Improving the Out-of-Distribution Generalization of Deep 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

Evgenia Rusak

aus Ufa/ Russland

Tübingen

2025

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

Tag der mündlichen Qualifikation:	10.08.2026
Dekan:	Prof. Dr. Thilo Stehle
1. Berichterstatter:	Prof. Dr. Oliver Bringmann
2. Berichterstatterin:	Prof. Dr. Hilde Kühne

Acknowledgements

Advisors First, I would like to thank Oliver Bringmann for accepting me as a PhD student, offering his guidance and support throughout all these years and allowing me the scientific freedom to find my own path. I would also like to express my deepest gratitude to Wieland Brendel for helping me become the researcher I am today. I have learned so much from our meetings which were as educative as they were fun. Further, I would like to thank Matthias Bethge in whose lab I have spent a considerable amount of time for building a collaborative and friendly scientific ecosystem where students can thrive and develop. I would like to acknowledge my TAC committee members, including Wieland Brendel, Oliver Bringmann, Matthias Bethge and Jörg Stückler. Their guidance, assurance, and feedback have been invaluable on my journey towards my PhD. I am very grateful to Ari Morcos and Kamalika Chaudhuri for hosting me at FAIR during my research internship for their guidance and support. I am particularly grateful to Ari for rekindling my appreciation for Brandon Sanderson's books. I started my career in computer science at the FZI in Karlsruhe when I switched topics from Physics to Machine Learning. I am grateful to Sebastian Reiter who was my mentor at the FZI for supporting and advising me. I would like to express my gratitude to Hilde Kühne for agreeing to be the second reviewer for my thesis and to Peter Gehler for being part of the thesis defense committee.

Co-authors I would like to thank all of my co-authors with whom I was fortunate to work with, in alphabetical order: Alexander S. Ecker, Attila Juhos, Benjamin Mitzkus, Claudio Michaelis, George Pachitariu, Julian Bitterwolf, Luisa Eck, Lukas Schott, Matthias Bethge, Max Wolff, Oliver Bringmann, Patrik Reizinger, Peter Gehler, Prasanna Mayilvahanan, Robert Geirhos, Roland S. Zimmermann, Steffen Schneider, Thomas Klein, Thaddäus Wiedemer and Wieland Brendel. In particular, I would like to thank Roland S. Zimmermann for his tremendous help throughout many joint projects. Due to his versatility, Roland has been able to assist me in so many things: From building singularity containers to asking exactly the right questions to jump-start a project from a dead-end. I am so happy we became friends over the years and I will miss our walks and cooking sessions. I am very grateful to Lukas Schott, without whom my very first project would not have been possible. I have learned so much from his coding skills which I could successfully apply in my subsequent projects. I would like to thank Patrik Reizinger for his unwinding enthusiasm and excitement as well as his general positive attitude which kept me afloat during the tough parts of our project when the progress has been slow. I am very grateful to Attila Juhos for giving me access to his tremendous math knowledge and structured thinking. I have also greatly enjoyed our tea, coffee and chess sessions as well as being roomies during ICLR. I would like to thank Prasanna Mayilvahanan for his humor, kindness and helpfulness. I would like to thank Thaddäus Wiedemer for interesting and helpful discussions.

Colleagues The time of my PhD have been the best years of my life. I attribute this first and foremost to the collaborative atmosphere at the labs which has made me feel at home. I would like to thank all of my wonderful colleagues from Wieland's, Oliver's, Matthias' and Sebastian's labs with whom I was fortunate enough to work with. I have immensely enjoyed the ski retreats at Oliver's lab. This is an amazing tradition which has persisted for over thirty years and which I hope will endure at least another thirty. The skiing lessons given by Konstantin Lübeck were really cool! I would like to thank Georg Volk and Joscha Benz for being awesome office mates. I am very grateful to Georg for taking the time to diagnose the issue and fix my laptop when a RAM stick broke.

Institutions I greatly appreciate the opportunity for having been part of an inspiring and collaborative environment at the University of Tübingen, the Max Planck Institute for Intelligent Systems, the Tübingen AI Center and the International Max Planck Research School for Intelligent Systems (IMPRS-IS). I am grateful to our IT admins, in particular, Kristina Kapanova, for building and maintaining our IT infrastructure as well as being extremely helpful in solving cluster related problems.

Friends, Family, Partner I would like to thank my dear friends for offering support and distraction in hard times. I am grateful to my and my husbands's family for their love and support. To my wonderful husband, Julian: I am infinitely grateful for your presence by my side throughout this journey. Thank you for being the one who encouraged me to switch from Physics to Machine Learning years ago, opening up this new and exciting world. Your never-ending support and humor were essential to keep me motivated and brightened my days even when the research was slow.

Abstract

Machine learning systems have become ubiquitous in everyday life: Artificial Intelligence (AI)–based models recommend music, generate code, create paintings, and even compose poetry. Unlike traditional programs that follow hand-crafted rules, these systems learn from data. The distribution of the training data dictates what features the model attends to and thus strongly shapes its decision-making process. For example, if an autonomous vehicle were trained only on highway scenes, it would struggle to navigate city streets—and vice versa. While collecting a dataset that spans both domains is feasible, it is impossible to capture every possible distribution shift. Can a self-driving car handle unusual weather conditions absent from its training data? More broadly, how can we quantify and improve the model's *generalization capabilities* beyond the data it has been trained on?

Addressing this question is essential for creating AI systems that are both safe and reliable for real-world deployment. For this reason, this thesis shows steps to better understand how Deep Neural Networks behave when they encounter distribution shifts as well as how their performance can be improved. Depending on the application, the trained system can be expected to be robust to different kinds of shifts out of the box or, if the distribution shift persists over a long time, it can be allowed to adapt to it. Both scenarios are covered in this thesis.

In the first setting, the goal is to learn general representations that transfer across diverse situations. One approach explored in this work is augmenting training data with adversarially generated noise patterns designed to maximally confuse a classifier. Jointly training the noise generator and the classifier yields models that are more resilient to a variety of image corruptions. Extending beyond classification, the robustness of different object detection models under corruptions is benchmarked and improved. The discussion then shifts to a different paradigm: Contrastive Learning (CL), where models are trained without labels. Prior research has established a theoretical connection between CL and identifiability, showing that under certain assumptions, CL effectively inverts the data-generating process. This thesis relaxes those assumptions, extends the theoretical guarantees, and verifies the findings empirically.

In the second setting, where distribution shifts persist over multiple samples, the model is allowed to adapt in an unsupervised manner. First, the efficacy of gradient-free adaptation of Batch Normalization statistics at test time is evaluated. Next, additional gains achieved through self-learning are investigated, where the model generates its own labels and uses them for further fine-tuning.

Taken together, this work provides tools to both estimate and improve model performance under distribution shift. Building models that generalize reliably is a critical milestone towards the safe, real-world deployment of AI systems.

Zusammenfassung

Maschinelle Lernsysteme sind heute allgegenwärtig: Künstliche Intelligenz (KI)–basierte Modelle empfehlen Musik, generieren Code, erstellen Gemälde und verfassen sogar Gedichte. Im Gegensatz zu klassischen Programmen, die festen Regeln folgen, lernen diese Systeme aus Daten. Die Verteilung der Trainingsdaten bestimmt, auf welche Merkmale das Modell achtet, und prägt somit maßgeblich seinen Entscheidungsprozess. So hätte beispielsweise ein autonomes Fahrzeug, das ausschließlich mit Autobahnszenen trainiert wurde, Schwierigkeiten, sich im Stadtverkehr zurechtzufinden – und umgekehrt. Zwar ist es möglich, einen Trainingsdatensatz zu sammeln, das beide Szenarien abdeckt, doch lassen sich nicht alle möglichen Verteilungsänderungen berücksichtigen. Kann ein autonomes Fahrzeug beispielsweise bei ungewöhnlichen Wetterbedingungen navigieren, die nicht Teil der Trainingsdaten waren? Und allgemeiner: Wie lässt sich die *Generalisierungsfähigkeit* eines Modells über die gelernten Daten hinaus messen und verbessern?

Die Beantwortung dieser Frage ist entscheidend für die Entwicklung von KI-Systemen, die sowohl sicher als auch zuverlässig für den Einsatz in der Praxis sind. Aus diesem Grund wird in dieser Doktorarbeit untersucht, wie Tiefe Neuronale Netze sich bei Verteilungsänderungen verhalten und wie ihre Leistung verbessert werden kann. Je nach Anwendung erwartet man entweder, dass das System von vornherein gegenüber verschiedenen Arten von Verteilungsänderungen robust ist oder dass es sich anpassen kann, falls die Änderung über einen längeren Zeitraum anhält. In dieser Dissertation werden beide Szenarien behandelt.

Im ersten Szenario besteht das Ziel darin, möglichst allgemeine Repräsentationen zu erlernen, die in unterschiedlichen Situationen hilfreich sind. Ein Ansatz hierzu ist die Erweiterung der Trainingsdaten durch adversarial generierte Rauschmuster, die so gestaltet sind, dass sie einen Klassifikator maximal verwirren. Durch das gemeinsame Training des Rauschgenerators und des Klassifikators wird Letzterer robuster gegenüber verschiedenen Bildkorruptionen. Über die Bildklassifikation hinaus wird die Robustheit verschiedener Objekterkennungsmodelle unter Bildkorruptionen untersucht und verbessert. Anschließend wird der Fokus auf ein anderes Lernparadigma verschoben: das Kontrastive Lernen (KL), bei dem Modelle ohne Klasseninformation trainiert werden. Eine vorherige Arbeit entwickelte eine theoretische Verbindung zwischen Identifizierbarkeit und KL, und zeigte, dass KL unter bestimmten Annahmen den datengenerierenden Prozess umkehrt. In dieser Arbeit werden diese Annahmen gelockert, die theoretischen Garantien erweitert und die Ergebnisse empirisch validiert.

Wenn die Verteilungsänderung über mehrere Datenpunkte hinweg anhält, kann dem Modell erlaubt werden, sich in einer unüberwachten Weise daran anzupassen. Zunächst wird die Wirksamkeit einer gradientenfreien Anpassung der Batch-Normalisierungsstatistiken zur Testzeit untersucht, um Verteilungsänderungen entgegenzuwirken. Danach werden zusätzliche Verbesserungen durch Selbstlernen betrachtet, bei dem das Modell eigene Labels generiert und diese für weiteres Finetuning verwendet.

Insgesamt bietet diese Arbeit ein Instrumentarium, um zunächst die Leistung eines Modells unter Verteilungsänderungen abzuschätzen und sie dann zu verbessern. Das Erlernen zuverlässig generalisiender Modelle ist ein entscheidender Meilenstein auf dem Weg zu einer Welt, in der KI-Systeme sicher in realen Anwendungen eingesetzt werden können.

List of publications

All publications I contributed to during my PhD are listed below. A star indicates joint first authorship. Detailed contribution statements for the papers in this dissertation can be found in the corresponding parts of Chapter 3.

Peer-reviewed conference and journal publications

[Oral] Evgenia Rusak*, Lukas Schott*, Roland S. Zimmermann*, Julian Bitterwolf, Oliver Bringmann, Matthias Bethge, Wieland Brendel. A simple way to make neural networks robust against diverse image corruptions. *ECCV 2020.*

[Poster] Steffen Schneider*, **Evgenia Rusak***, Luisa Eck, Oliver Bringmann, Matthias Bethge, Wieland Brendel. Improving robustness against common corruptions by covariate shift adaptation. *NeurIPS 2020.*

[Journal] Evgenia Rusak*, Steffen Schneider*, George Pachitariu, Peter Gehler, Oliver Bringmann, Matthias Bethge, Wieland Brendel. If your data distribution shifts, use self-learning. *TMLR 2022.*

[Poster] Evgenia Rusak*, Patrik Reizinger*, Attila Juhos*, Oliver Bringmann, Roland S. Zimmermann, Wieland Brendel. InfoNCE: Identifying the Gap Between Theory and Practice. *AISTATS 2025*

Peer-reviewed workshop publications

[Poster] Claudio Michaelis*, Benjamin Mitzkus*, Robert Geirhos*, **Evgenia Rusak***, Oliver Bringmann, Alexander S. Ecker, Matthias Bethge, Wieland Brendel. Benchmarking Robustness in Object Detection: Autonomous Driving when Winter is Coming. *Machine Learning for Autonomous Driving Workshop, NeurIPS 2019.*

Publications not part of this thesis

During working on my PhD, I have co-authored the following publications which are not a part of this thesis:

[Journal] Zhe Li, Josue Ortega Caro, **Evgenia Rusak**, Wieland Brendel, Matthias Bethge, Fabio Anselmi, Ankit B. Patel, Andreas S. Tolias, Xaq Pitkow. Robust deep learning object recognition models rely on low frequency information in natural images. *PLOS Computational Biology 2023, vol. 19, num. 3.*

[Poster] Prasanna Mayilvahanan, Thaddäus Wiedemer, **Evgenia Rusak**, Matthias Bethge, Wieland Brendel. Does CLIP's generalization performance mainly stem from high train-test similarity? *ICLR 2024.*

[Poster] Amro Abbas*, **Evgenia Rusak***, Kushal Tirumala, Wieland Brendel, Kamalika Chaudhuri, Ari S. Morcos. Effective pruning of web-scale datasets based on complexity of concept clusters. *ICLR 2024.*

[Poster] Max Wolff, **Evgenia Rusak**, Wieland Brendel. Low-Pass Filtering Improves Behavioral Alignment of Vision Models. *ICLR 2024 Re-Align Workshop, submitted to ICLR 2026*

[Poster] Prasanna Mayilvahanan, Roland S. Zimmermann, Thaddäus Wiedemer, **Evgenia Rusak**, Attila Juhos, Matthias Bethge, Wieland Brendel. In Search of Forgotten Domain Generalization. *ICLR 2025*

Contents

1 Introduction

For someone who travels abroad for the first time, navigating their way through a foreign city can be a daunting task. Even though many familiar aspects of everyday life are still encountered, such as bakeries, supermarkets or train stops which are prevalent in all cities of the globe, navigation can still be hard, especially if there is a language barrier. Navigation is the more challenging, the larger the difference between the home town and the visited foreign city: For someone from the US, navigating through a Canadian city will be easier compared to finding one's way through a European or Asian city. The described phenomenon is an example of the so-called *distribution shift* between the known and the unknown. Anytime a novel situation or challenge occurs, humans need relatively more time to solve it compared to a mundane task they solved hundreds of times. Even operating a new washing machine takes longer the first time compared to the hundredth time. However, over time, humans adapt to new tasks and become more and more proficient with them. Navigating through a foreign city becomes easier with each day we spend exploring it.

