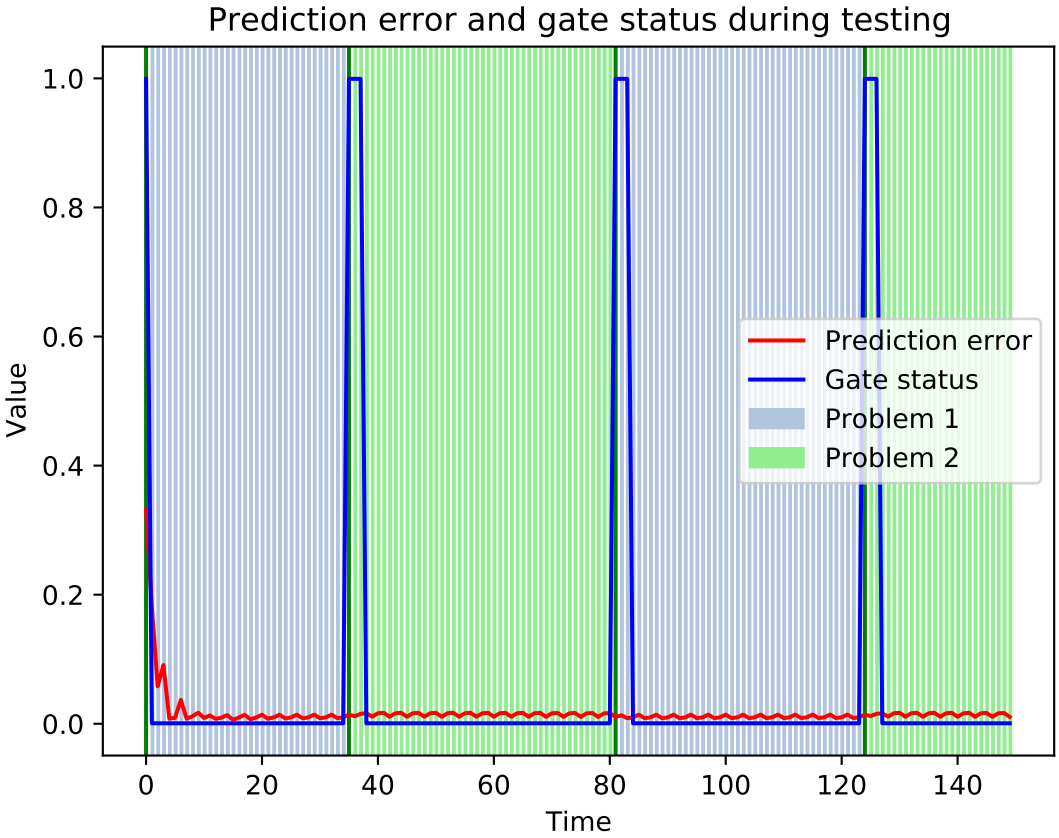


Figure 5.5: *Prediction error and surprise signals during the testing phase of the SUGAR$_b$ model on Problem1+2. The input to the event boundary anticipation module is an event boundary indicator plus random input.*

and vice versa. Then, the difference between the two prediction errors is calculated and used as a penalty term to be added to the loss function if the gate was open. Thus, such a signal could encourage the network to lean towards keeping the gate closed as much as possible.

The counterfactual regularization was included as an additional term in the gradient equation of the update gate as follows:

$$\delta^t_{\zeta,reg} = \delta^t_{\zeta} + \beta \left(\sqrt{(y^t_{act} - \hat{y}^t)^2} - \sqrt{(y^t_{cf} - \hat{y}^t)^2} \right), \tag{5.8}$$

where β indicates the gate status, with the value of 1 if the gate was open and 0 otherwise, y^t_{act} denotes the actual prediction of the event processing network, y^t_{cf} represents the alternative prediction if the gate had been in the counterfactual state, $\hat{y}^t$ is the true label, and $\delta^t_{\zeta,reg}$ is the gradient term that regularizes the utility of gate openings. Note that the values of β could be adjusted to more complex systems by setting it to be more scaled.

The results showed that the use of the counterfactual error term can result in more

stable context compressions, as shown in Fig. 5.8, compared with no counterfactual error, shown in Fig. 5.6. This might be due to further increasing the surprise signals at the event boundaries and further decreasing them within the events.

Having a closer look at the inter-event and intra-event changes in the prediction error and surprise signal, thus the gate status, reveals that the use of counterfactual regularization leads to less fluctuations in the surprise signal within the same event and more differences in the signal pattern among different events. These details can be seen in Fig. 5.7.

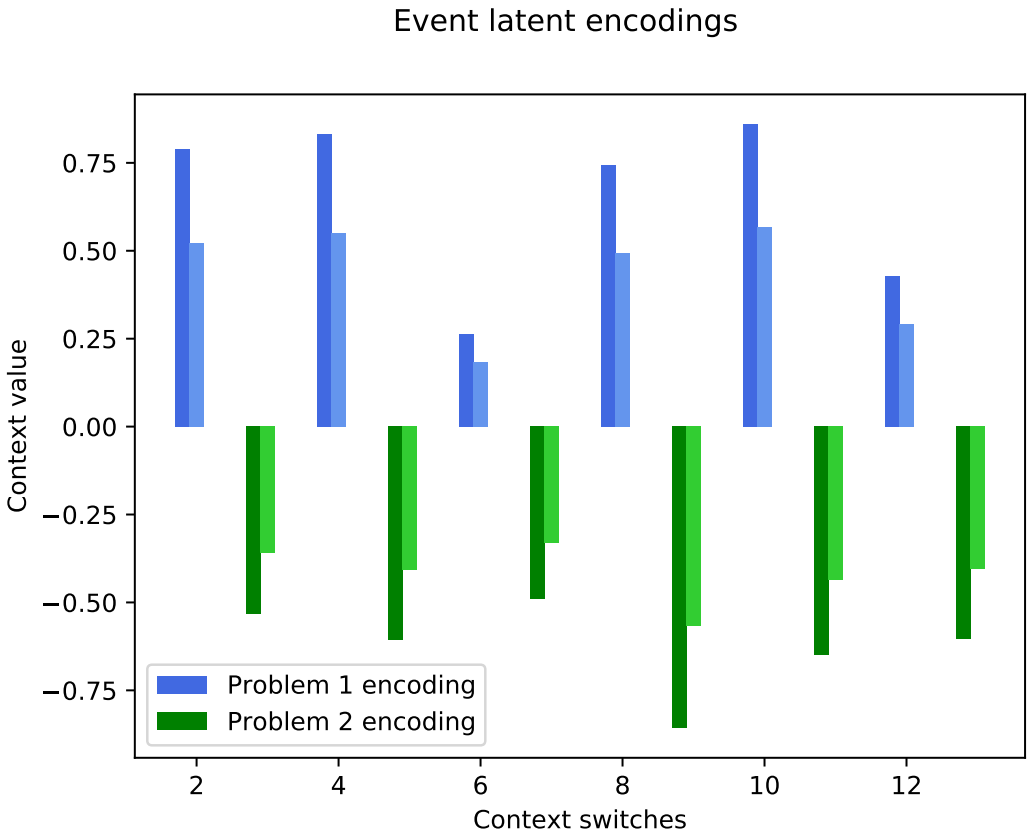


Figure 5.6: *Latent event encodings ($\mathbf{x}_o$) emerging when evaluating SUGAR$_b$ on Problem 1+2.*

5.6 Compositionality

An important skill in human learning is the ability to combine different experiences and use them within a new task that has some degree of similarity to each of the individual subtasks. In AI systems, many related learning techniques have been investigated. Continual learning represents the ability to remember different tasks