The challenges associated with distribution shifts do not only concern humans but also machines. While modern machine learning techniques have showcased exceptional competence in diverse fields like natural language processing [1], computer vision [2, 3], recommendation systems [4], etc, many studies emphasize the vulnerability of machine learning models to changes in data distribution. A change in the data distribution for a machine is akin to a novel and unexpected situation for a human, and the cost of error made by the machine can vary depending on the application. While a sub-optimal product recommendation or a spam email in the inbox which circumvented the spam filter are tolerable, other errors can potentially cost human lives. For example, a tragic and fatal car accident happened while Joshua Brown's Model S was heading east on US-27A, a divided highway in northern Florida. A tractor-trailer coming from the opposite direction turned left across the Tesla's path. At the time, the Tesla was operating in Autopilot mode. According to the Tesla team, "Neither Autopilot nor the driver noticed the white side of the tractor trailer against a brightly lit sky, so the brake was not applied" [5]. They state that several rare coincidences contributed to the severity of the accident, thereby emphasizing that the event was unexpected and "out-of-distribution" for the Autopilot. As a different example, a vision model trained to detect pneumonia from X-ray scans performed well when tested on data from the same distribution, i.e. from the same set of hospitals which were used to generate the training data. However, the trained model failed to generalize to scans from novel hospitals [6], show-casing another instance of failure to deal with a distribution shift. To avoid fatal accidents and to ensure safe model behavior, the question of *out-of-distribution generalization* is a central problem when trained models are to be deployed in the real world. Exploring ways how to achieve this goal is the central focus of this work.

2 Background

2.1 What does *Out-of-Distribution Generalization* mean?

In order to understand how machines can be taught to better deal with test-time distribution shifts, we need to define what distribution shift means. Suppose we are given a training set comprising n observations X, together with the corresponding labels Y. We wish to train a parametric model to serve as a mapping function from the data to the labels $f_\theta : X \to Y$, with a learnable parameter θ. In order to learn this mapping, we need to minimize a loss function ℓ which measures the distance between the predictions and the labels. A supervised learning problem can be defined as the following (adapted from Liu et al. [7]):

Definition 1 *Given a training set of n observations of the form $\{(x_1, y_1), \ldots, (x_n, y_n)\}$ which are drawn from the training distribution $P_{tr}(X, Y)$, a supervised learning problem is to find an optimal model f_θ^* which can generalize best on data drawn from the test distribution $P_{te}(X, Y)$:*

$$f_\theta^* := \arg\min_{f_\theta} \mathbb{E}_{X,Y \sim P_{te}}[\ell(f_\theta(X), Y)]. \tag{2.1}$$

A commonly made assumption about the training and test data is that they share the same probability distribution and each training or test sample is mutually independent from the others. More formally, this means that $P_{tr}(X, Y) = P_{te}(X, Y)$. This property is denoted as *independent and identically distributed* or *iid.* Under this assumption, to obtain the model f_θ^*, the optimal solution is to minimize the average loss on the training set since it will also minimize the loss on the test set. This principle is referred to as Empirical Risk Minimization (ERM, [8]).

However, the test distribution can differ from the training distribution in real world settings, i.e. $P_{tr}(X, Y) \neq P_{te}(X, Y)$. One then speaks of a *test-time distribution shift* which can arise due to various factors. This work focuses on *covariate shift* [9, 10] which assumes that the distribution shift occurs in the marginal distribution $P(X)$ and not in the conditional distribution $P(Y|X)$ since the same object has the same label in the training and test distributions:

$$P_{tr}(X) \neq P_{te}(X), \;\; P_{tr}(Y|X) = P_{te}(Y|X). \tag{2.2}$$

Why do we expect a model to generalize to a test distribution if the data has not been sampled in an *iid* fashion? As soon as we remove the *iid* assumption between training and testing, we cannot expect ERM to arrive at the optimal solution any longer since ERM will retrieve the optimal model based on the training data alone. If

we have no knowledge about the expected distribution shift at test time, we cannot make statements about expected model performance. In general, models generalize better to those test distributions which are closer to the training data. This is intuitive and works similarly for humans: The more similar a novel task is to tasks one does every day, the simpler it will be to complete. We here focus on the task of image classification which also makes the task of out-of-distribution (OOD) generalization easier: The general structure of natural images imposes certain biases on the model which aid generalization. On an abstract level, natural images show objects against distinct backgrounds, and features learned by the model to represent certain objects are helpful for other objects against other backgrounds.

As a next step, a few simple examples for distribution shifts will be shown to provide the reader with an intuition for what constitutes such a shift, before building a more abstract categorization of different distribution shifts.

2.2 Distribution Shifts: A Practical Example

To provide a simple example for different kinds of distribution shift, I took photos of three objects from ImageNet [11] with my smartphone camera against different backgrounds in my apartment, and used a ResNet50 [3] model trained on ImageNet to classify those images. The ImageNet dataset contains 1.3M images organized into 1000 classes.

The goal of this section is to visualize challenging distribution shifts under which a model misclassifies the input images while showing good *iid* performance. In order to avoid cherry-picking ImageNet test images which the model classifies correctly, I picked from the most easy classes where the ResNet50 model has the highest test accuracy. Intuitively, not all ImageNet classes are equally hard for the model; for example, the model has a test accuracy of 100% on the classes "Pomeranian", "echidna" or "killer whale". However, using either of those classes for this experiment is difficult, as I do not have convenient access to those animals to photograph them in unusual environments. Among the more feasible classes, the pretrained ResNet50 model achieves good test accuracies on the more boring classes "fig" (90% test accuracy), "banana" (94% test accuracy) and "Granny Smith" (92% test accuracy), all of which I could purchase in the local supermarket. I show visualizations of the three classes in Fig 2.1. In the top row, images from the ImageNet test set are shown; in the bottom row, I show images of these fruits taken in my apartment. As we can see, the ResNet50 correctly classifies all images, both from the ImageNet test set as well as the images taken with my smartphone camera. Despite the ImageNet test set being over ten years old, we can see that my smartphone camera (which is newer than that) does not constitute a challenging distribution shift.

Changing the background As a first step, I placed the fruits on backgrounds I thought should be challenging and unusual for the model: I put them on a chess board, on top of a painting of the Golden Gate bridge and in a wine glass. Since the painting of the bridge has been drawn by me, it is definitely out-of-distribution for the ResNet50. Still, the distribution shifts are not challenging enough for the most part: I took many images and almost none of them have been misclassified. I show non-cherry-picked examples in Fig. 2.2 where we can see that only the figs in the wine glass have been misclassified by the model. Arguably, this has been the most

Figure 2.1: ImageNet vs my own images: An ImageNet-trained ResNet50 classifies *iid* images correctly. Both the images taken from the ImageNet test set (top row) or images taken with my smartphone camera in my apartment (bottom row) are not challenging to the model.

Figure 2.2: Varying the background is not challenging (enough): Placing the fruits on a chess board, a painting of the Golden Gate bridge or in a wineglass does not constitute a challenging distribution shift in most cases. Among all tested combinations, the model only misclassifies the figs if they are put in the wine glass.

challenging distribution shift of all tested in this experiment, both for the model and the figs.

Stronger background and environment variations Next, I tried to change more than only the background of the images and placed the fruits in more challenging environments: In the arms of a (toy) octopus, in various door gaps and below a bed frame. As shown in Fig. 2.3, these distribution shifts prove to be sufficiently difficult for the model and all images are misclassified. One might argue that the distribution shifts are artificial and unrealistic and thus, it does not matter that the model fails. However, as a human, we are still able to classify the images correctly, despite the artificiality of the distribution shift. And in fact, many of the distribution shifts studied by the robustness community *are* artificial in the sense that they are simulated, and we know that robustness to artificial distribution shifts is a good proxy for performance in the real world.

A caveat in this argument chain is the limited label space of ImageNet models: Only a single label can be picked for each image and there are only 1000 classes an ImageNet-trained model has seen. Therefore, while a human might pick the octopus in the first, fourth and seventh images in Fig. 2.3 because this is the most prominent object, the ResNet50 does not know what an octopus is. Thus, it should ignore it and focus on the object it does know, similar to how the watermelon has been ignored in the right-most image in Fig. 2.1 in the top row when the ResNet50 chose to predict "banana".

Since the model does not know what an octopus is, we also cannot explain the misclassification of the images containing an octopus in Fig. 2.3 with its confusion seeing an octopus holding fruits. As octopuses are carnivores and do not eat fruit [12], this might have explained the misclassification if the label space contained the class

Figure 2.3: Stronger background variation is challenging: Placing the fruits in the arms of a (toy) octopus, in various door gaps or under a bed frame confuses the model and leads to a misclassification.

Figure 2.4: Smaller object sizes partially explain the previous challenging distribution shift: Placing the camera closer to the objects for the same distribution shifts studied in the previous experiment, we find that the model is again able to classify most of the fruits.

"octopus" but cannot explain the failure of the ImageNet-trained ResNet50.

Controlling for object size An attentive reader may have noticed a possible confounding factor for reduced performance in the varying object sizes between Figures 2.1, 2.2 and Fig. 2.3: In the last Figure, the objects are smaller compared to the previous images which imposes an additional challenge. To control for this confounder, I took additional photos with the same distribution shifts from the previous paragraph but put the camera closer to the objects. In Fig. 2.4, we see that the model performance has been restored for most objects, e.g. the "Granny Smith" apples are classified correctly in all three distribution shifts while the figs seem to be the most challenging class. This additional analysis highlights the importance of (literally) taking a closer look at different confounding factors which might offer an easier explanation for the observed effects.

In this section, I tried to show how a popular computer vision model handles practical and less practical distribution shifts for three easy ImageNet classes. In the next section, I will take a step back and provide a more general overview over possible distribution shifts.

2.3 Different Kinds of Distribution Shift

In this work, it is argued that the *iid* assumption is often violated in practice, making progress in out-of-distribution generalization essential for ensuring the safe deployment of machine learning models in real-world settings. As shown in the previous section, the *iid* scenario is comparatively straightforward. Although many of the distribution shifts evaluated in controlled environments have not posed significant challenges, the unfortunate reality is that once ML systems are deployed in real-world contexts, the nature of potential distribution shifts cannot be anticipated in advance. Schematically, this

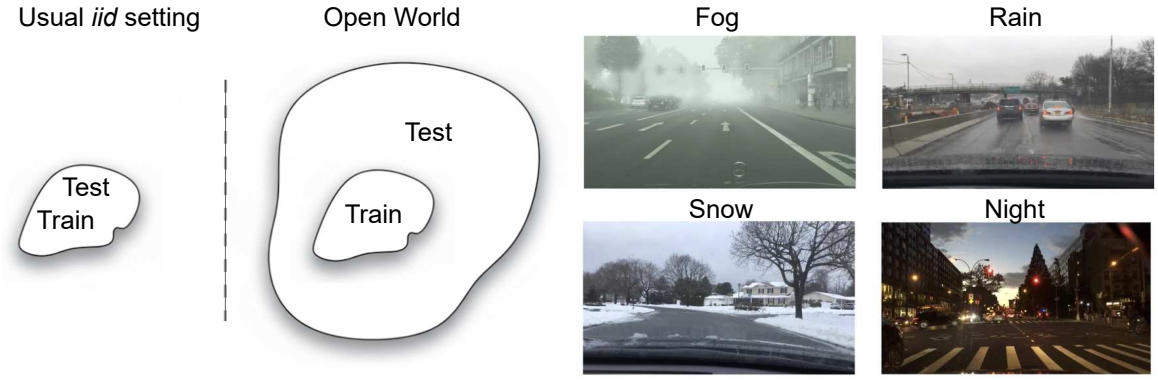


Figure 2.5: IID vs OOD: While Deep Neural Networks (DNNs) excel in settings when the training and test data are sampled from the same distribution, this assumption is rarely fulfilled in the real world where the model may enounter different kinds of distribution shifts. For example, an autonomous car needs to be able to deal with different variations of fog, rain, snow or dim lighting conditions.

dichotomy is shown in Fig. 2.5 (left): While in the usual *iid* setting, the train and test data points are drawn from the same distribution, the test data is drawn from a much wider distribution in the real world compared to the training distribution. We can refer to the setting encountered in the real world as the "Open World" scenario which reflects the uncertainty with respect to the expected distribution shifts.

What are some of the possible ways how the *iid* assumption can be violated? Usually, the expected or unexpected distribution shifts depend on the application. In certain medical applications, different conditions are diagnosed based on biomarkers which are extracted from biomedical measurements. A reliable biomarker is one that accurately detects a specific medical condition. However, biomarkers are often derived from a cohort that differs from the target population. This can compromise the biomarker's applicability to new individuals [13]. Dataset shifts like these occur frequently in biomedical research due to factors like recruitment biases.

A prominent example for recruitment bias in medicine occurred when studying the efficacy of ivermectin for the treatment of Covid-19. While some studies, e.g. [14], show a faster recovery rate with ivermectin treatment compared to a placebo (at $p < 0.001$), other studies, e.g. [15], show that administering ivermectin does not result in any noticeable reduction in either the intensity of COVID-19 symptoms or the duration for which these symptoms are experienced by the patients. Scott Alexander analyzes a multitude of studies on ivermectin efficacy for the treatment of Covid-19 on his blog [16] and relates the success rates to geography: Being a dewormer medicine, ivermectin is most successful in countries where the prevalence of worm infections is high, such as in Bangladesh, eastern India or Colombia. He reasons that "being full of worms may impact your ability to fight coronavirus", citing [17]. Ivermectin successfully treats a worm infection, thereby allowing the patient's organism to focus on the Covid-19 infection. If no worm infection is present, ivermectin treatment is either neutral or harmful due to its side effects. Given how prominently ivermectin has been discussed as a possible treatment option on various social media platforms, it is interesting to see that the geography related distribution shift in worm infection prevalence may explain the discrepancy in its efficacy across different studies.

In autonomous driving, different weather conditions or an unusual combination thereof

can constitute a distribution shift, see Fig. 2.5 (right). From my own experience with a Tesla Model 3, the Autopilot worked well during moderate rain, but refused to operate as the rain became strong. While the list of possible distribution shifts is extensive, the following provides an overview of the most well-studied cases.

Domain Shift [18] Domain shifts occur when a model trained on one domain (e.g. natural images) is tested on data from a different domain (e.g. sketches). For example, X-ray scans from different hospitals may exhibit different characteristics due to differences in scanner systems [6, 19]. A "trick" to produce more training data can be the generation of synthetic data; however, deploying a model trained on synthetic data in the real-world will lead to a sim-to-real domain shift [20, 21]. An autonomous car that has learned to navigate in a certain city may experience a domain shift when it is deployed in a different city [22]. Images captured during the day might have different visual characteristics compared to images captured at night due to changes in lighting conditions [23]. Another example is shown in Fig. 2.6 where a DNN trained on clean data misclassifies an image of a snow leopard when artificial snow is added to the image.

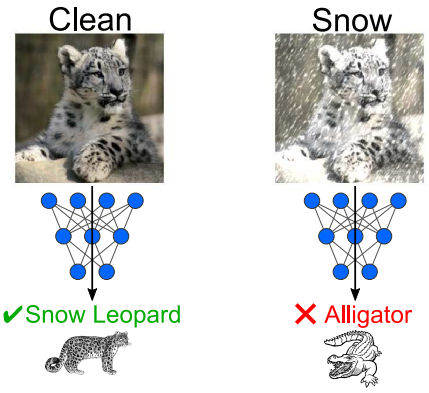


Figure 2.6: Example of a test-time distribution shift: A model which has only seen clean images during training misclassifies an image of a snow leopard when snow is added to the image.

Data evolution over time [24] In many applications, the data distribution is not static but evolves over time. The training data is then collected over a certain period of time, and new data is generated during deployment. In this scenario, the expected test-time distribution shift is often unknown because the test data is being generated continuously. For example, an autonomous car may encounter an unexpected weather condition it has not been trained for which leads to poor performance [25–27]. Even worse, a combination between multiple known weather conditions may result in a new unknown condition. Models deployed for image-based quality control may perform worse over time due to accumulation of dust and debris on the camera lens [28]. The drifting prevalence of certain diseases in the population needs to be taken into account during diagnosis: This has been especially notable with Covid-19 and its strongly varying seasonal prevalence in the population [29]. Other examples include the increased prevalence of diabetes [30] or melanoma [31].

Imbalanced Data [32] One speaks of an imbalanced dataset if certain classes (minority groups) are underrepresented compared to other classes (majority groups). Such skewed data distributions are known to result in classifiers which are biased against underrepresented minority groups. At test-time, the class-balance may shift, incurring a distribution shift. Larrazabal et al. [33] investigated thoracic diseases under different gender imbalance conditions and found a consistent decrease in performance for underrepresented genders when a minimum balance has not been fulfilled. Fraud detection [34, 35] is another example where class-imbalance naturally occurs since fraud events are anomalies. Another curious example is bird sound classification [36]

where class imbalance causes poor performance on underrepresented classes.

Adversarial Attacks [37, 38] Adversarial Attacks constitute a hand-crafted and malicious distribution shift where a test image is changed in an imperceptible way to be misclassified by a trained model. While adversarial attacks target a specific model, they have been shown to generalize across architectures and training sets [39], and can also be successfully applied in the physical world [40]. Further, access to the weights of the target model is not necessary and the attack can be successful solely based on the target model's decisions [41]. It has been demonstrated that adversarial attacks can also be applied to other tasks, such as semantic segmentation or object detection [42, 43].

There are many other kinds of distribution shift. For example, in this work, it is assumed that the objects the model needs to classify belong to the set of classes the model has been trained for. The problem of teaching a model to deal with unseen classes is referred to as *Open Set Recognition* [44] and is beyond the scope of this work.

2.4 How do we measure OOD generalization?

In order to quantify progress in improving OOD generalization, we need to be able to measure the current state of affairs. For this, researchers use so-called *benchmarks*. The Merriam-Webster Dictionary [45] defines a benchmark as "a standardized problem or test that serves as a basis for evaluation or comparison (as of computer system performance)". In Computer Vision, there exists a range of such benchmarks which aim to test the models' performance in a standardized way. I will briefly introduce the most popular benchmarks used in image classification; other vision tasks such as object detection and semantic segmentation have their own set of benchmarks. Benchmarks can be grouped based on the task the model needs to solve.

The simplest task which requires no model training is measuring the *accuracy on an OOD test set*. For example, one may evaluate the stability or *robustness* of model predictions when fed images corrupted with Gaussian or Salt-and-Pepper noise. The most prominent benchmark measuring model robustness under input corruptions is the dataset ImageNet-C [46] which is comprised of corrupted versions of ImageNet validation images, where each corruption has been applied with five severity levels. The resulting dataset therefore has 75 different domains, given by the different corruptions. These corruptions include different noises, blurs and image artifacts. Due to the popularity of ImageNet-C in the research community, the used image corruptions have also been applied to other common datasets resulting in Pascal-C, Coco-C and Cityscapes-C [47] and MNIST-C [48].

Other notable benchmarks which use OOD accuracy as a measure for OOD generalization are ImageNet-R [49], ImageNet-A [50], ImageNet-Sketch [51] and ObjectNet [52]. ImageNet-R contains images with various artistic renditions of 200 classes of the original ImageNet dataset, ImageNet-A contains natural images which have been selected to yield chance level classification performance of ImageNet-trained ResNet-50 models, ImageNet-Sketch contains sketches of ImageNet classes, and ObjectNet contains photos of ImageNet classes shot from unusual angles.

Figure 2.7 shows exemplary visualizations of the distribution shifts present in the benchmark datasets described above. Although the images are not particularly hard

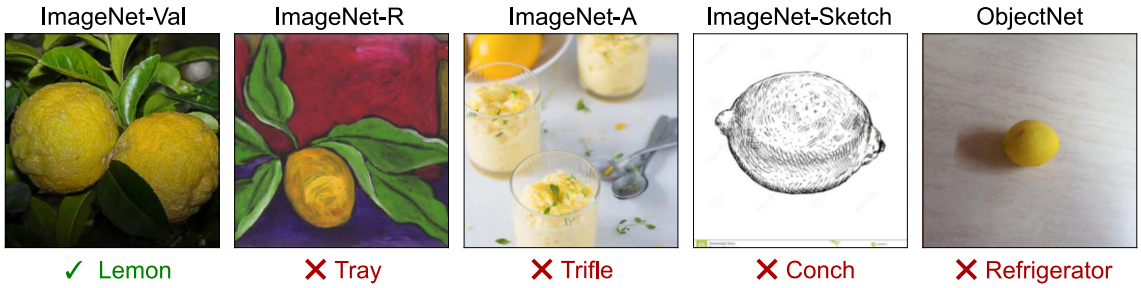


Figure 2.7: Examples of OOD benchmarks: The distribution shifts present in the benchmark datasets ImageNet-R, ImageNet-A, ImageNet-Sketch and ObjectNet are exemplified with an image from the class "lemon". An ImageNet trained ResNet50 only classifies the test image from the ImageNet validation set correctly and misclassifies the images from the shifted datasets.

for a human, a ResNet50 trained on ImageNet misclassifies all images from the shifted datasets, and only the image taken from the *iid* ImageNet validation set is classified correctly. I would argue that the ImageNet-A image actually does slightly look like a trifle which is a "layered dessert of English origin", see e.g. Amanda's Cookin' [53] for a traditional recipe and example images. On the other hand, the example image from ImageNet-Sketch definitely does not look like a conch which is a marine snail of the subclass Prosobranchia [54]. We can conclude that some errors made by the model are more interpretable than others.

It is noteworthy that many OOD generalization benchmarks listed here are ImageNet-based. This is because the introduction of the ImageNet dataset has revolutionized Computer Vision as it was the first large-scale dataset with high-quality labels. For a long time, progress in Computer Vision has been achieved and measured by training models on ImageNet and testing them in ways described above. Recently, the more wide-spread use of large-scale models trained on web-scale datasets in a self-supervised way, for example CLIP [55], ALIGN [56] or ChatGPT [1] has shifted our understanding of how OOD generalization should be measured. For example, CLIP has popularized the metric of "zero-shot" accuracy allowing to evaluate a model trained in a self-supervised way on a classification task. A more broad discussion what the advent of web-scale datasets means for how OOD generalization should be evaluated can be found in the Discussion chapter of this work.

Another popular way of measuring OOD generalization is the area of *transfer learning*, where the representations learned by a model are evaluated on some downstream task. Transfer learning in Deep Learning began with Donahue et al. [57] who showed in their seminal paper that ImageNet-trained models transfer well to a diverse set of downstream tasks, and established the practice for machine learning practitioners to finetune a model pretrained on ImageNet on the target task instead of training the model from scratch. Thus, in contrast to simply evaluating accuracy, in transfer learning, one either trains a linear classification module on top of the frozen model weights, or allows the model to be finetuned using a small learning rate. Examples of transfer learning tasks include fine-grained flowers [58] or bird species [59] classification, land use and land cover classification [60] or texture classification [61]. Popular benchmarks to evaluate transfer learning in a standardized way are VTAB [62] and WILDS [63].

Related to transfer learning is the area of *domain generalization* where a model is

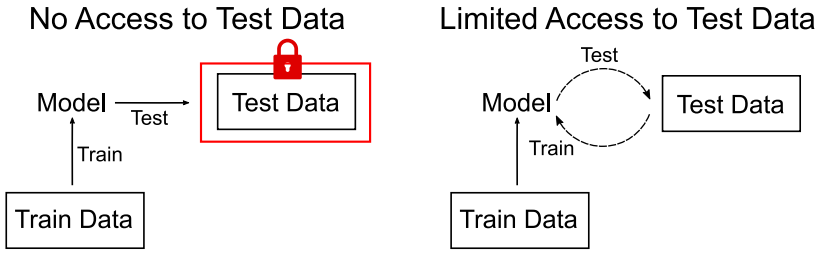


Figure 2.8: Access to test data determines the robustification strategy: (Left) If there is no access to the test data at training data, our best strategy will be to train the most general model; (right) if there is limited access to unlabeled test data during training, we can adapt the model to the distribution shift.

trained on $N-1$ domains and needs to generalize to the held-out domain. Gulrajani and Lopez-Paz [64] have shown that with proper model selection, training the model on the concatenation of the $N-1$ domains available during training with standard ERM actually leads to state-of-the-art results. They released the benchmark DomainBed [65] which standardizes the training and evaluation of models on the domain generalization task, including fair model and hyperparameter selection.

In the area of *adversarial robustness*, the goal is usually to find the smallest perturbation in the image space which will shift the prediction of the target model from the correct label to an incorrect one. There exist a plethora of different adversarial attacks and defenses, with the caveat that many published defenses tend to be broken later with a more advanced adversarial attack. Foolbox [66, 67] and Cleverhans [68] have been introduced as frameworks to adversarially attack models. Later, AutoAttack [69, 70] has been proposed to standardize the attacks, and to streamline the attack procedure by always using a diverse set of attacks. The benchmark RobustBench [71] has been established to track progress in adversarial robustness.

2.5 Strategies to improve OOD generalization

Having established that test-time distribution shifts can be an issue, I have introduced the notion of standardized benchmarks to measure robustness to such shifts. Can we now arm our models to better withstand the distribution shifts? There is a large number of different methods, algorithmic and data-driven, how OOD generalization can be improved. A categorization of robustification methods can be attempted in multiple ways. Here, I choose a taxonomy based on the *knowledge* of the expected distribution shift. In some applications, such as in online streaming applications when different users upload their own photos of a certain object, each new incoming data point will have a unique distribution shift which is unknown beforehand. On the other hand, in autonomous driving, weather changes relatively slowly and many images with the same weather conditions will be taken as the car drives around. Due to the smoothness of weather changes, it is reasonable to expect that an image taken a second or even few minutes later than the previous one, will have the same distribution shift. In medical imaging, scanners can differ between hospitals. However, if I wish to classify X-ray scans from a certain hospital, I could ask a technician to provide me with some number of images to get an understanding of the expected distribution shift.

With no information of what the distribution shift might be, the best solution will be to train the most general model we can, such that the performance drop on any possible shift is minimal, see Fig. 2.8 (left). If some information about the expected distribution shift is available, such as for example an unlabeled set of images, the best course of action would be to use those images to somehow align the model representations to those images, as depicted in Fig. 2.8 (right). A strong argument to allow the model to adapt to the distribution shift at test-time is that humans adapt all the time without noticing. For example, when we enter a dark room, we see nothing in the first few seconds, until our pupils widen to allow for more light in and we start seeing objects. To better understand a quiet speaker, we can 'strain our ears' and attempt to drown out other noises. Finally, Alexey Efros, a famous researcher working in test-time adaptation likes to use the analogy of a human-tiger encounter: The first time a human meets a tiger and needs to defend themselves, this is a classic test scenario with an unexpected test-time shift. If the human survives, the next tiger encounter will be smoother, until the human masters tiger handling and tiger encounters no longer pose an issue.

It is fitting to recall the "no-free lunch theorem" [72, 73] here which states that no single optimization algorithm is universally superior to all others over all possible problems. It has implications for understanding the limitations of optimization algorithms and the importance of selecting appropriate algorithms for specific problem domains. In the context of OOD generalization, the theorem implies that there does not exist a single solution which will be optimal for all distribution shifts, and the best solution will depend on the target task. That said and being mindful of the assumed knowledge discrepancy, I will now provide an overview of techniques used to improve OOD generalization.

2.5.1 Strategies if there is no access to test data during training

Not having access to the test data during training is a standard assumption in machine learning. When performing model selection, the standard technique is to mimic a test set with a held-out validation set which has been sampled from the training data. An example in the context of OOD generalization when this assumption would be reasonable could be a mobile app with the functionality to classify images of flowers the users take, possibly with the option to offer ordering the flowers for them. Each new image could be taken under new lighting conditions and depend on the specifics of the smartphone camera. The distribution shift of each new incoming image would be unknown. This setting is also reflected in Section 2.2 where I classified different fruits in unusual environments in my apartment. With no knowledge of the expected test-time distribution shift, the optimal solution would be to learn the most general model. How could this be accomplished?

More training data A general solution to all kinds of machine learning problems seems to simply be adding more training data. Defying the no-free-lunch theorem in some sense and being a highly unsatisfactory solution from an intellectual point of view, training the model on a larger and more diverse dataset seems to be the best, albeit often not feasible solution. Increasing the dataset size works best if also the model size is increased at the same time, thereby increasing the training cost.

Further, in supervised learning, increasing the dataset size involves long and arduous labeling efforts. Nevertheless, increasing the dataset and model size are known to do wonders to model performance. In the area of Deep Learning, scaling the training dataset and/ or model size has been studied across different disciplines, e.g. by Zhai et al. [74], Hestness et al. [75], Kaplan et al. [76], Henighan et al. [77], Rosenfeld et al. [78], Gordon et al. [79], Hernandez et al. [80]. In the context of OOD generalization, having more training data has been shown to increase performance across a wide range of OOD benchmarks [81, 82]. An avenue to reduce the model size after training to make it more efficient to deploy is knowledge distillation which involves transferring knowledge from a larger model to a smaller one, thereby demonstrating the potential benefits of scaling down models while retaining performance [83].