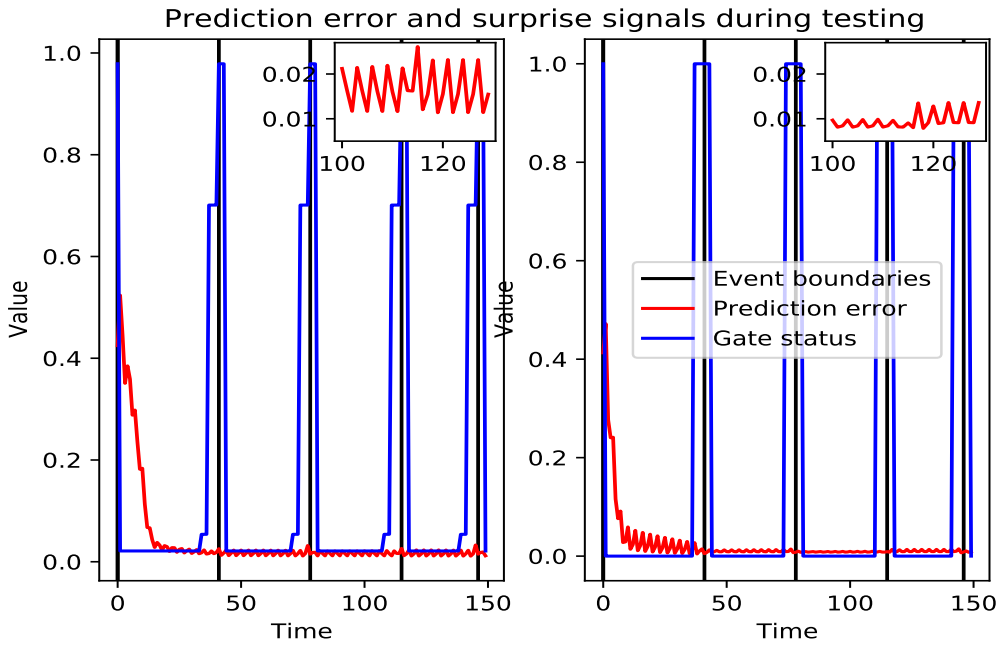


Figure 5.7: *Inter-event and intra-event gate status and prediction error for SUGAR$_b$ (left) and SUGAR$_c$ (right). The upper right corner includes a zoomed image of the error value when switching to a new event.*

without forgetting previously learned ones. Efficient learning means the ability to learn from a few examples only. Transfer learning denotes the capacity to transfer the experience acquired at a certain task to perform another that has something in common. These characteristics, especially the last two, represent areas in need for further research. In fact, as we humans have these techniques, we might say that scientists can inspire from neuroscience in developing them in artificial agents [37].

In this experiment, we wanted to investigate the developed encodings of different problems that have certain characteristics in common. Training the structure on problems that are partially similar brings up insights inspired by the principle of transfer learning, and how latent encodings formed from a stream of sensorimotor observations could be inspired or partially learned from encodings constructed based on somehow similar observations. Using the problems in Fig. 5.4, we investigated the latent event encodings developed in SUGAR$_b$ for the three problems. Fig. 5.9 illustrates these encodings for different runs with differently initialized networks. It could be observed that the encoding for Problem 3 is more similar to that of Problem 1. This is because, apart from the fact that the symbols (subproblems) in Problem 1 are used in Problem 3 as well, Problem 3 starts with the same symbols as in Problem 1. Still, there is a difference between the two encodings, which could be due to the fact that the symbols in Problem 2 are also included in Problem 3, but they do not

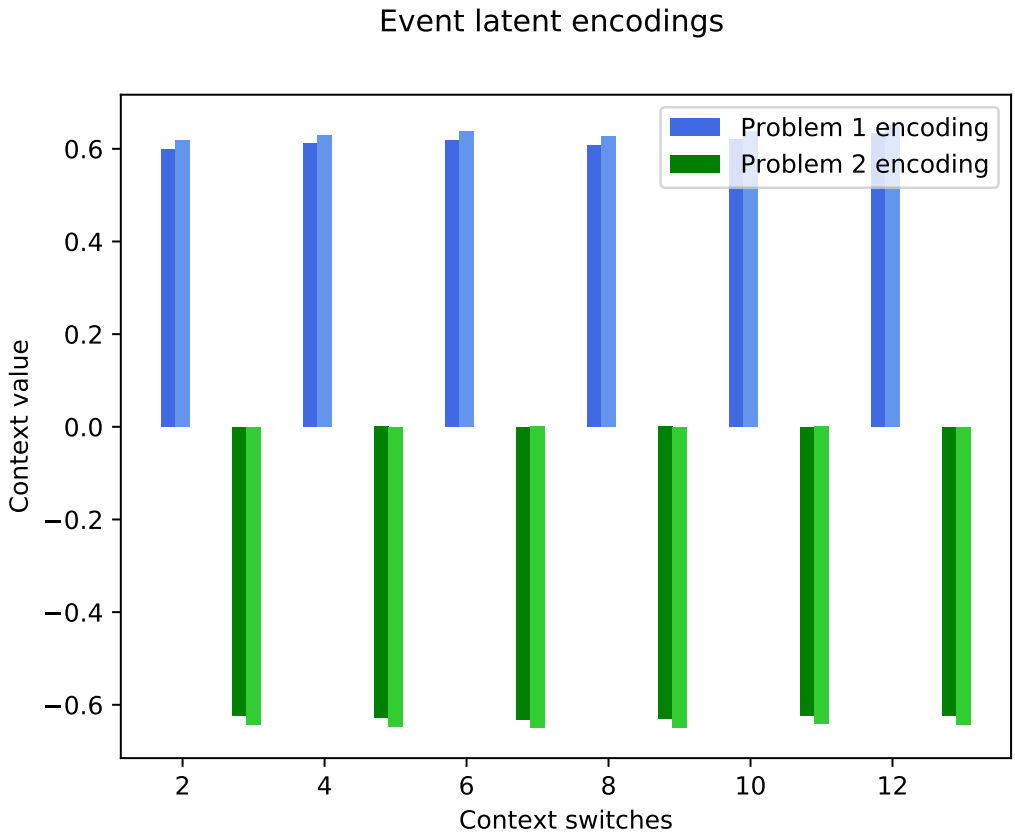


Figure 5.8: *Latent event encodings ($\mathbf{x}_o$) for Problem 1+2 emerging in SUGAR$_c$.*

appear in the very beginning.

5.7 Conclusion

Modeling cognitive functions using artificial intelligence is a very important research direction. This is not only due to its applications in the fields of robotics, natural language perception, etc., but also because connecting theories on how the mind works with practical simulations brings us closer towards human-level AI. Event cognition drives much of our behavior and helps us to solve the problems and understand the world around us [62]. In this chapter, we investigated an important aspect of event cognition, that is, seamless event segmentation. First, we showed that an appropriate segmentation of sensorimotor experience into events can be achieved using surprise signals developed based on significant increases in the prediction error. However, calculating surprise signals based on the prediction error causes a delay in identifying the beginning of a new event. This negatively affects the performance. Inspired by how our brain preempts surprise [6], we included predicted surprise signals using

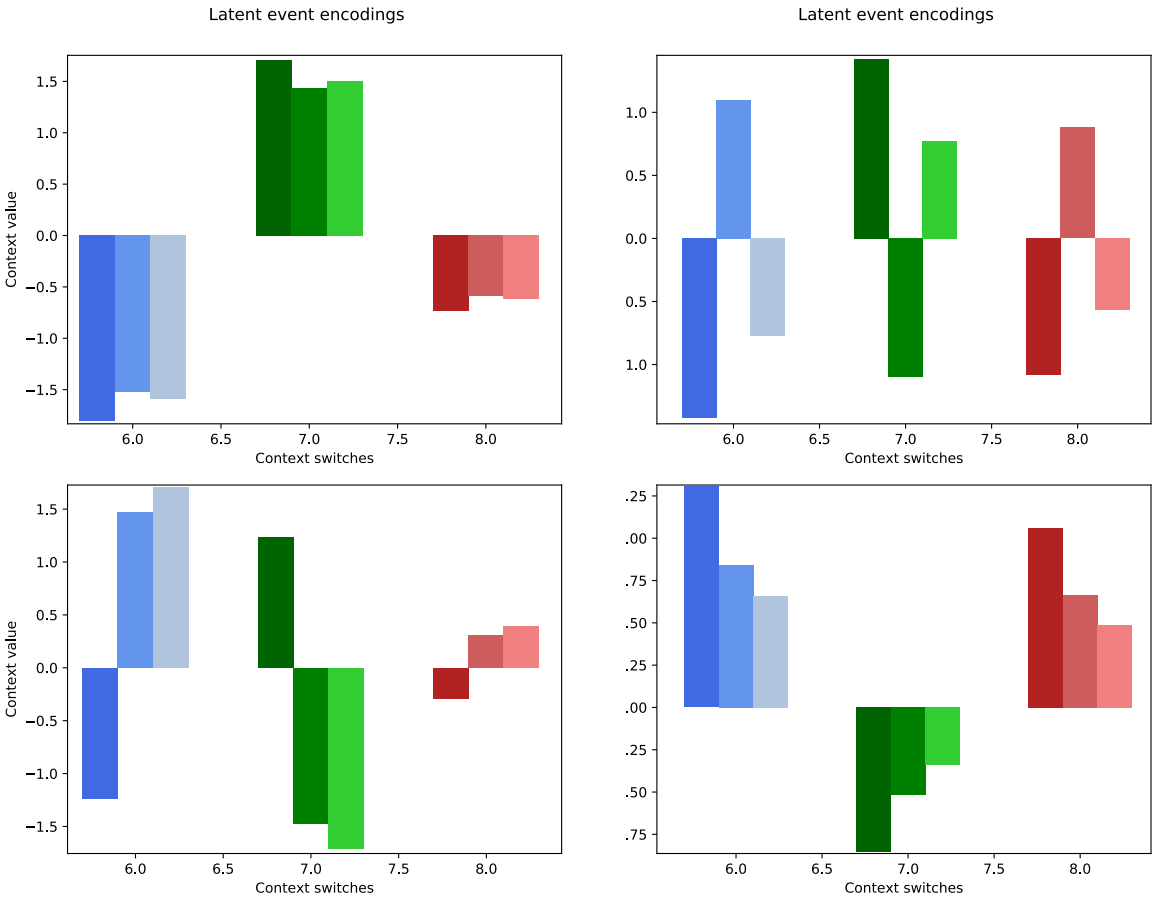


Figure 5.9: *Latent event encodings ($\mathbf{x}_o$) for four randomly initialized networks trained and evaluated on Problem 1-3. We can notice that the code for Problem 3 lies in between the code for Problem 1 and 2 (blue and green bars, respectively). Blue bars: code for Problem 1, green bars: code for Problem 2, red bars: code for Problem 3.*

provided information on when an event switch will happen. The result was good performance in the prediction task and a smooth, seamless transition between the events in terms of the prediction error. Still, if the provided information to the surprise prediction module includes further indications to prepare for the switch even before it happens, this might mislead the structure to let new contextual information flow in the top-down direction while the switch still has not yet happened. Thus, we added a counterfactual loss term to reason about the decisions of the network in letting the information flow or not, and adding a penalty when this happens at inappropriate times. As a result, we investigated the latent event encodings developed in our structure. Finally, we discussed how problems with similar subproblems can have similar latent event encodings to a certain extent.

Chapter 6

Discussion

In this final chapter, I will conclude this thesis by summarizing the presented ideas and reflecting on the resulting discussions. I will quickly revisit the research questions and the proposed investigations towards finding the answers. Then, I will discuss some of the limitations that appeared while working on this thesis. While some of them were possible to overcome, others remained as challenges that could represent interesting ideas for future work. Finally, I will discuss possible further investigations that could complement the presented work.

6.1 Summary

This thesis aimed to discuss the following questions:

- What role does contextual information play when performing predictive tasks? In other words, how does our knowledge regarding the overall ongoing event affect our ability to predict what might happen next?
- Can we design a hierarchical structure that simulates the proposed two-way bottom-up/top-down information flow in the brain?
- How do event-predictive encodings develop and how do they change between intra-event and inter-event moments?
- What role does anticipating surprise signals play in surprise-gated event switching structures?
- What would be the effects of having the ability to reason about alternative decisions: would this have happened had I acted in a different way?

6.1.1 The role of contextual information

An important part of robust perception of the surrounding world is the ability to explain what we observe in the light of a larger context, i.e., understand opening the door of the fridge to pick up milk as one step of larger activity of preparing coffee with milk. If such guiding information is not known, i.e., I saw someone opening the fridge without knowing their intent to prepare coffee with milk, then I will not be able to predict what will they do next.

A simple example of a predictive task was used to show the utility of top-down contextual information. Using the REPRISE structure [12], an evaluation of the context inference process was performed. The used example included three different vehicles moving towards a target location. The network received the current location of the vehicle and predicted the location at the next time step. The other two inputs to the network were a random motor command to move the vehicle and a one-hot vector encoding the currently controlled vehicle. The network performed two tasks:

- Prospectively, actively inferring the motor commands needed to reach the goal;
- Retrospectively inferring the identity of the currently controlled vehicle.

First, we trained the network while providing the one-hot encoding, which indicated the currently controlled vehicle (contextual information). Then, we tested the

network without providing the contextual information, having the network infer it on the fly. The results showed that the network was able to identify the three different vehicles by performing retrospective inference based on the observed different sensorimotor dynamics. The resulting prediction error indicated a better performance when the network used its inferred encodings of the vehicles instead of using the one-hot vector values. Although the network managed to produce context values that lie in different distinguishable clusters, the inferred values were not stable, and the clusters overlapped. To this end, the structure was extended to overcome these limitations.

6.1.2 Hierarchical structure with top-down/bottom-up information flow

In order to investigate the phenomenon of event segmentation, an extended structure was built. In addition to the sensorimotor predictive layer, another layer was added to be responsible for the generation of event encodings. The flow of information between the two layers was controlled using a middle gating layer. The point of controlling the information flow was to allow it only when new information is needed. In other words, as long as the same event is going on, no need to pass new contextual information. Once an event switch happens, a new event encoding is passed from the upper contextual layer to the lower sensorimotor layer.

As an indicator of event switching, we used the surprise signal. We started by providing this signal to the gating layer at the event boundaries. When a surprise signal is provided, the gate is open, and new contextual information flow from the upper layer. Then, a new event encoding is passed on to the lower layer. The predictions can now be performed in light of the new event.

Once we start to execute a certain task, we start to plan for the next one. We simulated this act by starting to prepare for the next event few steps after starting the current one. The contextual information provided to the upper layer were changed to indicate the next event few steps after the event has started. However, this preparation only happens at the contextual layer, while the sensory layer is always performing the sensory prediction task within the current event. Here appears the role of keeping the same event encoding provided to the lower layer to perform good prediction and only allow the encoding to be updated once a switch to the next event has happened. This was achieved by controlling the information flow at the gating layer. The experimental results showed that we obtain the best prediction error when the gate is only open at event switches (cf. Table 4.2).

6.1.3 Latent event-predictive encodings

Providing the network with contextual information helps to reduce the prediction error between the sensorimotor predictions and actual observations. In this work, we showed that better prediction results can be achieved when different events have distinct encodings. These encodings are formed in the upper layer, LSTMc in the model with provided surprise signals or the event anticipation layer in SUGAR. Then, they flow down to the sensorimotor processing layer.

Having distinct event compressions helps to identify the ongoing event and use this knowledge to better predict the upcoming observation. Besides, when studying the hierarchical event cognition structure with many levels of abstraction (see the example in [44]), these distinct event compactions can well represent the different events across levels.

6.1.4 Anticipating event boundaries and the role of surprise

When it comes to the surprise signal controlling the gating functionality of the middle GRU-like layer, different sources of this signal can be investigated. We started by testing the effects of providing perfect surprise signals to the gating layer. The results showed that using a high signal to indicate an event switch can well control an efficient top-down flow of context encodings.

Then, we implemented an increasing prediction error as an indicator of the event switches in the SUGAR Model 1. We added a surprise generating unit, which calculates how far away is the current prediction error from the moving average value. A large positive difference indicates a switch. The experimental results showed that it is promising to use spikes in the prediction error to represent the switch to a new event.

In SUGAR Model 2, we assigned the task of generating surprise signals to the event boundary anticipation unit. The point here was to simulate the idea of preempting surprise signals using some information related to the event boundaries. By evaluating this model, we found that even when masked event boundary information is provided, the network is still able to make use of the provided information to predict surprise signals and use them to switch the top-down event contextual encoding just in time.