Data Augmentation Data augmentation is an artificial way to mimic a larger and more diverse training dataset, without actually having to collect and label it. Some preprocessing functions which diversify the training data, such as horizontal flips or random crops, are routinely included as part of the data preprocessing pipeline in all major Deep Learning frameworks. The data augmentation strategy can also be chosen to mimic distribution shifts one might expect at training time. For example, if we think that the lighting conditions might change for some test inputs, we could mimic this by choosing an appropriate preprocessing function. In the context of autonomous driving, training on images which were augmented with synthetically generated rain drops has been shown to increase detection performance on road scenes with real rain [26]. Using stylization as data augmentation has been shown to increase OOD performance [84, 85], likely due to the strong diversity of the training data it produces [86]. Hendrycks et al. [49] use an autoencoder trained to perfectly reconstruct the training data and turn it into a data augmentation strategy by adding random perturbations to the autoencoder's weights. Another popular data augmentation technique is AugMix [87] where augmented images are mixed with each other while enforcing consistency in the embeddings of the augmented images. This technique results in increased robustness and improved uncertainty calibration. Mintun et al. [88] study the effect of different data augmentation techniques on the model's performance to different image corruptions. They find a strong correlation between the perceptual similarity of the data augmentation to the input corruption, thereby demonstrating that using additional knowledge about the expected distribution shift can help us design suitable augmentation functions. In the adversarial robustness domain, adversarial training [89] has been established as the best performing defense against attacks, and is a form of data augmentation where the training dataset is augmented with adversarially modified images to stabilize the model to them. Several extensive survey articles on data augmentation for images in Deep Learning have been published [90–92].

Nonlinear Independent Component Analysis Nonlinear Independent Component Analysis (ICA) seeks to recover the underlying independent factors of variation from observations that have undergone complex, nonlinear transformations [93, 94]. In the nonlinear setting, the observed data is assumed to be generated by a well-defined, invertible function g, mapping the latent sources z to the observations $x = g(z)$ [95, 96]. Nonlinear ICA can thus be viewed as a *demixing* problem: the objective is to learn a function f that approximates the inverse of the generative process, $f \approx g^{-1}$, enabling the original latent sources to be recovered. This framework has important implications for OOD generalization. For instance, a model trained on cows mostly in

pastures may fail to recognize cows on a beach if its latent representation entangles object identity with background features—the classic "cow-on-the-beach" problem [97]. By explicitly recovering the independent generative factors, nonlinear ICA produces latent representations that are robust to irrelevant variations in the scene, such as background, lighting, or object pose [93, 94]. Recent approaches further enhance identifiability by leveraging temporal structure, inductive biases and supervision, or contrastive learning [93, 98, 99], grounding the learned representations in true causes of variation and improving generalization to novel scenarios.

2.5.2 Strategies if there is limited access to unlabeled test data at training time

In some situations, the test-time distribution shift is known and one has access to (some) unlabeled test data. For example, one could imagine a camera-based quality control system which is tasked with monitoring a production line in manufacturing. Even if the system has been trained on images from the very same production hall, there will be slight variations depending on the location, the camera position and the lighting conditions for every new camera. However, assuming that the position of the camera is fixed, the position-related distribution shift will not change over time. Thus, we could collect some amount of images from the new camera location and use them to adapt the model. The optimal solution would be to label a portion of the test data to allow for supervised adaptation, but even without labels, unsupervised adaptation can strongly boost performance.

Unsupervised Domain Adaptation The research area of unsupervised domain adaptation is the most relevant one to this dissertation when we have access to the test set during training. Domain adaptation aims to adapt a model trained on one domain (the source domain) to perform well on another domain (the target domain) without requiring labeled data from the target domain. "Unsupervised" refers precisely to this requirement of only using unlabeled data from the target domain. Important works in this realm include the publication by Ganin et al. [100] where the authors introduce domain-adversarial neural networks, a method for unsupervised domain adaptation that leverages adversarial training to align feature distributions across domains. Another important paper is Ganin and Lempitsky [101] which proposes an approach for unsupervised domain adaptation using a gradient reversal layer to align feature distributions between domains during backpropagation. Rebuffi et al. [102] introduce a configurable architecture capable of dynamically adapting to various visual domains using adapter residual modules. Tzeng et al. [103] propose Adversarial Discriminative Domain Adaptation (ADDA), a method that aligns feature distributions between domains while simultaneously training a domain discriminator to distinguish between source and target domain features. Unsupervised domain adaptation is a huge research area which has been covered in extensive survey articles, e.g. [104]. In particular, domain adaptation is especially relevant in medical imaging [105].

2.5.3 Other approaches

Important approaches which are less relevant for the dissertation, but nonetheless should be mentioned here include Causal and Invariant Learning.

A fundamental challenge for deep neural networks (DNNs) in achieving out-of-distribution (OOD) generalization arises from their reliance on correlations rather than causal relationships. DNNs excel at identifying statistical regularities in training data, but these correlations often do not reflect the true underlying causal structure of the world. As a result, models can perform well on in-distribution data yet fail dramatically when confronted with small perturbations or shifts in the input distribution, such as noise or background changes. This limitation occurs because the network has learned to associate spurious features with the target label rather than understanding which features causally determine the outcome.

Causal learning explicitly aims to identify and model the underlying cause-and-effect relationships between variables. By uncovering the mechanisms that generate observed associations, causal methods are expected to maintain robust performance under distributional shifts, as the fundamental causal structure is presumed to remain stable across environments. Incorporating causal reasoning into machine learning thus provides a promising pathway for improving OOD generalization, allowing models to distinguish between features that are truly predictive and those that are merely correlated. Seminal works on causality include Schölkopf [106], Pearl [107], Spirtes et al. [108]. Invariant learning techniques further build on this idea by acquiring representations that remain stable across diverse environments. By focusing on elements of the data that exhibit consistency across settings, these methods enhance the model's ability to generalize to novel and unseen distributions. Invariant Risk Minimization (IRM) is the most widely studied approach [109], with numerous extensions and improvements proposed [110–114].

2.6 Research Questions

This dissertation aims to explore techniques for improving the OOD generalization capabilities of DNNs. The previous section introduced a categorization of methods based on test-data access (Fig. 2.8), which can also serve here to broadly formulate the research questions addressed in this work as:

High-level categorization of Research Questions

How can the performance on shifted data be improved if there is **no access** to the distribution shift during training?

How can the performance on shifted data be improved if there is **limited** access to the distribution shift during training?

The core studies comprising this dissertation can be grouped into two categories. A schematic overview of these studies is shown in Fig. 2.9. Briefly, the first two investigate two data augmentation strategies, followed by testing the trained model on various distribution shifts. The third study is situated in the area of nonlinear ICA, focusing on learning an identifiable representation of the different constituents that make up a scene. The final two studies examine how test-time adaptation can enhance model performance.

To provide additional context, Part One, where studies P1–P3 are situated, begins by examining how specific data augmentation techniques influence OOD generalization.

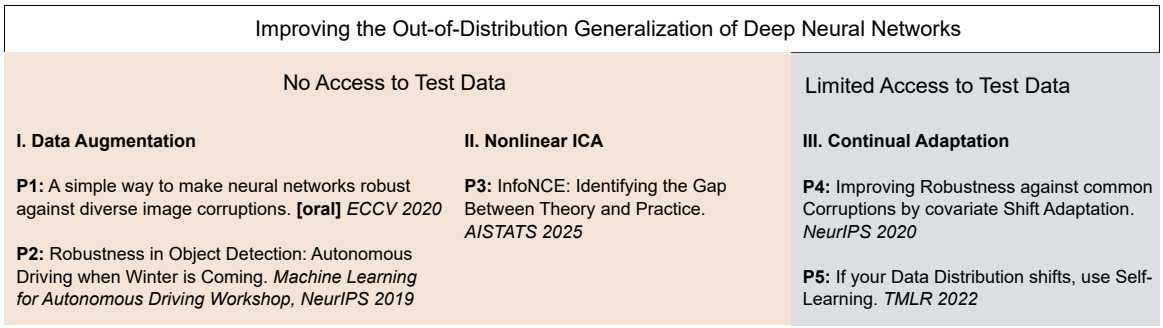


Figure 2.9: Schematic organization of the studies in the dissertation: The studies are organized into two main categories based on whether there is any access to the test data during training. The studies P1-P3 do not assume test data access, while studies P4-P5 assume limited access to unlabeled test data.

The studied setting has been introduced in Section 2.5.1, where no access to test data at training time is assumed. In P1 ("A Simple Way to Make Neural Networks Robust against Diverse Image Corruptions"), a simple generative model is trained in an adversarial manner to produce worst-case noise for an image classifier. The generative model and classifier are subsequently co-trained in a minimax framework, yielding improved OOD generalization results across multiple benchmarks. In P2 ("Benchmarking Robustness in Object Detection: Autonomous Driving when Winter is Coming"), robustness of object detectors to image corruptions is shown to be enhanced through the use of stylization. To quantify these improvements, the ImageNet-C benchmark is generalized to several object detection datasets.

Next, attention is directed toward a fundamentally different approach to representation learning compared to supervised methods, namely Nonlinear ICA. The assumptions and benefits of this paradigm have been discussed in Section 2.5.1. In P3 ("InfoNCE: Identifying the Gap Between Theory and Practice"), the gap between theory and practice in unsupervised representation learning is investigated by relaxing certain theoretical assumptions on identifiability guarantees, with observed benefits in controlled experimental settings. The remaining discrepancies between theoretical results and practical performance are also highlighted.

Part Two of the following chapter comprises the studies P4 and P5, which investigate how limited access to unlabeled test data can be leveraged to improve OOD generalization. This setting has been introduced in Section 2.5.2. In P4 ("Improving OOD Generalization with Limited Access to Test Data"), it is demonstrated that unsupervised, gradient-free adaptation of Batch Normalization layers on the test set can significantly improve performance. In P5 ("If Your Data Distribution Shifts, Use Self-learning"), the additive benefits of the widely used domain adaptation technique of pseudo-labeling over Batch Normalization adaptation are further explored.

2.6.1 Outline

While Deep Learning has made remarkable progress in recent years and increasingly finds application in various domains, the out-of-distribution (OOD) generalization capabilities of such models remain uncertain, potentially limiting their future applicability. This chapter categorizes different types of distribution shifts and provides examples of how they may occur. Additionally, the notion of benchmarking is in-

troduced, and methods for measuring OOD generalization are explained. Finally, strategies to improve OOD generalization are presented.

The following chapter summarizes the individual studies conducted during the PhD. For each study, the motivation and context are outlined, and the main results are briefly presented, followed by a discussion.

After presenting these studies in Chapter 3, the thesis concludes with a comprehensive discussion and an exploration of the key open questions emerging from the experimental findings in Chapter 4, ending with a final conclusion in Chapter 5.

3 Main Contributions

> Writing long books is a laborious and impoverishing act of foolishness: expanding in five hundred pages an idea that could be perfectly explained in a few minutes. A better procedure is to pretend that those books already exist and to offer a summary, a commentary.
>
> *Jorge Luis Borges*

3.1 P1: A Simple Way to Make Neural Networks Robust against Diverse Image Corruptions.

Evgenia Rusak*, Lukas Schott*, Roland S. Zimmermann*, Julian Bitterwolf, Oliver Bringmann, Matthias Bethge, Wieland Brendel. A simple way to make neural networks robust against diverse image corruptions. **[Oral]** *ECCV 2020*.

The full paper can be found in the Appendix A, starting from page 70.

Author contributions W.B. and M.B. developed the idea of the adversarial noise training; E.R. and W.B. conceived a realization with input from L.S. and R.S.Z.; E.R., L.S., R.S.Z. and W.B. designed and performed the experiments on ImageNet-C with input from W.B. and J.B.; L.S., E.R. R.S.Z. and W.B. designed and performed the experiments on MNIST-C; L.S., R.S.Z., E.R. and W.B. designed and performed adversarial robustness evaluation; E.R., L.S., R.S.Z. and W.B. wrote the manuscript with input from J.B., O.B. and M.B.; E.R. designed figures 1-3 with input from L.S., R.S.Z. and W.B.; R.S.Z. designed figure 4 with input from E.R., L.S., and W.B.

Motivation This research paper examined the critical issue of robustness in Deep Neural Networks (DNNs). As discussed in Chapter 2, DNNs frequently encounter difficulties when handling test-time distribution shifts. This lack of robustness presents a significant challenge for the reliable deployment of DNNs in real-world applications.

Among the various types of distribution shifts, this paper focused on model performance under common corruptions. These refer to naturally occurring perturbations, such as image noise, weather conditions (fog, rain), or compression artifacts, which do not alter the underlying semantic content of the input. Ideally, DNNs should maintain high accuracy in the presence of such corruptions.

At the time of this study, robustness research largely concentrated on adversarial attacks, as introduced in Chapter 2, while naturally occurring distribution shifts remained comparatively understudied. Investigating common corruptions offers a valuable pathway for achieving several objectives: enhancing the overall robustness of DNNs, narrowing the gap between human and machine resilience to input variations, and potentially advancing techniques in domain adaptation. Addressing common corruptions thus represents a fundamental step towards making DNNs reliable and trustworthy in complex and unpredictable real-world environments.

Exploring the Power of Gaussian Noise for Image Robustness As a first step, the effectiveness of a simple approach was evaluated: Adding Gaussian noise during training via data augmentation. The ImageNet-C benchmark, which simulates various image corruptions such as noise, blur, weather effects, and digital artifacts, was used for testing.

When carefully calibrated, Gaussian noise was found to significantly improve the robustness of a standard ResNet50 model across nearly all corruption categories, with the exception of the "Fog" corruption[1].

The key factor was the selection of an appropriate noise intensity (standard deviation). Previous attempts using noise-based techniques showed mixed results, often with minimal improvement, likely due to the use of either very low or very high noise levels. These findings indicated that fine-tuning a pre-trained model with a carefully chosen noise level was critical for success. This approach provided a surprisingly effective method for enhancing the resilience of image recognition models against distortions encountered in real-world scenarios.