6.1.5 Comparing the outcomes of different decisions

When the decision to open the gate allowing the change in context encodings is made based on the surprise signal, this decision is not always optimal. Sometimes, the taken decision is wrong due to an incorrect interpretation of the event boundary

information, and sometimes the decision is not directly wrong, but unnecessary, possibly leading to a waste of resources, e.g., computational resources, time, etc. Thus, we enhanced the structure with a counterfactual regularization unit, in the SUGAR Model 3. In this case, the prediction errors resulting from both possible decisions are compared. If the gate opening did not improve the prediction accuracy, the opening action is punished with an additional, regularizing loss. The results showed that this counterfactual regularization helps to reach even more stable and expressive event encodings.

6.2 Limitations

Although this work managed to suggest plausible answers to the proposed questions, it has some limitations. Firstly, we always needed to provide an external signal to indicate event switches. These signals were inspired by the guidance that we receive when a certain event is about to finish. For example, when the lecturer says "For next week, we will work on ..", we expect that the current lecture is about to end. Although this information would be included in any model simulating event cognition, a more indirect information might cover the cases when the received hints are not very clear, yet we manage to predict a change. Hence, this could help to embrace the potential of the structure in predicting the not-so-straightforward predictable event switches.

The presented structure is a starting point for many enhancements. While many signals, direct and masked, were provided to the network to indicate the ongoing event and event boundaries, a promising extension to the structure would be to infer the anticipated event boundaries using direct signals flowing back up from the sensory predictions. In the same direction, the provided ground truth data to represent the currently ongoing event could be limited or changed into more indirect indicators. One possible representation of the event could be a latent vector compressed from the observations using an autoencoder. This will also be similar to the observations we receive, let it be images, sounds, smells, etc., all of them are compressed into predictive identifiers of what is currently going on.

Secondly, we used the backpropagation through time (BPTT) algorithm in the learning process. As the observations get more complex, performing the prospective and retrospective calculations in the recurrent structure can be computationally expensive. Relying more on expected information gain to modify the random behavior generation during the training process can enhance the performance. Furthermore, given how AI is inspired by neuroscience and how it tries to simulate brain functionality, we need more biologically plausible methods to update the beliefs. The heavily used backpropagation method has long been discussed to be biologically somewhat implausible [76] [40].

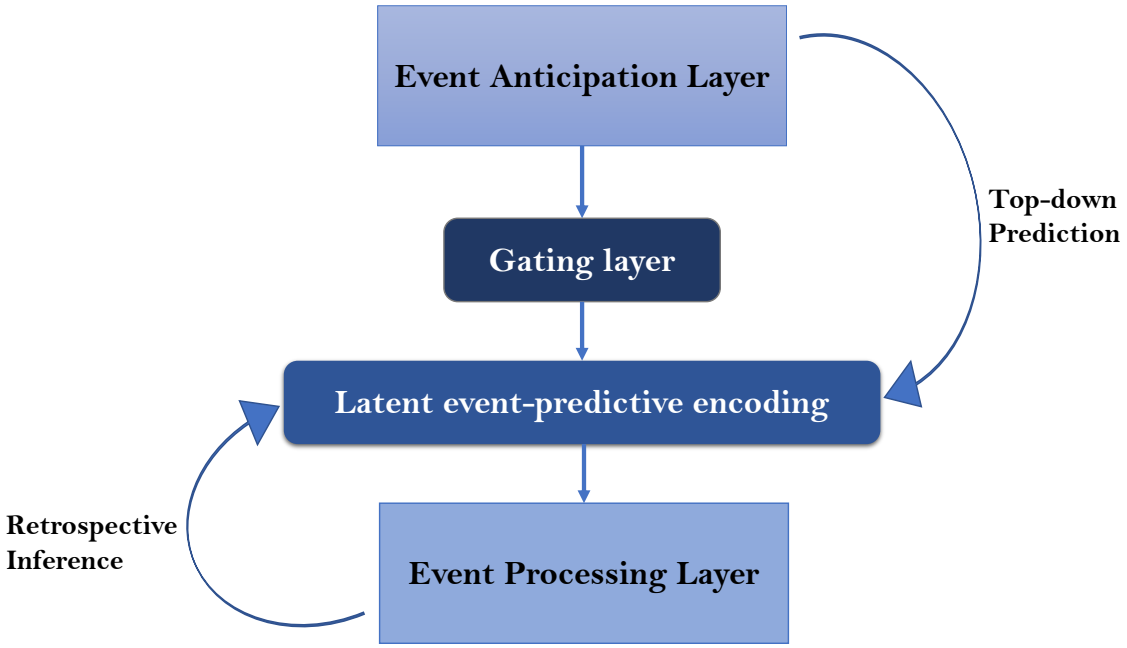


Figure 6.1: *The two processes updating the latent event-encodings when retrospective inference is implemented within the SUGAR structure. If the two updates are not well coordinated, the encodings will not be properly updated.*

Thirdly, the integration of retrospective inference with the SUGAR structure could not yet be achieved. We speculate that the reason behind this is that the same piece of information, i.e., event encoding, is being updated in two directions, as shown in Fig. 6.1. On the one hand, the performed top-down prediction drives the updates from the upper layer. On the other hand, the retrospective inference also tries to update the encodings. This situation might lead to conflicting or untimely updates of the latent even encodings. Integrating the retrospective inference would not only help to better shape the event encoding to be more coherent with the sensory experiences, but also the hidden states of the recurrent neural network to better fit the ongoing event.

Fourthly, given that the structure was tested on few examples with a limited set of events, it will be interesting to investigate the scalability of the network when applied on further problems with more events on one or more layers. Hence, the structure could be used for the processing of further sequences that can be seen as a series of events.

Finally, an important aspect in this structure that deserves to be further explored is the calculated surprise in SUGAR Model 1. In this model, direct calculation of surprise signals from the prediction error reflects the idea of event switches being initiated by large surprising increases in the prediction error. A value could be considered as surprising if it is significantly larger that the average error within a pre-determined

time window (calculated as the difference between the current error and the mean error within the time window, divided by the standard deviation of the error). In the proposed structure, we managed to practically implement this idea such that the gate would open to allow the flow of contextual information when the prediction error is suddenly increased due to the change in context. However, when the events are somehow similar, such that a change in the currently ongoing event would not lead to a highly increased error at the boundaries, the structure could not precisely identify event switches. Thus, the gate would sometimes open at unnecessary times or keep closed while information on the new context is needed. In future work, calculated surprise should be further studied to enhance its performance and explore its potentials. Possible enhancements could include new formulas of calculating the surprise based on prediction error, such that the intra-event error is clearly distinguishable compared with the inter-event error. One example may be to search for an explanation of why this would represent a surprising event [20]. Another example may be to combine the factors of event probability, outcome valence and outcome meaningfulness [41].

6.3 Conclusions

Hierarchical representation of knowledge in the brain The established research in a wide range of fields, including cognitive science, neuroscience and psychology, has shown that our brain can be characterized as a generative, predictive system. Approximating Bayesian inference, the brain compares sensations, which flow in a bottom-up direction, with predictions transmitted in a top-down manner [39]. The upper and lower levels in these directions indicate the more abstract representation of knowledge and the more sensational representation, respectively. This hierarchy has also been anatomically discussed in the early 90s [56], investigating how cortico-cortical connections update raw low-level data and high-level abstract data. Although this hierarchical representation of the information flow in the brain has been discussed in many works, it is still unclear how many levels there truly are and how the information is partitioned within a level

This work investigated how to segment and compact contextual encodings, for which we need hierarchies. A very good example of the usage of hierarchies is the divide-and-conquer method. It is widely used not only in computer algorithms but in everyday plans. As the name reveals, it is based on the idea of dividing a problem into many subproblems, forming a hierarchical representation of the main problem in the form of smaller problems with different levels of difficulty. As long as you can still simplify the problem by dividing it into further subproblems, new levels will be added to the hierarchy. In our brain, when we want to understand an idea, a representation of the idea is built. Moving from the very abstract representation of

the idea, simplifying it into a more materialized form, until we reach the very tangible sensorimotor data.