Beyond Simple Noise: Introducing Adversarial Noise Inspired by the success of using Gaussian noise to make models more robust, a natural follow-up question was what the optimal noise distribution should be. This question could be turned into an optimization problem to learn the optimal noise distribution. Thus, instead of random Gaussian noise, a small neural network called the Noise Generator was introduced. This Noise Generator was designed to create a special type of noise that was highly effective at confusing a given image classification model. This custom-tailored "Adversarial Noise" offered a potentially more powerful way to improve model robustness during training. A schematic and an exemplary pattern of Adversarial Noise is shown in Fig. 3.1 A. The term "worst-case" has a very specific meaning: It means one needs less noise when sampled from this distribution compared to when sampling Gaussian, Uniform or Salt-and-Pepper noise to change the prediction of a classifier. The amount of additive noise necessary for a misclassification was measured by the ℓ_2-distance between the noisy and the clean image.

The Game of Noise: Adversarial Noise Training for universal Robustness Finally, the Noise Generator and the image classifier could be optimized jointly. Rather than introducing random perturbations, the Adversarial Noise Generator (ANG) systematically generated noise patterns that maximally challenged the classifier. This process can be conceptualized as a dynamic minimax game:

[1]This has later been explained by Yin et al. [115] with the low-frequency Fourier spectrum of the "Fog" corruption. Data augmentation with Gaussian noise increases invariance to high-frequency corruptions, making the model more vulnerable to low-frequency ones.

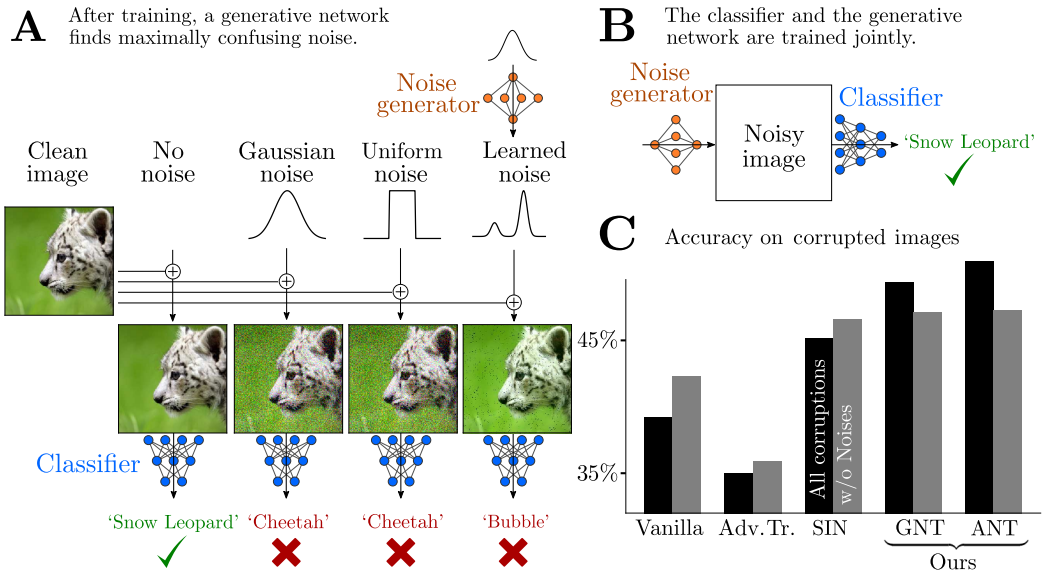


Figure 3.1: Schematic overview of the proposed approach. A: First, a small generative network was trained against a vanilla trained classifier to find the worst-case noise (adversarial noise). B: To achieve robustness against adversarial noise, the classifier and the noise generator were trained jointly. C: Robustness results against common corruptions for a vanilla, adversarially trained (Adv. Tr.), trained on Stylized ImageNet (SIN), trained via Gaussian data augmentation (GNT) and trained with the means of Adversarial Noise Training (ANT) were reported. With the proposed methods, the highest accuracy on common corruptions was achieved, both on all and non-Noise categories. Figure adopted from the paper [116].

- The Noise Generator sought to identify perturbations that most effectively disrupt the classifier's predictions.

- The classifier continuously adapted to these perturbations, learning to maintain robust performance in the presence of challenging noise.

A schematic representation of Adversarial Noise Training (ANT) is shown in Fig. 3.1 B.

Through this iterative, adversarial interaction, the classifier progressively acquired robustness. Importantly, this training procedure not only mitigated vulnerability to the generated adversarial noise but also substantially enhanced resilience to a wide range of naturally occurring image corruptions. At the time of publication, this approach achieved a new state-of-the-art performance on the ImageNet-C benchmark. Further improvements were obtained by integrating ANT with stylized image training [117]. Summarized results, alongside comparisons with baseline methods, are presented in Fig. 3.1 C.

The idea of ANT was inspired by regular adversarial training (AT) [89] which is the most successful adversarial defense to date. The difference between the ANT scheme and AT is that in AT, one optimizes the images directly whereas in ANT, one trains a noise generation network which produces the noise distribution. In AT, every image pixel is optimized separately whereas in this case, the noise distribution has to be optimal for all pixels on average.

Correlated Noise Patterns To enable the Noise Generator to model locally correlated noise patterns, such as snowflakes, experiments were conducted using a generator with a receptive field size of 3×3. Training with this more powerful Noise Generator provided additional benefits.

Discussion Previous attempts to use simple additive noise during image model training to improve robustness against common corruptions have produced mixed results. However, this work demonstrated that carefully selected noise types, combined with training on both clean and noisy images, could substantially enhance robustness against such corruptions. Inspired by adversarial training, further improvements were achieved by exposing the model to a "worst-case" noise generator. Although performance against certain corruptions could still be improved, promising directions included exploring more diverse noise generators, investigating complex interactions between noise and image content, and tailoring noise generation to specific image features.

Potential extensions of this work include experimenting with different architectures for the Noise Generator. In the current setup, the Noise Generator produced either completely uncorrelated or locally correlated noise patterns. Consequently, it could not model noise patterns spanning many pixels, such as smoke or clouds. Increasing the receptive field size could alleviate this limitation, but would require careful regularization: for instance, a full-image unconstrained Noise Generator could produce Universal Adversarial Perturbations (UAPs; [118]), which are unlikely to improve corruption robustness due to their generally small magnitude.

More powerful Noise Generators have been explored by Calian et al. [119], including architectures such as super-resolution models [120] and compressive autoencoders [121]. These approaches learn a perturbation vector δ to distort the Noise Generator's weights and train the classifier and generator jointly to enhance robustness. A direct comparison with the current approach is not feasible, as their evaluation used a downscaled version of ImageNet.

Another limitation of this work is that evaluation was conducted solely on ImageNet-C, which contains only synthetic corruptions. Evidence suggests that interventions improving performance on synthetic blurs in ImageNet-C also enhance robustness to real-world blurs [49]. Additionally, Kar et al. [122] extended ImageNet-C corruptions to 3D by estimating image depth maps, making the corruptions depth-dependent, and found that performance on 3D corruptions correlates with performance on the original 2D corruptions. Follow-up studies [47] also observed strong correlations between performance on synthetic and natural corruptions for object detection. Taken together, these findings suggest that evaluation on the original ImageNet-C corruptions provides a strong proxy for real-world robustness and can serve as a reliable indicator of expected performance in practical applications.

3.2 P2: Benchmarking Robustness in Object Detection: Autonomous Driving when Winter is Coming.

Claudio Michaelis*, Benjamin Mitzkus*, Robert Geirhos*, **Evgenia Rusak***, Oliver Bringmann, Alexander S. Ecker, Matthias Bethge, Wieland Brendel. Benchmarking Robustness in Object Detection: Autonomous Driving when Winter is Coming. *Machine Learning for Autonomous Driving Workshop, NeurIPS 2019.*

The full paper can be found in the Appendix A, starting from page 103.

Author contributions The initial project idea for improving detection robustness was developed by E.R., R.G. and C.M. The initial idea of benchmarking detection robustness was developed by C.M., B.M., R.G., E.R. & W.B. The overall research focus on robustness was collaboratively developed in the Bethge, Bringmann and Wichmann labs. The `Robust Detection Benchmark` was jointly designed by C.M., B.M., R.G. & E.R.; including selecting datasets, corruptions, metrics and models. B.M. and E.R. jointly developed the pip-installable package to corrupt arbitrary images. B.M. developed code to stylize arbitrary datasets with input from R.G. and C.M.; C.M. and B.M. developed code to evaluate the robustness of arbitrary object detection models. B.M. prototyped the core experiments; C.M. ran the reported experiments. The results were jointly analysed and visualized by C.M., R.G. and B.M. with input from E.R., M.B. and W.B.; C.M., B.M., R.G. & E.R. worked towards making our work reproducible, i.e. making data, code and benchmark openly accessible and (hopefully) user-friendly. Senior support, funding acquisition and infrastructure were provided by O.B., A.S.E., M.B. and W.B. The illustratory figures were designed by E.R., C.M. and R.G. with input from B.M. and W.B. The paper was jointly written by R.G., C.M., E.R. and B.M. with input from all other authors.

Motivation The motivation for this research paper is rooted in the urgent need to enhance the reliability and safety of autonomous vehicles (AVs) in diverse weather conditions. Imagine a future where AVs are commonplace, seamlessly integrated into our daily lives. However, this vision is compromised when AVs fail to navigate through unexpected weather events, such as a heavy snowfall that has not been part of their training data, leading to widespread traffic disruptions. This scenario underscores a critical vulnerability in Convolutional Neural Networks (CNNs), which are at the heart of AVs' vision systems. While CNNs excel in recognizing objects under normal conditions, their performance deteriorates when faced with novel environmental distortions like large snowflakes. The current approach of augmenting training data with various distortions is insufficient for achieving robustness against all potential unknown corruptions.

The paper's motivation extends beyond AVs, addressing a broader issue in machine learning: the generalization of CNNs beyond their training domain. CNNs often struggle with images that deviate slightly from their training set, whether it's unusual object poses or minor distributional changes. While model robustness has mostly been studied for classification tasks, object detection models have not been scrutinized for

Figure 3.2: Visualization of COCO and Stylized-COCO. The three different training settings are: standard data (top row), stylized data (bottom row) and the concatenation of both (termed 'combined' in plots). Adopted from Michaelis et al. [47].

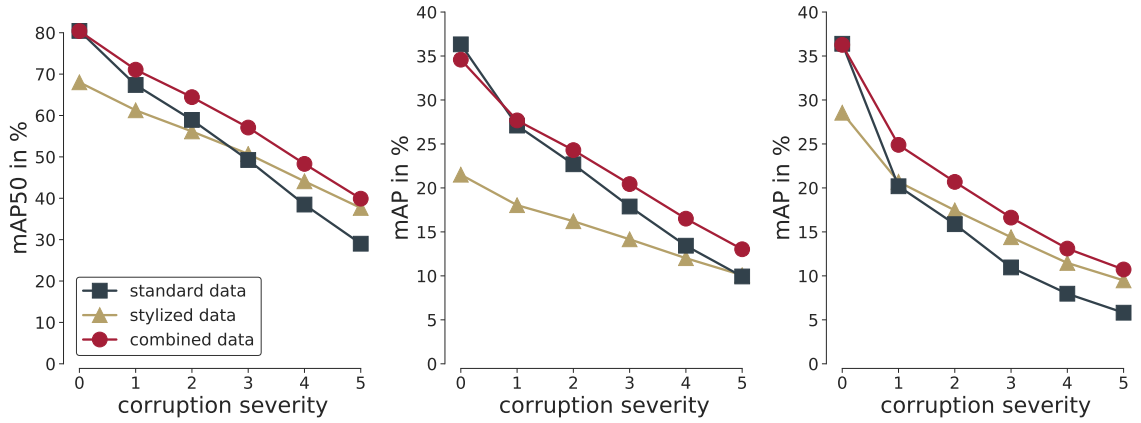
their distortion robustness. To address this gap, a novel approach was proposed for evaluating the robustness of detection models against a broad range of distortions not present in their training data. Three benchmark datasets—PASCAL-C, COCO-C, and Cityscapes-C—were introduced to provide a practical method for assessing distortion robustness in object detection. These datasets consist of modified versions of the original object detection datasets, corrupted with 15 distinct distortions at varying levels of severity.

To facilitate this, the code from the original ImageNet-C benchmark was generalized to support arbitrary datasets and released as a pip-installable Python package. Evaluation of conventional object detection algorithms on these benchmark datasets showed that a simple data augmentation technique—applying stylization [117] to training images—substantially improved robustness across corruption types, severity levels, and datasets.

In essence, this work aimed to bridge the gap between the current capabilities of CNNs and the real-world demands of visual recognition applications. Developing models capable of withstanding diverse image corruptions represents a critical step toward deploying machine learning systems, such as autonomous vehicles, safely in unpredictable real-world environments.

Robust Detection Benchmark The Robust Detection Benchmark was inspired by the ImageNet-C benchmark [46] and focuses on assessing the robustness of object detection models against corrupted images. The benchmark provides 15 corruptions across five severity levels which cover a broad range of different types, including noise, blur, digital, and weather-related corruptions. It is essential to note that these corruption types are not meant to be used as augmentations during training. Instead, they serve to measure a model's robustness against previously unseen corruptions. For model validation, four separate corruptions are provided: Speckle Noise, Gaussian Blur, Spatter, and Saturate.

Style transfer as data augmentation In the context of image classification, style transfer—a technique that combines the content of one image with the style of another



(a) Pascal-C **(b)** Coco-C **(c)** Cityscapes-C

Figure 3.3: Object detection results on different robustness benchmarks. Adopted from Michaelis et al. [47].

[117]—has been shown to significantly improve corruption robustness. This approach was extended to object detection datasets, exploring two settings. In the first setting, all training images were replaced with their stylized versions. Stylization was performed using the AdaIN (Adaptive Instance Normalization; 123) method, where the original texture information was substituted with randomly selected textures from Kaggle's Painter by Numbers dataset [124]. In the second setting, stylized versions were added alongside the original training images. Exemplary stylized images are presented in Fig. 3.2.

Results Using the introduced robustness benchmarks, it was first shown that performance degrades under corruption across a wide range of models. Additionally, robustness was observed to increase with backbone capacity and in-distribution performance. A broader discussion of the relationship between in- and out-of-distribution performance is provided in Chapter 4.3.

Stylization enhanced corruption robustness in object detection across different benchmarks (see Fig. 3.2). To evaluate the relevance of synthetic benchmarks for real-world datasets, training data was augmented with stylized images on two real-world benchmarks: BDD100k [125] and Foggy Cityscapes [126]. Improved performance is observed across various shifts and severity levels in these real-world benchmarks when stylization was applied for data augmentation.

Discussion This work demonstrated that object detection models experience substantial performance degradation when confronted with corrupted images. This effect, previously observed in image recognition models, motivated the creation of the Robust Detection Benchmark, which includes three user-friendly datasets: PASCAL-C, COCO-C, and Cityscapes-C. Evidence suggests that performance on these benchmarks correlates with robustness to natural distortions. Furthermore, a straightforward data augmentation technique—adding stylized copies of training data—effectively improves robustness.

Potential extensions to this benchmark include incorporating additional distribution shifts or more complex corruption types, such as realistic snow or fog, and expanding to other tasks, such as semantic segmentation. Future studies could investigate whether

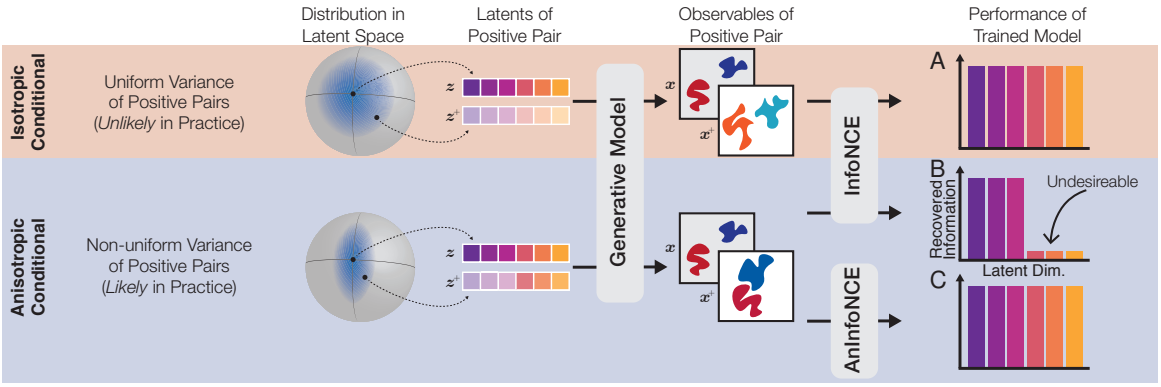


Figure 3.4: Mismatch Between Standard Contrastive Learning Model and Practical Scenarios: A: In practice, the commonly used InfoNCE objective in CL is identifiable only when all latents change to the same extent across the positive pair [99]. However, this scenario is unlikely to occur. **B:** A more realistic situation arises when augmentations affect different latents to varying degrees. In such cases, there is a risk of dimensional collapse and information loss. **C:** To address this, the Anisotropic InfoNCE (AnInfoNCE) objective was proposed. AnInfoNCE models features that can vary to different degrees within the positive pair, preventing collapse. Adopted from Rusak et al. [128].

similar patterns hold for more advanced backbone architectures, such as transformers [127]. Only one robustness intervention (stylization) was evaluated in this work; a more comprehensive analysis would require testing additional popular data augmentation techniques, such as AugMix [87] or DeepAugment [49].

3.3 P3: InfoNCE: Identifying the Gap Between Theory and Practice.

Evgenia Rusak*, Patrik Reizinger*, Attila Juhos*, Oliver Bringmann, Roland S. Zimmermann, Wieland Brendel. Contrastive Learning: Reducing the Gap Between Theory and Practice. *AISTATS 2025*

The full paper can be found in the Appendix A, starting from page 124.

Author contributions WB conceived the project idea. ER led the project. ER, RSZ, and WB designed the experiments and ER executed them. ER conducted the analysis with support from RSZ and PR. ER and RSZ jointly created all figures. PR conceived how to incorporate anisotropy in SimCLR. This idea led to WB and RSZ developing the main theorem. AJ developed corollaries for hard negative sampling and loss ensembling with feedback from PR and RSZ. PR conceived the learning dynamics analysis with feedback and corrections from RSZ. RSZ and WB jointly supervised the project. AJ was responsible for presenting the theoretical results in a standardized way. ER, PR, AJ, RSZ, and WB contributed to writing the manuscript. Senior support was provided by OB, RSZ and WB.

Motivation This work is situated in Self-Supervised Learning (SSL) theory, specifically in Contrastive Learning (CL), where the connections between theoretical assumptions and guarantees and experimental practice are investigated. CL is a very popular training objective [129] where the representational similarity between two augmentations of the same image is maximized while the similarity to augmented views of other images is minimized. The similar image pairs are referred to as the *positive pairs* while the dissimilar images are referred to as the *negative pairs*. Trained with the CL objective, the network learns to cluster images of the same class close together without label supervision.

The InfoNCE objective (introduced by Oord et al. [130]) is commonly used in Contrastive Learning (CL). Zimmermann et al. [99] demonstrated that, under specific assumptions about the Data Generating Process (DGP), a model trained with InfoNCE can recover ground-truth latent factors. However, their identifiability result relies on an isotropic change across all latent factors within the positive pair. In a subsequent study, von Kügelgen et al. [131] relaxed the isotropic assumption by partitioning positive conditional distributions into two categories: content (δ distribution) and style (non-degenerate distribution). Although this relaxation allows for more flexibility, it comes at the cost of a weaker block-identifiability result.

In this work, it was argued that neither of the above assumptions, i.e. that all latent factors vary to the same degree or that there exist two categories of content and style latents are realistic in practice. In real-world datasets, views for CL are generated using augmentations. Typically used augmentations include strong cropping, color jitter, setting the image to gray scale with a certain probability and Gaussian blurring. These augmentations correspond to changes in latent factors within an underlying Latent Variable Model (LVM), as modeled by the positive conditional distribution. The impact of these augmentations on latent factors varies. For instance: Strong cropping may significantly alter some latent factors as it discards a large portion of the image. Gaussian blurring, applied half of the time, moderately affects other latent factors (e.g., object sharpness) while latents related to class information remain largely unchanged.

It was therefore proposed that augmentations induce anisotropic changes across different latent factors. This setting has not yet been covered theoretically. Despite identifiability guarantees from prior work [99], [131], empirical studies reveal that CL still loses information [129], [132]. To bridge this gap, Anisotropic InfoNCE (AnInfoNCE) was introduced, a variation of InfoNCE which demonstrates identifiability for latent factors which vary on a continuum. The results were validated on synthetic and real-world datasets. Additionally, a discussion was included which extends the theory to address challenges like hard negative mining and loss ensembles.

The InfoNCE loss The InfoNCE loss is a popular training objective in SSL where the cosine similarity between the positive pair $(\boldsymbol{x}, \boldsymbol{x}^+)$ is maximized and the cosine similarity between the negative pair $(\boldsymbol{x}, \boldsymbol{x}^-)$ is minimized. Positive pairs are drawn from a conditional distribution centered around the observation $\boldsymbol{x}^+ \sim p(\boldsymbol{x}^+|\boldsymbol{x})$ while dissimilar points are drawn uniformly from the dataset: $\boldsymbol{x}^- \sim^{iid} p(\boldsymbol{x})$. The InfoNCE loss takes the form:

$$\mathcal{L}_{\text{InfoNCE}} = \mathop{\mathbb{E}}_{\substack{\boldsymbol{x},\boldsymbol{x}^+ \\ \{\boldsymbol{x}^-\}}} \left[-\ln \frac{e^{\boldsymbol{f}^\top(\boldsymbol{x})\boldsymbol{f}^\top(\boldsymbol{x}^+)/\boldsymbol{\tau}}}{e^{\boldsymbol{f}^\top(\boldsymbol{x})\boldsymbol{f}^\top(\boldsymbol{x}^+)/\boldsymbol{\tau}} + \sum_{i=1}^{M} e^{\boldsymbol{f}^\top(\boldsymbol{x})\boldsymbol{f}^\top(\boldsymbol{x}_i^-)/\boldsymbol{\tau}}} \right], \tag{3.1}$$

where $\boldsymbol{\tau}$ is the temperature and $\boldsymbol{f}$ is the trainable encoder. It is assumed that the observations are generated by an underlying data generative process (DGP) $\boldsymbol{g} : \mathcal{Z} \to \mathcal{X}$. Then, the samples follow a marginal distribution $p(\boldsymbol{z})$ over the latent space $\mathcal{Z}$ [99]. A change in the latent variable $\boldsymbol{z}$ manifests in an image $\boldsymbol{x}$ via the decoder $\boldsymbol{g}$. Zimmermann et al. [99] showed that, under certain assumptions, CL inverts the DGP, i.e., the composition $\boldsymbol{h} = \boldsymbol{f} \circ \boldsymbol{g}$ is a trivial map.

The AnInfoNCE loss A crucial assumption for the identifiability proof in Zimmermann et al. [99] is that the positive conditional distribution follows an isotropic van-Mises-Fischer distribution, see Fig. 3.4 (top row). This corresponds to latents which are varied to the same extent within the positive pair $(\boldsymbol{z}, \boldsymbol{z}^+)$. Arguably, augmentations usually employed in CL do not change all latent factors evenly, but rather on a continuum. In this setting, when one has an anisotropic positive conditional distribution, the model only learns a subset of the latent factors when trained with the InfoNCE loss, see Fig. 3.4 (middle row). Therefore, a modified version of the InfoNCE loss (the AnInfoNCE loss) was developed which accounts for latents which can vary on a continuum. The AnInfoNCE loss takes the form:

$$\mathcal{L}_{\text{AnInfoNCE}} = \mathop{\mathbb{E}}_{\substack{\boldsymbol{x},\boldsymbol{x}^+ \\ \{\boldsymbol{x}^-\}}} \left[-\ln \frac{e^{-\|\boldsymbol{f}(\boldsymbol{x}^+)-\boldsymbol{f}(\boldsymbol{x})\|^2_{\hat{\boldsymbol{\Lambda}}}}}{e^{-\|\boldsymbol{f}(\boldsymbol{x}^+)-\boldsymbol{f}(\boldsymbol{x})\|^2_{\hat{\boldsymbol{\Lambda}}}} + \sum_{i=1}^{M} e^{-\|\boldsymbol{f}(\boldsymbol{x}_i^-)-\boldsymbol{f}(\boldsymbol{x})\|^2_{\hat{\boldsymbol{\Lambda}}}}} \right], \tag{3.2}$$

where *the concentration* $\hat{\Lambda}$ is a trainable diagonal matrix which scales the learned latent factors and the weighted and squared ℓ_2 distance, $-\|\boldsymbol{f}(\boldsymbol{x}^+) - \boldsymbol{f}(\boldsymbol{x})\|^2_{\hat{\boldsymbol{\Lambda}}} = -(\boldsymbol{f}(\boldsymbol{x}^+) - \boldsymbol{f}(\boldsymbol{x}))^\top \hat{\Lambda}(\boldsymbol{f}(\boldsymbol{x}^+) - \boldsymbol{f}(\boldsymbol{x}))$, is the *similarity function.* The trainable parameter $\hat{\Lambda}$ models the ground-truth positive conditional distribution which is assumed to follow

$$p(\boldsymbol{z}^+|\boldsymbol{z}) \propto e^{-(\boldsymbol{z}^+-\boldsymbol{z})^\top \Lambda (\boldsymbol{z}^+-\boldsymbol{z})}, \tag{3.3}$$

where a higher value of Λ_{ii} means a higher concentration of $\boldsymbol{z}^+$ around the anchor $\boldsymbol{z}$ along dimension i. This positive conditional distribution models a multi-variate Gaussian distribution around an anchor point, see Fig. 3.4 (bottom). This means that different latent factors can vary to a different degree which is more realistic in practice. The main contribution of this paper is the Theorem 3.1. which proves the identifiability of anisotropic CL:

Theorem 1 *(Theorem 3.1 in [128]) If a pair* $(\boldsymbol{f}, \hat{\Lambda})$ *minimizes* $\mathcal{L}_{AnInfoNCE}$*, then* $\boldsymbol{f} \circ \boldsymbol{g}$ *is a block-orthogonal transformation, where each block acts on latents with equal weight* Λ_{ii}*. In other words,* $\boldsymbol{z}$ *is identified up to a block-orthogonal transformation.*

Theorem 1 states that if the encoder $\boldsymbol{f}$, typically parameterized by a neural network, is expressive enough to minimize the loss function $\mathcal{L}_{AnInfoNCE}$, then the reconstructed latent representation $\hat{z} = \boldsymbol{f}(\boldsymbol{x})$ is connected to the ground-truth representation $\boldsymbol{z}$ through a straightforward orthogonal linear transformation. This latter ambiguity is inconsequential because solving any downstream task usually incorporates at least one linear layer trained on top of the backbone $\boldsymbol{f}$. Although the assumptions with respect

to previous work were relaxed, the proof still relies on several assumptions. Please refer to the paper in Appendix A for a full overview over the assumptions required for Theorem 3.1.

AnInfoNCE recovers more latent factors in experiments Extensive experiments were conducted, both on synthetic and real-world data to validate AnInfoNCE. On synthetic data where full control over the DGP is available, it was observed that AnInfoNCE recovered all latent factors while training with InfoNCE led to information loss if the ground truth positive conditional was set to an anisotropic distribution. On real-world datasets (CIFAR10 and ImageNet), it is impossible to explicitly calculate the recovery of ground truth factors of variation because these are unknown. For this reason, the recovery of latent factors was estimated via calculating the linear readout accuracy on the augmentations used during training to judge how well the latent factors were recovered. When training with AnInfoNCE compared to training with InfoNCE, much better results on the augmentations readout accuracy were observed. However, this came at the cost of classification accuracy as the model trained with AnInfoNCE had lower classification accuracy compared to the model trained with InfoNCE.

Taken together, AnInfoNCE outperformed InfoNCE on synthetic and well-controlled data, where all latent factors could be recovered, while training with InfoNCE resulted in information loss. However, when augmentations were used to generate views, a trade-off arose between accuracy and identifiability. Further investigation indicated that this trade-off was caused by the augmentations themselves, rather than by the specific dataset or model architecture. Given an underlying latent variable model, augmentations manifest themselves as variations in the latent space, which are modeled by a positive conditional distribution. This observation highlights a potential reason for unexpectedly low performance of AnInfoNCE when augmentations are used: the conditional distribution inferred from the augmentations might not align with the one assumed by the loss function. Specifically, the implicitly defined form of the conditional distribution through augmentations may not follow a Gaussian or even uni-modal distribution. Additionally, high concentration parameters associated with these augmentations can lead to a flat loss landscape and practical optimization challenges.