The Free Energy Principle One formulation of the "Bayesian Brain" is the "Free Energy Principle" [25] [27] [39], which was proposed by the neuroscientist Karl Friston. The notion of free energy comes from statistical physics, and is used to ease the handling of complex integration problems. This principle starts from the point that any biological system always strives to keep a homeostatic state. As a result, it tries to move among a set of predefined states that are familiar and avoid any surprising actions. But how can one actually avoid any surprises? This would mean taking into account anything that might happen and prepare the suitable reactions to each and every situation, which is not realistic. Instead, the system can try to minimize an upper bound on surprise, which is the proposed free energy. In mathematical forms, the free energy is defined as

$$F = -\ln p(\tilde{y}) + D(q(\vartheta;\lambda)||p(\vartheta|\tilde{y})) \tag{6.1}$$

where $\tilde{y}$ represents the sensory observations by the system, ϑ denotes the environment state (thus, the causes of the sensory inputs), and λ is the system state. Simply explained, the free energy would then be equal to the surprise signal plus the difference between the predictive posterior (q) and future posterior (p) beliefs. Hence, minimizing the free energy means that the posterior density will be closer to the prior one, so the system can infer the causes of its sensory input.

Despite the many follow-up works on the free energy and related aspects [24] [28], the free energy principle of the brain still suffers from being too difficult and too general as well. Although it discusses surprising events and the role of active inference, it does not explain the specifics of event cognition. Hence, it has been criticized for offering an incomplete theory of cognition [16] [73]. Furthermore, it is not clear yet how can the presented equations be translated into practical applications.

The closest form of application has been in starting from the surprise point. In order to update our beliefs to be more accurate compared with the actual observations from the surrounding environment, our brain compares them, and the resulting error is used to update our ideas. When the error is significantly large, then what happened is more than small discrepancies between what we believe and what we observed, it means that the whole context of what is going on around us is now different. This large error is then called a "Surprise" signal. Surprise signals are avoided, or at least minimized, by doing the following [39]:

- Adapting the actions that will affect the environment.
- Updating the priors based on the acquired experience.

- Optimizing the generative model of the world in the brain.

Learning from children If building perceptive and cognitive systems is a main task in the field of AI, why not learning from the best example of building knowledge over time, that is, learning from children? As Turing previously asked, why not trying to produce a system that simulates the child's mind? [79, p. 456] Observing how knowledge is accumulated in children is a great inspiration for AI systems. We can explain why a human being is able to learn different tasks by only few samples (while an AI system might need millions of them) by the accumulated knowledge and deep experience collected across the years. This makes it easy for human beings to apply what they learned in one task on other similar tasks; a method already applied in AI under the name of "Transfer Learning" [37], or learn a task using few examples as in zero- or one-shot learning [34].

An important example is the development of goal-directed gaze in infants. Starting from about six months onward, infants start to move their gaze from a moving action to the goal of this action before it ends. Gumbsch and colleagues [32] studied this behavior, and simulated it in terms of a generative event-predictive model, which performs active inference. As a result, the system begins to fixate the goal briefly after the movement of the agent started it, inferring both the reaching event and the need to look at the goal to minimize uncertainty about the upcoming interaction.

In 2019, the scientist Gül Deniz Salalı, who does field research in Congo studying Mbendjele BaYaka hunter-gatherers wrote an interesting article comparing their learning behavior to the unsupervised learning in AI. In both cases, the agent (being a human-gatherer or an AI model) learns by exploring without having a predefined goal. She reported that learning by being taught accounted only for 6% of the episodes observed by the team, while the remaining learning was based on free exploration and learning from others [67]. This could be reflected in event-predictive systems by transfer-learning the event-predictive encodings learned in one system to another. Hence, enriching the encoding reservoir of the system faster. By combining these observations, advances in artificial cognitive models may indeed be inspired from human cognitive processes, especially as it is being developed in children.

Compositionality Simply put, compositionality refers to the property of having new representations constructed from a limited set of primitive elements [48]. It plays a very important role in cognition, since it enables us to use the existing information, especially in an abstract form, to expand the knowledge base. A very good example of representations that are compositional is handwriting. A previous work of Lake and colleagues [46] showed how we combine sub-parts to create letters; the same sub-parts can be used to generate a large number of final representations. Fig. 6.2 shows how primitive elements can constitute sub-parts, which can then be

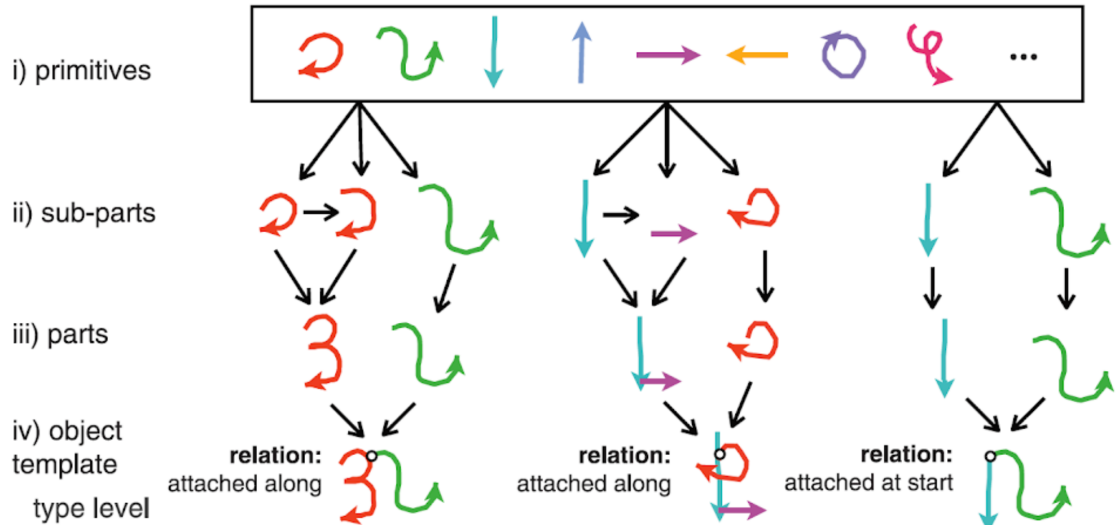


Figure 6.2: *An example of creating different handwriting objects from the same sub-parts. Figure form [46]*

combined to create larger parts, until a full object is created.

Compositionality can be observed in compressed, event-predictive encodings. In a work from 2019 [58], modular RNN modules were used to learn compositional dynamics and infer the subdynamics. More recently, the formation of such compositional encodings has been shown to help solve the Omniglot challenge [19]. The Omniglot challenge aims at developing algorithms that can act on a human level, by constructing one model that can perform many concept learning tasks; examples include One-Shot learning for handwritten characters [47]. In [19], a generative RNN was used to solve the Omniglot challenge by learning compositional representations, which could be reassembled into new character trajectories. They showed that compositionality can be introduced as an inductive bias into a simple LSTM network, enhancing its ability to solve different tasks.

With the many levels of abstraction in an event-cognitive schema come many levels of complexity in the respective event-predictive encodings. Encodings of higher-level events might be composed of the encodings of lower-level events. Indeed, our experiments have shown that some compositionality properties could be observed in the developing event compressions. For example, Fig. 5.9 showed how the event encodings of similar events had similar sub-parts and characteristics. Future work should thus further explore and foster the emergence of compositional event-predictive structures.

Learning to learn Establishing cognitive systems is not only about the learning results, but also about the learning process itself. In order to achieve human-level

AI, two main components should be included: meta-learning and meta-reasoning. These two components and the interaction between them were beautifully described in the work of Griffiths and colleagues in 2019 [31]. They discussed how meta-reasoning, that is, the efficient deployment of computational resources, and meta-learning, that is, the efficient usage of cognitive resources, could be combined to explain the ability in humans to learn a lot from very little data. This has remained an important research topic in machine learning, as researchers try to develop models that learn successfully with the least amount of training data. Examples of such trials include zero-, one- and few-shot learning models [42] [83]. Similar to meta-learning, meta-cognition can be seen as the control and awareness of one's cognitive functions [3].