Discussion This paper investigated the connection between SSL theory and practice. From the theory perspective, there is a connection between the learned representations and the ground truth latent factors of variation when a model is trained with the InfoNCE loss. However, in practice, there is information loss in the final outputs of the model, even necessitating the use of a projection head: Usually, one discards the final layer or even multiple layers of a trained model and relies on intermediate layers for a downstream task. The implicit assumption of an isotropic positive conditional distribution is unrealistic in practice which led to the modification of the InfoNCE loss to allow for anisotropic positive conditional distributions. Although improvements were observed on synthetic and well-controlled data, no corresponding benefits were seen in downstream accuracy on real-world datasets, indicating that a gap remains. It would be a fruitful endeavor to analyze this gap further and to try to understand which other modifications to the InfoNCE objective are necessary to better align it with the practice of using augmentations. Augmentations imply a certain change in the ground truth latents: Which form does the positive conditional distribution need

to take in order to model these changes in the latent space? Alternatively, it might be possible to design a different set of augmentations which fit better to the currently assumed Gaussian conditional distribution. For example, one used augmentation is setting the image to gray scale 20% of the time; this augmentation cannot be modeled by a Gaussian distribution since it is a binary process. It could be possible to design a set of augmentations which are easier to model.

3.4 P4: Improving Robustness against Common Corruptions by Covariate Shift Adaptation.

Steffen Schneider*, **Evgenia Rusak***, Luisa Eck, Oliver Bringmann, Matthias Bethge, Wieland Brendel. Improving robustness against common corruptions by covariate shift adaptation. *NeurIPS 2020.*

The full paper can be found in the Appendix A, starting from page 160.

Author contributions StS conceived the project and proposed the original formulation of batch norm adaptation, WB suggested the partial adaptation setting. StS and ER implemented and conducted the experiments with input from WB and MB. StS prepared visualizations and analysis with input from ER and WB. StS and LE conducted the theoretical analysis. ER, WB, StS and MB wrote the manuscript, all authors participated in reviewing and editing. StS and ER contributed equally to the project. WB, MB and OB contributed equally to advising the project.

Motivation This paper explored how limited access to unlabeled test data can enable models to adapt to distribution shifts at test time. Similar to how human vision adjusts in darkness by widening the pupils, or how the auditory system strains to discern a normally quiet speaker, models may benefit from mechanisms that allow adaptation to encountered distribution shifts.

So far, popular methods for evaluating DNN robustness have assumed the model has no prior knowledge of the distortions it encounters, even if it sees the same distortion repeatedly. This is not how real-world scenarios play out. Imagine a self-driving car encountering a sandstorm. The car would experience this change consistently over many images, not just once. Ideally, the model should be able to adapt.

A research field called domain adaptation (DA) focuses on helping models work effectively when the input data changes (like going from clean images to distorted images). Surprisingly, DA techniques haven't been widely tested on the benchmarks used to measure robustness. This paper aimed to bridge this gap, by diving into a common DNN technique called batch normalization (BN, [133]). BN relies on statistics calculated from the training data. The issue is that these statistics may not be accurate for corrupted images. A simple solution is proposed: recalculating the batch normalization (BN) statistics using the corrupted images themselves. This enables the model to adjust to the distribution shift and substantially improves performance.

In a nutshell, this paper explored how techniques from domain adaptation can significantly improve the robustness of DNNs. By incorporating these methods, DNNs can

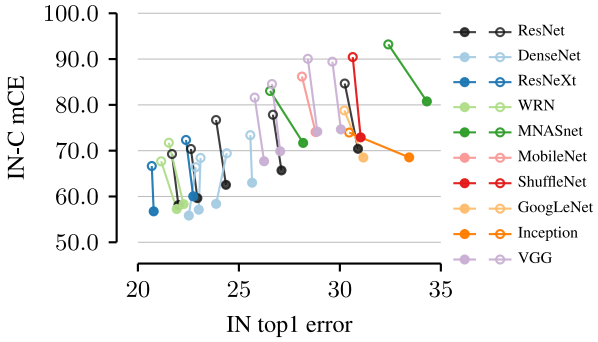


Figure 3.5: BN adaptation improves performance. Across 25 model architectures in the `torchvision` library, the baseline mCE (○) improves with adaptation (●), often on the order of 10 percent points. Adopted from Schneider et al. [134].

become more adaptable and function better in real-world conditions where variations and distortions are inevitable.

Batch-Normalization during Training and Testing Batch-Normalization (BN) layers have been introduced by Ioffe and Szegedy [133] as a means to stabilize training and increase convergence speed. BN layers within a DNN calculate averages (mean) and spread (variance) of data within each feature map. During training, these statistics (mean, variance) are calculated for each small batch of data. During testing, the model normally uses statistics gathered over the entire training dataset. The issue arises when the data the model sees during testing differs from its training data. If the statistics change, the feature maps will not be accurately adjusted. In this case, the activations of each subsequent feature map are no longer normalized to zero mean and unit variance, breaking a crucial assumption that all downstream layers depend on. In the end, this violated assumption accumulates over all network layers and results in worse downstream performance.

Mean Corruption Error (mCE) The mean Corruption Error (mCE), introduced in the ImageNet-C benchmark [46], is a standardized metric for evaluating model robustness to common input corruptions such as noise, blur, weather effects, and compression artifacts. Each corruption type is applied at five severity levels, and a model's classification error across these levels is normalized by the corresponding error of AlexNet [2] to yield a corruption error (CE) per corruption type. The mCE is then computed as the average CE across all corruptions, with lower values indicating higher robustness. This normalization allows fair comparison between models and highlights relative performance under distribution shifts.

Unsupervised Adaptation of BN statistics improves performance. Strong performance improvements were observed across a wide range of architectures when the batch normalization (BN) statistics were recalculated at test time (see Fig. 3.5). Performance gains were seen for both vanilla-trained models and models trained with robustness interventions, demonstrating the general effectiveness of this approach. This result highlights the benefit of this simple technique for different pretrained models.

The quality of the batch normalization (BN) statistics depends on the batch size used for their estimation. To simulate scenarios with limited test samples, the dependence of

performance on batch size was examined. Benefits from BN adaptation were observed even when test-time statistics were estimated from as few as eight samples, with improvements increasing as batch size grows until saturating around a batch size of approximately 100.

Extending this approach, an adaptation regime with very few samples was considered, investigating whether BN adaptation can be effective when only a single novel data point was available at test time. In this case, the training statistics were used as a prior, and the test statistics were estimated as a weighted linear combination of the training and test statistics. This method enabled adaptation benefits even with just a single sample.

Discussion A simple domain adaptation technique was explored here to improve performance under test-time distribution shifts. Strong performance improvements were observed on ImageNet-C across the full range of tested models. While results on this robustness benchmark were promising, limitations of the technique remain, and opportunities exist for further improvement.

First, the idea of recalculating the BN statistics at test-time is only successful for simple, low-level shifts, such as those present in ImageNet-C. ImageNet-C is comprised of corrupted versions of ImageNet validation images. Thus, high level details such as camera perspective, viewpoints or object poses have not been altered. BN adaptation can undo low-level corruptions such as blurs or noises, but has been less successful on other datasets with higher-level shifts. In particular, limited benefits on ImageNet-R [49] which contains renditions of several ImageNet classes or ObjectNet [52] which contains images of ImageNet objects taken in unusual poses in cluttered scenes were observed. Therefore, as a machine learning practitioner, it is important to understand what kind of distribution shift is present to make an informed decision whether BN adaptation is likely to succeed.

Second, the success of BN adaptation opens an opportunity for other, more advanced domain adaptation techniques. BN adaptation is the simplest method from domain adaptation which does not require computing any gradients or performing back-propagation steps. However, domain adaptation is a mature research field with established techniques which have so far been overlooked by the robustness community. A natural next step would be to apply the most successful DA method to robustness benchmarks. In fact, this approach was pursued, leading to the next project, which is discussed in the following chapter.

3.5 P5: If your Data Distribution Shifts, Use Self-learning.

Evgenia Rusak*, Steffen Schneider*, George Pachitariu, Peter Gehler, Oliver Bringmann, Matthias Bethge, Wieland Brendel. If your data distribution shifts, use self-learning. *TMLR 2022*.

The full paper can be found in the Appendix A, starting from page 195.

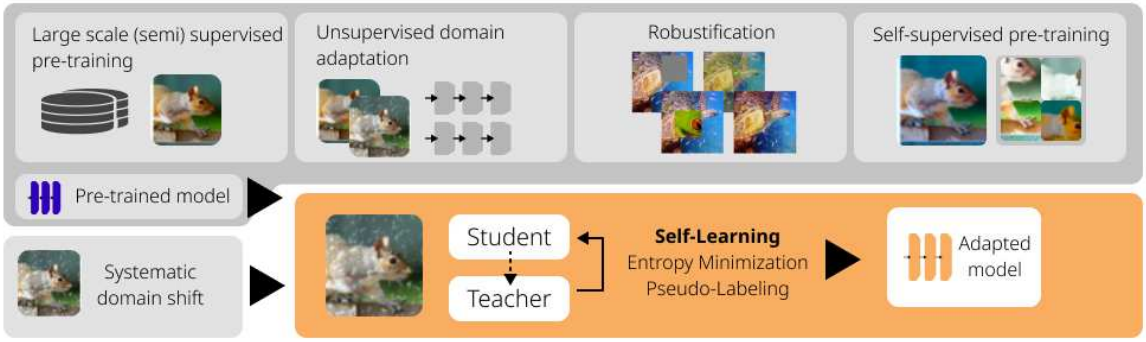


Figure 3.6: Schematic overview of the training pipeline. Traditionally, researchers have made image-recognition models more robust and adaptable to new types of data using robust pre-training (with various image changes or extra data), unsupervised domain adaptation (letting the model see unlabeled examples from the new environment), or self-supervised learning (training it on additional data). The key finding is that regardless of the model's architecture, size, or how it was initially trained, one can always further improve its ability to handle a new environment by applying simple self-learning techniques. Adopted from Rusak et al. [135].

Author contributions StS and ER conceived the project with suggestions from MB and WB. ER ran initial experiments. StS performed large scale experiments on ImageNet-C,R,A with support from PG, ER and WB. ER performed domain adaptation and CIFAR experiments with suggestions from StS. ER implemented all baselines that have been included as comparisons in the manuscript. GP ran DINO adaptation, WILDS, and self-ensembling experiments with suggestions from ER & StS. ER developed the ImageNet-D dataset with suggestions from StS and WB, and performed the large scale experiments on ImageNet-D. LE and StS developed the theoretical model, ER contributed the CIFAR-C experiments. WB, ER and StS wrote the initial manuscript with help from all authors. ER wrote an extensive related work section. ER reviewed and edited the manuscript in the course of the author-reviewer discussion period. Senior support, funding acquisition and infrastructure were provided by O.B., M.B. and W.B.

Motivation This work extended the previous approach discussed in P4, in which a model was allowed to adapt to shifted distributions at test time. A simple domain adaptation technique was employed, consisting of adjusting the batch normalization (BN) statistics to the shifted distribution. In the present study, this approach was taken further by applying a more powerful domain adaptation method to popular robustness benchmarks.

So far, the robustness community has been using an ad-hoc evaluation setting for benchmarking model robustness: Models are trained on large datasets (like ImageNet) with various modifications (data augmentation, pre-training) to make them more resilient to unforeseen changes. However, the models aren't specifically adapted to any particular change they might encounter.

On the other hand, in the community of domain adaptation, models are trained to perform well both on a labeled source and an unlabeled target dataset. The most common domain adaptation setting is Unsupervised Domain Adaptation (UDA) which

I discussed in Chapter 2.5.2. This approach focuses on smaller datasets where both the training data and the unseen test data (with a different distribution) are available. The model can then adapt to the test data during training. This method performs well, but it requires access to both sets of data, which may not always be possible (e.g., pre-trained models with unknown source data).

This paper explored an intermediate approach—**source-free domain adaptation** [136]. Here, the model can adapt to the new, unseen test data without needing the original training data. This is relevant for many real-world applications: Self-driving cars adapting to changing weather conditions without needing to be trained on every possible scenario, image-based quality control software adapting to different lighting conditions at various locations or medical imaging systems performing well on scanners different from the ones used for training.

A key component of many domain adaptation techniques is some form of *self-learning*. The term "self-learning" dates back to 1968 [137] and describes how a system can learn without being told directly if its responses are right or wrong. This label was chosen here for methods like pseudo-labeling [138, 139], robust pseudo-labeling [140], entropy minimization [141, 142], and others because they work in a similar way. These techniques allow the model to adapt and improve even without having the correct "answers" (labels) for the new data it encounters. The paper explored various self-learning methods in this source-free domain adaptation setting and demonstrated that they significantly improve performance across different benchmarks and models, see Fig. 3.6. Self-learning based methods can be further enhanced when combined with other techniques, such as e.g., consistency and diversity regularization [143].

Self-Learning Universally Improves Model Performance This paper presented an extensive empirical study analyzing the benefits of various self-learning techniques across different datasets, model architectures, and pretraining schemes. Self-learning was found to consistently improve model performance, regardless of pretraining status. While previous approaches showed that BN adaptation did not enhance performance for models pretrained on large-scale datasets, self-learning demonstrated benefits even in this setting.

Notably, self-learning further improved domain-adapted models that have already been exposed to the target domain, effectively fine-tuning them to the target distribution.

A large-scale hyperparameter analysis was conducted to identify critical experimental design choices for successful self-learning. For instance, only adapting the affine batch normalization parameters was essential for convolutional neural networks, whereas full model adaptation outperformed affine-only adaptation for vision transformers.

In order to test the models on a wider range of distribution shifts, the datasets from the DomainNet benchmark [144] were adapted to be compatible with ImageNet-trained models. This benchmark consists of six constituent datasets, but has a different label hierarchy compared to ImageNet. For example, ImageNet has more than 200 individual dog breeds while DomainNet only has one (parent) dog class. Thus, to make ImageNet models compatible with DomainNet, it was necessary to map predictions made by ImageNet-trained models to the DomainNet label space. A smaller version of DomainNet which is compatible with ImageNet-trained models out-of-the box (ImageNet-D) was released.

Best practices for test-time adaptation. Based on the extensive experimental results, a set of "best practices" when using self-learning for test-time model adaptation was formulated. A rigorous cross-validation of crucial hyperparameters was recommended on the holdout set of ImageNet-C, with the best-performing values subsequently applied when adapting the model to other datasets. Implementing simple baseline models for comparison was also advised, as these have been shown to perform on par with more sophisticated methods when hyperparameters are fairly tuned. Since test-time adaptation consistently improves performance, using the most robust baseline model as a starting point and further enhancing it with self-learning is effective. Key hyperparameters identified for tuning include the adaptation learning rate, the number of epochs, and the adaptation parameters of the model (i.e., affine BN parameters or all model parameters).