Earlier in 1987, Schmidhuber, who later co-created the extensively used LSTM units along with Hochreiter [38], presented his ideas on learning how to learn. In his thesis [68], he explained how recurrent neural networks can learn to run different learning algorithms on themselves, what is called being "self-referential". Then, it would be reasonable to consider the generalization and abstraction abilities of the model essential for combining different algorithms and tasks. In the event-compression model presented in this work, the constructed event-encodings allow the model to generalize among tasks as required. The abstraction level achieved through the higher level encodings give the model a self-referential property.

Conscious machines and the role of cognition As more advances are being achieved in computational cognitive science, one might think: how far are we from cognitive machines? As a matter of fact, combining all the required units that are similar to what we have in the human body will not create a cognitive being. What creates our cognitive abilities? Before diving into cognition, another point that needs answering is how cognition is related to consciousness. Does the latter have a cognitive nature? Or are these two separated? [8]. With the emergence of new technologies to study the brain in even more detail, including functional magnetic resonance imaging (fMRI) and electroencephalography (EEG), more research approached the origin of consciousness. The difficulty starts from the task of defining consciousness. Many theories on human consciousness [17] [52] looked at it as a form of information processing, continuously processing the sensory information and making predictions. These predictions are compared with the observations and updated accordingly. Recently, Kotchoubey [43] defined consciousness as a behaviour, which controls movements. An opinion by van Gaal and colleagues [81] discussed how cognitive control, which is a general term for how our goals and plans affect our behaviour in a flexible way, is affected by conscious and unconscious information. The functions related to cognitive control, including responding to stimuli and switching tasks, have been linked with prefrontal areas in the cortex, thus seemingly encoding

top-down cognitive control. This initiates the question of what role does this top-down control play in event cognition, being itself updated in a top-down/bottom-up scheme of information flow.

Top-Down, goal-directed control In speech perception, top-down processing relies on language experience to realize meaning, which is combined with bottom-up processing, mainly depending on instant auditory input [70]. Top-down control has been described by Melloni et al. [53] as the way our inner goal affects our selection of the stimulus. They presented the example of looking for a pencil. The sensory observations would represent the bottom-up control, while the top-down control would be our endogenous control to meet our goals.

It is believed that event cognition is an essential cognitive aspect to achieve a robust perception of the world. All the observations that we make about what is going on around us contain different forms of data used to update our internal beliefs. Hence, when there is a sudden change in what we observe compared with what we have been observing for a while, we realize that an event has ended and another has started. This process of identifying event boundaries and the formation of distinct event-predictive encodings in the brain has been widely discussed in cognitive neuroscience. However, when it comes to the modeling of this process, there certainly still remains rather huge room for improvement.

This work has developed a structure that can be used to compress latent event-predictive encodings in different tasks. One usage of this structure can be in decision making tasks, such that alternative options are evaluated to decide on the best action to perform next within an overall policy. Besides, the information flow is controlled at the gating layer, which allows the use of this structure in event-triggered learning, achieving efficient information flow. Further usages would be in planning when the states are partially observable, in reasoning and in finding fast and generalizing solutions in reinforcement learning problems [34]

6.4 Future directions

As difficult as it is to picture how future artificial cognitive models would look like, one can plan developmental lines that emerged from the achievements made in this thesis, limitations encountered during the work and enhancements inspired from the reviewed literature.

Although the property of predictive inference is investigated in event-predictive models, the power of inference can still be more deeply exploited. Our brain relies on active inference to achieve precise perceptions, but it is not very clear how the information can be indirectly encoded and observed to predict future events. This point can be further explored in future research, by providing input information to

the model in more indirect ways, such that more inference is performed and more coherent updates of the predictive encodings are achieved. This does not only include prospective inference but also retrospective inference. Combining both would help in problems where the context is not provided, and yet a task that relies on identifying the context should be performed. An example would include a virtual reality environment in which the agent needs to use a tool to move some cubes. However, the agent does not know the characteristics of the tool to be used. In a recent work, retrospective inference has indeed been used to tune parametric bias neurons in order to solve the binding challenge [66].

The surprise-gated top-down layer presented in this work can be integrated within other structures, in which an efficient and controlled information flow is needed. It can be an essential part of any model that can make use of compositional event-predictive encodings. An important example comes from language learning, where a structure that forms such encodings called 'LEARNA' has been recently proposed to infer events as an interpretation of ambiguous utterances [75]. The surprise-gated layer can also be used in a form of reward shaping in reinforcement learning tasks. The flow of information can be controlled to determine the switching between exploration and exploitation, a challenging point in reinforcement learning. The recent work of Gumbsch et al. [34] has actually presented a novel recurrent architecture capable of maintaining stable, sparsely changing latent states by incorporating such an inductive bias.

The modular nature of the presented structure makes it practical to edit the details. This means that varied input/output/bias combinations could be used to determine which of them has the most effect on the performance and which could be ignored or masked. Besides, the inter-layers connections could be edited. For example, residual connections could be added to the input of the contextual and/or sensorimotor layers. It would be interesting to see the effect of such connections on the gradient, and thus on the formed encodings.

The problems used in the evaluation of the model are only examples of how the structure can be used. Further studies can use the structure in varied problems in which hierarchical surprise-gated compression of event-predictive encodings can be useful. An example could be controlling an unidentified moving entity across different environments. Additionally, although two main levels of abstraction were used here, the resulting encodings can be used as the representational basis to conduct conceptual, event-predictive reasoning and planning on multiple levels of abstraction. Furthermore, it would be interesting to see how can the resulting compositional encodings be linked with language encodings. An example would be a scenario in which two agents communicate and form language encodings from auditory observations.

As a final word, working in the field of neuro-cognitive modeling to try to under-

stand the brain and simulate the functionalities it carries within may be driven by the desire to achieve human-level AI. When would we be able to build highly intelligent agents that are as cognitive as human beings? Event-cognition constitutes a key aspect to be considered here. A human-level AI should go beyond the processing of individual observations to realizing how events are unfolding in the surrounding environment. There is now more than ever a trend in research labs to form interdisciplinary groups of researchers. Combining philosophers, psychologists, computer scientists, computational neuroscientists and others, the advancements toward super AI are expected to be enormous. While the futurist Ray Kurzweil predicted that the tome when AI systems will pass the Turing test and reach the level of human intelligence or more will be as early as 2029 [45], the historian Yuval Harari thinks that not even in 200 years we could think of merging our mind with computers [35].

Away from predictions, this work showed the importance of the hierarchical flow of event-predictive encodings in minimizing prediction error compared with single level structures. Besides, it showed the emergence of compositional encodings: encodings representing similar problems were composed of similar sub-units. This is very important for transfer learning and domain adaptation. The combination of top-down and bottom-up signals seem to be essential in artificial cognitive systems to act in a goal-directed manner. The gathered observations support the belief that hierarchical event-predictive models will constitute an important unit in more advanced and complex artificial cognitive systems with an increasing level of human-like or even super-human intelligence.

Abbreviations

AI	Artificial Intelligence
ADAM	ADAptive Moment estimation
ANN	Artificial Neural Network
BPTT	BackPropagation Through Time
EB	Event Boundary information
EEG	ElectroEncephaloGraphy
CFR	CounterFactual Regularization
CI	Contextual Information
fMRI	functional Magnetic Resonance Imaging
GRU	Gated Recurrent Unit
LSTM	Long Short-Term Memory
LSTMc	LSTM contextual
LSTMf	LSTM functional
PCA	Principal Component Analysis
REPRISE	REtrospective and PRospective Inference SchEma
RI	Retrospective Inference
RL	Reinforcement Learning
RNN	Recurrent Neural Network
SUGAR	SUrprise-GAted Recurrent neural network

Glossary

Abstraction: Moving towards a more conceptualized idea instead of a concrete specific instance.

Context: The overall conditions that define the occurrence of an event.

Event: a segment of time in a given location at which a certain activity happens that can be observed to have a beginning and an end.

Event boundary: The time point at which an event ends and the next starts.

Event-predictive encodings: a code that represents an event; it can be used to predict the upcoming observation in light of the corresponding event.

Hierarchical structure: A multi-level model in which each layer is more general and abstract than the layer below it. The information flows between the layers subsequently.

Inductive bias: the assumptions affecting the output that will result from processing an input not previously encountered.

Meta-learning: observing the learning algorithms and developing better ones that can learn quickly and efficiently.

Model: A structural representation of an object or an event.

Modularization: Reorganizing a model into submodels that can have shared inputs/outputs but can also have different connections to achieve separate tasks.

Prospective: looking ahead into the future time steps to update the current coefficients as to decrease the expected error.

Retrospective: looking back into the past time steps to update the current coeffi-

cients as to decrease the expected error.

Bibliography

[1] Toshitake Asabuki, Naoki Hiratani, and Tomoki Fukai. Interactive reservoir computing for chunking information streams. *PLOS Computational Biology*, 14(10):e1006400, October 2018.

[2] B. Harvey B. Klein and S. Dumoulin. Attraction of position preference by spatial attention throughout human visual cortex. *Neuron*, 84:227–237, 2014.

[3] L. Baker. Metacognition. *International Encyclopedia of Education*, page 204–210, 2010.

[4] J. Balaguer, H. Spiers, D. Hassabis, and C. Summerfield. Neural mechanisms of hierarchical planning in a virtual subway network. *Neuron*, 90:893–903, 2016.

[5] C. Baldassano, J. Chen, A. Zadbood, J. W. Pillow, U. Hasson, and K. A. Norman. Discovering event structure in continuous narrative perception and memory. *Neuron*, 95:709–721, 2017.

[6] D. A. Baldwin and J. E. Kosie. How does the mind render streaming experience as events? *Topics in Cognitive Science*, 2020.

[7] Matthew Botvinick and Marc Toussaint. Planning as inference. *Trends in Cognitive Sciences*, 16(10):485–488, 2012.

[8] Richard Brown. Consciousness doesn't overflow cognition. *Frontiers in Psychology*, 5, Dec 2014.

[9] M. Butz and E. Kutter. *How the mind comes into being: Introducing cognitive science from a functional and computational perspective*. Oxford University Press, 2016.

[10] M. V. Butz, A. Achimova, D. Bilkey, and A. Knott. Editors' review and introduction: Topic event-predictive cognition: From sensorimotor via conceptual to language-based structures and processes. *Topics in Cognitive Science*, 2020.

[11] Martin V. Butz. Towards a unified sub-symbolic computational theory of cognition. *Frontiers in Psychology*, 7(925), 2016.

[12] Martin V. Butz, David Bilkey, Alistair Knott, and Sebastian Otte. Reprise: A retrospective and prospective inference scheme. *Proceedings of the 40th Annual Meeting of the Cognitive Science Society*, 2018, in press.

[13] M.V. Butz. Towards strong ai. *KI - Kuenstliche Intelligenz*, 2021.

[14] Kyunghyun Cho, Bart van Merriënboer, Çağlar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (ENMLP)*, pages 1724–1734, Doha, Qatar, 2014.

[15] Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555, 2014.

[16] Andy Clark. *Embodied Prediction*. Frontiers, 2015.

[17] D. C. Dennett. *Consciousness explained*. Back bay books. Little, Brown, 1. paperback ed edition, 1991.

[18] W. Dunin-Barkowski. Editorial:toward and beyond human-level ai. *Front. Neurorobot*, 2020.

[19] Sarah Fabi, Sebastian Otte, and Martin V. Butz. Fostering compositionality in latent, generative encodings to solve the omniglot challenge. *Lecture Notes in Computer Science*, page 525–536, 2021.

[20] Meadhbh I. Foster and Mark T. Keane. Predicting surprise judgments from explanation graphs. In *International Conference on Cognitive Modelling*, 2015.

[21] Nicholas T. Franklin, Kenneth A. Norman, Charan Ranganath, Jeffrey M. Zacks, and Samuel J. Gershman. Structured event memory: a neuro-symbolic model of event cognition. *bioRxiv*, page 541607, February 2019.

[22] K. Friston. The free-energy, principle: a rough guide to the brain? *Trends Cogn. Sci.*, 13:293–301, 2009.

[23] K. Friston and K. Stephan. Free-energy and the brain. *Synthese*, 159:417–458, 2007.

[24] Karl Friston. Hierarchical models in the brain. *PLoS Computational Biology*, 4(11):e1000211, Nov 2008.

[25] Karl Friston. The free-energy principle: a unified brain theory? *Nature Reviews Neuroscience*, 11(2):127–138, Feb 2010.

[26] Karl Friston. The history of the future of the bayesian brain. *NeuroImage*, 62(2):1230–1233, 2012. 20 YEARS OF fMRI.

[27] Karl Friston, James Kilner, and Lee Harrison. A free energy principle for the brain. *Journal of Physiology-Paris*, 100(1–3):70–87, Jul 2006.

[28] Karl Friston, Francesco Rigoli, Dimitri Ognibene, Christoph Mathys, Thomas Fitzgerald, and Giovanni Pezzulo. Active inference and epistemic value. *Cognitive Neuroscience*, 6(4):187–214, Oct 2015.

[29] Gary H. Glover. Overview of functional magnetic resonance imaging. *Neurosurg Clin N Am*, 22(3):133–vii, 2011.

[30] Y. C. Goh, X. Q. Cai, W. Theseira, G. Ko, and K. A. Khor. Evaluating human versus machine learning performance in classifying research abstracts. *Scientometrics*, 125:1197–1212, 2020.

[31] Thomas L Griffiths, Frederick Callaway, Michael B Chang, Erin Grant, Paul M Krueger, and Falk Lieder. Doing more with less: meta-reasoning and meta-learning in humans and machines. *Current Opinion in Behavioral Sciences*, 29:24–30, 2019.

[32] Christian Gumbsch, Maurits Adam, Birgit Elsner, and Martin V. Butz. Emergent goal-anticipatory gaze in infants via event-predictive learning and inference. *Cognitive Science*, 45(8), 2021.

[33] Christian Gumbsch, Martin V. Butz, and Georg Martius. Autonomous identification and goal-directed invocation of event-predictive behavioral primitives. *IEEE Transactions on Cognitive and Developmental Systems*, 13(2):298–311, 2021.

[34] Christian Gumbsch, Martin V. Butz, and Georg Martius. Sparsely changing latent states for prediction and planning in partially observable domains. In *Advances in Neural Information Processing Systems (NeurIPS 2021)*, December 2021. to appear.

[35] Yuval Noah Harari. *Homo deus: A brief history of Tomorrow*. Vintage Publishing, 2017.

[36] P. Hart and A. Knoll. Using counterfactual reasoning and reinforcement learning for decision-making in autonomous driving. *arXiv preprint arXiv:2003.11919*, 2020.

[37] D. Hassabis, D. Kumaran, C. Summerfield, and M. Botvinick. Neuroscience-inspired artificial intelligence. *Neuron*, 95:245–258, 2017.

[38] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[39] Jeremy Holmes and Tobias Nolte. "surprise" and the bayesian brain: Implications for psychotherapy theory and practice. *Frontiers in Psychology*, 10:592, Mar 2019.

[40] Bernd Illing, Wulfram Gerstner, and Johanni Brea. Biologically plausible deep learning — but how far can we go with shallow networks? *Neural Networks*, 118:90–101, 2019.

[41] James Juergensen, Joseph S. Weaver, Kevin J. Burns, Peter E. Knutson, Jennifer L. Butler, and Heath A. Demaree. Surprise is predicted by event probability, outcome valence, outcome meaningfulness, and gender. *Motivation and Emotion*, 38(2):297–304, 2013.

[42] Suvarna Kadam and Vinay Vaidya. Review and analysis of zero, one and few shot learning approaches. *Advances in Intelligent Systems and Computing*, page 100–112, 2019.

[43] Boris Kotchoubey. Human consciousness: Where is it from and what is it for. *Frontiers in Psychology*, 9:567, Apr 2018.

[44] G. R. Kuperberg. Tea with milk? a hierarchical generative framework of sequential event comprehension. *Topics in Cognitive Science*, 2020.

[45] Ray Kurzweil. *The singularity is near: When humans transcend biology*. Duckworth, 2018.

[46] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

[47] Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. The omniglot challenge: A 3-year progress report. *Current Opinion in Behavioral Sciences*, 29:97–104, 2019.

[48] Brenden M. Lake, Tomer D. Ullman, Joshua B. Tenenbaum, and Samuel J. Gershman. Building machines that learn and think like people. *Behavioral and Brain Sciences*, 2017.

[49] S. Legg. A collection of definitions of intelligence. *ArXiv e-prints*, 0706.3639, 2007.

[50] Lester C. Loschky, Adam M. Larson, Joseph P. Magliano, and Tim J. Smith. What would jaws do? the tyranny of film and the relationship between gaze and higher-level narrative film comprehension. *PloS One*, 10(11):e0142474, 2015.

[51] G. Buracas M. Saenz and G. Boynton. Global effects of feature-based attention in human visual cortex. *Nature Neuroscience*, 5:631–632, 2002.

[52] Tiago V. Maia and Axel Cleeremans. Consciousness: converging insights from connectionist modeling and neuroscience. *Trends in Cognitive Sciences*, 9(8):397–404, Aug 2005.

[53] Lucia Melloni, Sara van Leeuwen, Arjen Alink, and Notger G. Mueller. Interaction between bottom-up saliency and top-down control: How saliency maps are created in the human brain. *Cerebral Cortex*, 22(12):2943–2952, 2012.

[54] K. Metcalf and D. Leake. Modeling unsupervised event segmentation: Learning event boundaries from prediction errors. In *Proceedings of the 39th Annual Conference of the Cognitive Science Society*, pages 2717–2722, 2017.

[55] S. Mittal, A. Lamb, A. Goyal, V. Voleti, M. Shanahan, G. Lajoie, N. Mozer, and Y. Bengio. Learning to combine top-down and bottom-up signals in recurrent neural networks with attention over modules. In *Proceedings of the 37th International Conference on Machine Learning*, pages 6972–6986. PMLR, 2020.

[56] D. Mumford. On the computational architecture of the neocortex: I. the role of the thalamo-cortical loop. *Biological Cybernetics*, 65(2):135–145, Jun 1991.

[57] J. Nassi and E. Callaway. Parallel processing strategies of the primate visual system. *Nature Review Neuroscience*, 10:360–372, 2009.

[58] Sebastian Otte, Patricia Rubisch, and Martin V. Butz. Gradient-based learning of compositional dynamics with modular rnns. *Artificial Neural Networks and Machine Learning – ICANN 2019: Theoretical Neural Computation*, page 484–496, 2019.

[59] J. Pearl. *The limitations of opaque learning machines. In: J. Brockman (ed.) Possible minds: 25 ways of looking at AI, chap. 2, pp. 13–19*. Penguin Press, New York, 2020.

[60] Leonid Perlovsky. Language and cognition. *Neural Networks*, 22(3):247–257, 2009. Goal-Directed Neural Systems.

[61] Kyle A. Pettijohn, Alexis N. Thompson, Andrea K. Tamplin, Sabine A. Krawietz, and Gabriel A. Radvansky. Event boundaries and memory improvement. *Cognition*, 148:136–144, March 2016.

[62] G. A. Radvansky and J. M. Zacks. *Event cognition*. Oxford University Press, 2014.

[63] Gabriel A. Radvansky and Jeffrey M. Zacks. Event perception. *WIREs Cognitive Science*, 2(6):608–620, 2011.

[64] J. R. Reynolds, J. M. Zacks, and T. S. Braver. A computational model of event segmentation from perceptual prediction. *Cognitive Science*, 31:613–643, 2007.

[65] Jeremy R. Reynolds, Jeffrey M. Zacks, and Todd S. Braver. A computational model of event segmentation from perceptual prediction. *Cognitive Science*, 31(4):613–643, July 2007.

[66] Mahdi Sadeghi, Fabian Schrodt, Sebastian Otte, and Martin V. Butz. Binding and perspective taking as inference in a generative neural network model. *Lecture Notes in Computer Science*, page 3–14, 2021.

[67] Gul Deniz Salali, Nikhil Chaudhary, Jairo Bouer, James Thompson, Lucio Vinicius, and Andrea Bamberg Migliano. Development of social learning and play in bayaka hunter-gatherers of congo. *Scientific Reports*, Jul 2019.

[68] Jürgen Schmidhuber. *Evolutionary principles in self-referential learning, or on learning how to learn: The meta-meta-... hook*. Diplomarbeit, Technische Universität München, München, 1987.

[69] Ana Serrano, Vincent Sitzmann, Jaime Ruiz-Borau, Gordon Wetzstein, Diego Gutierrez, and Belen Masia. Movie editing and cognitive event segmentation in virtual reality video. *ACM Transactions on Graphics*, 36(4):1–12, July 2017.

[70] Lan Shuai and Tao Gong. Temporal relation between top-down and bottom-up processing in lexical tone perception. *Frontiers in Behavioral Neuroscience*, 8, 2014.

[71] M. Sigman and S. Dehaene. Brain mechanisms of serial and parallel processing during dual-task performance. *The Journal of Neuroscience*, 28:7585–7598, 2008.

[72] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. vandenDriessche, J. Schrit-twieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach,

K. Kavukcuoglu, T. Graepel, and D. Hassabis. Mastering the game of go with deep neural networks and tree search. *Nature*, 529:484–489, 2016.

[73] Andrew Sims. A problem of scope for the free energy principle as a theory of cognition. *Philosophical Psychology*, 29(7):967–980, 2016.

[74] Friedrich Solowjow, Dominik Baumann, Jochen Garcke, and Sebastian Trimpe. Event-triggered learning for resource-efficient networked control. *2018 Annual American Control Conference (ACC)*, pages 6506–6512, June 2018. arXiv: 1803.01802.

[75] Christian Stegemann-Philipps and Martin V. Butz. Learn it first: Grounding language in compositional event-predictive encodings. *2021 IEEE International Conference on Development and Learning (ICDL)*, 2021.

[76] Stork. Is backpropagation biologically plausible? *International Joint Conference on Neural Networks*, 1989.

[77] R. Sutton and A. Barto. *Reinforcement Learning, second edition: An Introduction*. MIT Press, Cambridge, MA, 2018.

[78] P. Thegard. *Mind: Introduction to cognitive science*. MIT Press, 2005.

[79] A. M. TURING. I.—computing machinery and intelligence. *Mind*, LIX(236):433–460, 1950.

[80] E. Uenal and A. Papafragou. Relations between language and cognition: Evidentiality and sources of knowledge. *Topics in Cognitive Science*, 2:115–135, 2018.

[81] Simon van Gaal, Floris P De Lange, and Michael X Cohen. The role of consciousness in cognitive control and decision making. *Frontiers*, Jan 2012.

[82] G. R. VandenBos. Apa dictionary of psychology. American Psychological Association, 2007.

[83] Yongqin Xian, Christoph H. Lampert, Bernt Schiele, and Zeynep Akata. Zero-shot learning—a comprehensive evaluation of the good, the bad and the ugly. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(9):2251–2265, 2019.

[84] J. M. Zacks and B. Tversky. Event structure in perception and conception. *Psychological Bulletin*, 127(1):3–21, 2001.

[85] Jeffrey M. Zacks, Nicole K. Speer, Khena M. Swallow, Todd S. Braver, and Jeremy R. Reynolds. Event perception: A mind-brain perspective. *Psychological Bulletin*, 133(2):273–293, 2007.

[86] Jeffrey M. Zacks and Khena M. Swallow. Event segmentation. *Current directions in psychological science*, 16(2):80–84, April 2007.

[87] Jiaying Zhao, Ulrike Hahn, and Daniel Osherson. Perception and identification of random events. *Journal of Experimental Psychology. Human Perception and Performance*, 40(4):1358–1371, August 2014.

[88] Q. Zhu, W. Zhang, T. Liu, and W. Y. Wang. ounterfactual off-policy training for neural dialogue generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 3438–3448, 2020.