Discussion This study investigated how self-learning, a technique commonly employed in domain adaptation and self-supervised training, could enable image recognition models to adapt to new challenges. State-of-the-art results were achieved on several image corruption benchmarks (ImageNet-C, -R, and -A), and a new, more challenging benchmark (ImageNet-D) was introduced. Finally, guidance for selecting appropriate hyperparameter values was provided.

Overall, self-learning was found to be highly effective in improving model performance across diverse scenarios, including different datasets, model architectures, and training methods. A limitation is that these methods can sometimes become unstable during longer adaptation periods. Press et al. [145] analyze this setting and show that most current test-time adaptation techniques collapse to random-chance performance if the adaptation time is long enough. Further, they propose a new continual learning benchmark where the distribution shifts over time. This setting was not covered in this work as a stationary distribution shift was assumed which does not change over time. Press et al. [145] show that popular self-learning techniques struggle with non-stationary distributions. Their proposed solution is to reset the model weights in periodic intervals to their original values to counteract collapse.

A potential extension of this work is to combine self-learning with other techniques for further improvements. For example, training the model with the Soft Likelihood Ratio loss (SLR [146]) performs better than the simple self-learning setting presented in this study on the highest severity of ImageNet-C. SLR is an extension of entropy minimization where the entropy minimization loss is replaced with a version to ensure non-vanishing gradients of high confidence samples; in addition, there is a penalty to encourage diverse predictions and a module to partially adjust for changes between the training and test data. The success of SLR over simpler self-learning methods highlights the potential benefits of extending self-learning with additional objectives. It also reinforces the findings on the general effectiveness of self-learning approaches presented here. However, to test the generality of SLR, a similarly extensive study across multiple datasets, pretraining schemes and model architectures would be necessary.

4 Discussion

This work has discussed several methods for improving the out-of-distribution (OOD) generalization capabilities of deep neural networks (DNNs). In this chapter, a broader perspective is adopted to examine expectations regarding OOD generalization, informed by the current understanding of the underlying challenges. The discussion covers insights emerging from progress on widely used OOD benchmarks, as well as the implications of the advent of Foundation Models [147] for OOD generalization requirements and benchmarking. Overall, the OOD generalization community has experienced a significant shift in recent years, which is elaborated upon in the following sections. The first addressed topic is the determination of a reasonable upper bound on OOD generalization performance in light of the continually improving performance of DNNs.

4.1 Is there a trade-off between the human-machine alignment and the DNN's performance?

The deployment of Deep Neural Networks (DNNs) in real-world applications necessitates continued focus on enhancing performance, particularly concerning robustness to various forms of distribution shift. This persistent drive for increased capability, however, is accompanied by escalating concerns regarding potential risks. A growing body of researchers is now prioritizing the topic of AI Safety, specifically focusing on human-machine alignment and the existential risks associated with Artificial General Intelligence (AGI). For example, over 33000 researchers signed an open letter to pause giant AI experiments for at least six months [148]. Prominent AI researchers such as Geoffrey Hinton, Yoshua Bengio, Demis Hassabis and Sam Altman have signed the statement from the Center for AI Safety saying "Mitigating the risk of extinction from AI should be a global priority alongside other societal-scale risks such as pandemics and nuclear war" [149]. While critics of such gestures denounce them as mere virtue signaling without much substance in terms of concrete action, being aware and analyzing possible risks as well as taking steps to create non-malevolent AI systems seems prudent.

To actively mitigate these potential catastrophic risks, one compelling strategy is to enhance the human-likeness of DNN decision-making. The underlying premise is that if a powerful machine employs a decision strategy similar to that of a human, the risk of catastrophic misalignment, where a system achieves goals through unexpected or opaque failure modes, is substantially lowered. Critically, similarity in strategy is inferred not merely by matching overall accuracy, but by demonstrating high error consistency; the systems must make errors on the same individual inputs. This consistency is vital because it establishes a baseline of predictable failure, which serves as a crucial form of interpretability that is necessary for safety. The shape–texture

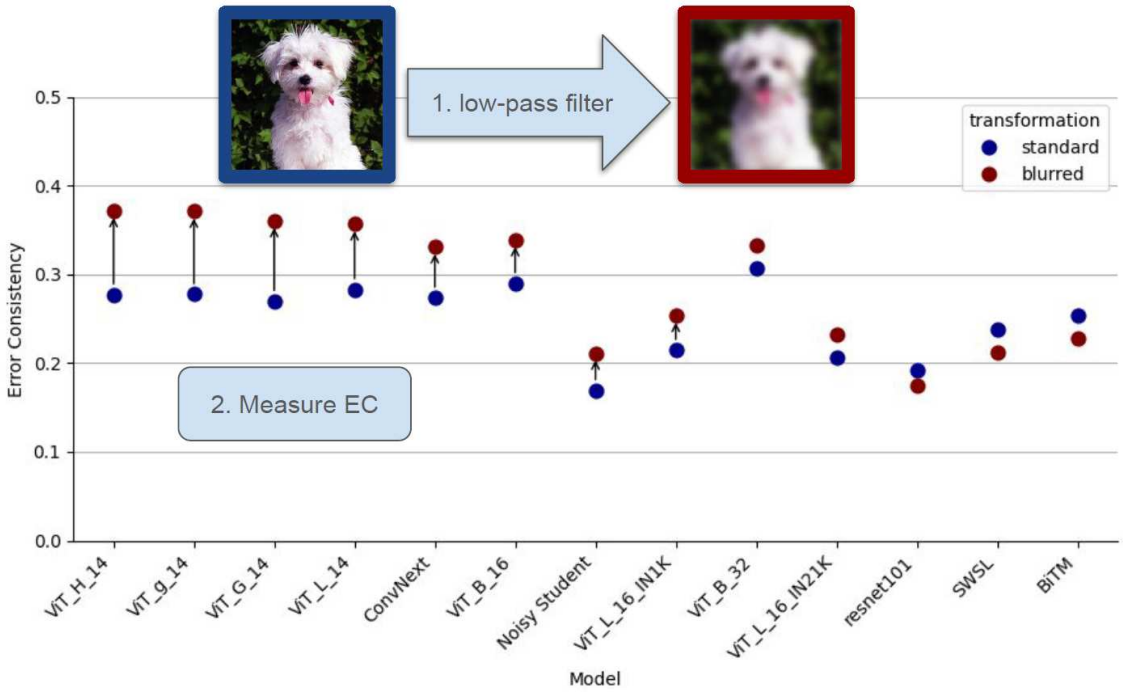


Figure 4.1: Applying low-pass filters to images before models evaluate them increases the consistency between the models' errors and those made by human observers.: For most models tested, this filtering resulted in a higher, sometimes significantly higher, rate of human-machine error consistency. *Figure adopted from Wolff et al. [152].*

bias discrepancy provides a salient example of this dangerous misalignment. In a cue-conflict classification task, Geirhos et al. [150] showed that when an image displayed the shape of a cat with the texture of elephant skin, humans predominantly selected the cat, whereas DNNs favored the elephant. This divergence in decision strategy indicates that humans prioritize object shape while DNNs decide based on object texture.

Given this, the goal from an AI safety and alignment perspective must be to engineer DNNs that minimize the error consistency gap with human observers. Such high alignment provides two key safety benefits: enhanced interpretability, by allowing us to understand and predict model failure boundaries, and a reduction in X-risk by ensuring that powerful systems operate based on human-cognizable principles. To enable the principled measurement of this alignment, Geirhos et al. [151] conducted a series of psycho-physical experiments, comparing error consistency across human participants and deep models, culminating in the release of the `modelvshuman` benchmark. Their initial findings established that despite consistency within humans and within machines, the errors made by machines were fundamentally different from human errors. Although later work [82] has shown a diminishing robustness gap, with leading models achieving superhuman accuracy on many OOD datasets, a profound image-level consistency gap remains. This persistent gap is the core measure of poor human-machine alignment, as it confirms that even highly accurate models are achieving their performance using fundamentally alien, and thus unpredictable, failure modes compared to human perception.

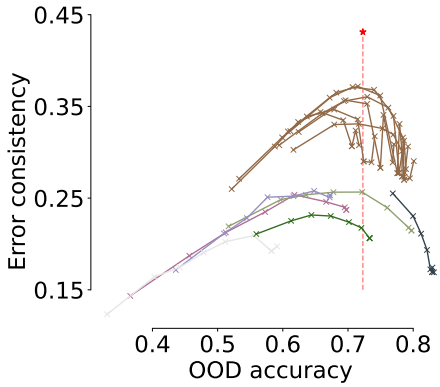


Figure 4.2: Relationship between OOD accuracy and error consistency: Models from different model families are color-coded and connected via solid lines. Human performance is demarcated with a red star. For models with subhuman OOD accuracy, error consistency and OOD accuracy are positively correlated. For models with superhuman OOD accuracy, this trend reverses. *Figure adopted from Wolff et al. [152].*

In their recent paper, Wolff et al. [152] show that the error consistency between various CLIP models [55, 153–155] and humans can be increased with simple preprocessing techniques applied to images before feeding them into the models. Contrastive Language-Image Pre-training (CLIP) is a very popular pretraining method which pairs images with their corresponding captions and contrasts them from other images and captions. In particular, Wolff et al. [152] observed that preprocessing techniques which remove high-frequency information, such as blurring, significantly improve human-machine alignment, as shown in Fig, 4.1. A possible explanation of this phenomenon is from a physiological viewpoint: The human visual system naturally filters incoming light. The human eye's optics act as an initial low-pass filter on light reaching the retina [156, 157], meaning images are not available at full spatial resolution. Further filtering of spatial frequencies occurs in the visual cortex [158, 159]. The final perceived reduction in contrast, which depends on both the spatial frequency of the input [158] and its temporal frequency [159], is described by the Contrast Sensitivity Function (CSF). The CSF's tuning changes with temporal frequency; very high temporal frequencies cause the typical band-pass filter to shift towards lower spatial frequencies, becoming a low-pass filter [159]. The authors suggest that by adding low-pass filters to vision models, their input effectively matches the input available to the human visual cortex, and their best-performing low-pass filters approximate the CSF quite well.

Incorporating a low-pass filter into the input pipeline reduces the spectral content of the input data, thereby limiting the features available for the model's decision-making process. This constraint invariably impacts the model's overall performance: Since human performance on the `modelvshuman` benchmark is not perfect, an increase in the model's error consistency with human observers naturally correlates with a decrease in the model's accuracy, since the model begins to replicate human error patterns. As shown in Figure 4.2, the relationship between OOD accuracy and error consistency exhibits an inverse U-shaped function, with the peak consistency coinciding with human-level performance. This indicates that for models operating at subhuman OOD performance, accuracy and error consistency are positively correlated. Conversely, this trend reverses for models that achieve superhuman OOD performance.

This result implies a fundamental *trade-off* between human-machine alignment and OOD accuracy: A machine achieving superhuman accuracy is likely to rely on features that humans typically do not use. This inevitably leads to differences in how the machines and humans make decisions. In the future, choosing between superhuman accuracy and human-like behavior may become a necessary trade-off for practitioners. From a human-machine alignment perspective, human OOD generalization performance might be a reasonable upper bound for DNNs.

A practical example for this trade-off is driving during strong rain. Human drivers would experience issues with properly seeing the road as well as keeping track of other vehicles, and would therefore slow down. An autonomous car with superhuman OOD performance might not experience such navigation issues and would therefore not slow down in strong rain. This however might impact other traffic participants, such as pedestrians, cyclists or non-autonomous cars which would be dangerously affected by an unusually fast and confident car in such adverse conditions.

Take-away

Human OOD performance might be a sensible upper bound on the networks' OOD performance from a human-machine alignment standpoint.

4.2 Why do DNNs generalize worse than humans?

Having discussed a possible upper bound on OOD accuracy from a human alignment perspective, it is still the case that models generalize a lot worse than humans to different kinds of distribution shift. While every distribution shift has its own flavor which informs the specificity of the particular failing mode, a few general statements about failures in OOD generalization behavior can be made.

Geirhos et al. [160] argue that DNNs often fail in OOD generalization tasks because they have learned shortcuts or so-called *spurious correlations* instead of the *intended solution*. The intended solution represents the desired decision strategy that a learning system is expected to acquire to solve a task robustly. In contrast, models relying on "shortcut" decision rules may achieve high accuracy on simplified benchmarks, but these rules often fail to generalize to real-world situations with greater variation and nuance. Recall the example from Chapter 2.5.1 about the "cow-on-the-beach" problem where the model learned to associate the object (cow) with its background (pasture) in such a profound way that it failed to recognize the cow in a different environment. The model may not even have learned to distinguish the cow from its background, or it may have learned to only pay attention to the background and ignore the cow completely. The intended solution in this case would have been to learn separate representations for the cow and the background, such that the generalization to a new environment would be successful. However, it is impossible or not scalable to formally encode such intended solutions into the learning process. Instead, the Deep Learning field has a data-centric view, in the sense that the training data determines the properties of learned representations. This particular failure mode can be explained by a limitation in the training data where cows simply often co-occur with pasture backgrounds, such that the model has no incentive to learn differentiating between the two.

Another more recent example for a shortcut solution concerns failures in composi-

tionality for vision-language models (VLMs). Yuksekgonul et al. [161] show that state-of-the-art VLMs behave like bag-of-words models, with little relational or compositional understanding. For example, the studied models are almost at chance level when tasked to pick between the correct caption given an image and its shuffled version. For instance, the model cannot distinguish between the correct caption "the horse is eating the grass" and the incorrect one "the grass is eating the horse". The explanation that the authors provide is again rooted in the training data: To minimize the training loss, it is sufficient for the model to simply learn to associate words in a caption with their occurences in the image in a bag-of-words-like manner without learning spacial relations or a causal relationship between objects.

What are possible strategies to overcome shortcut learning? Several methods on improving OOD generalization capabilities of DNNs have been described throughout this thesis. Over the last few years, the main solution to the problem of OOD generalization has emerged as to "simply use more data". In particular, we can see that a more diverse dataset is more likely to contain cows on beaches, breaking the observed co-occurrence of cows on pastures. In Yuksekgonul et al. [161], the authors show that enriching the dataset with hard negative examples, such as explicitly telling the model that the captions "the grass is eating the horse" or "the cow is eating the grass" are wrong for that particular image, has improved the model's compositional capabilities.

The observed benefits from training larger models on more data across a whole range of OOD generalization tasks have led researchers to training their models on gigantic web-scale datasets. These datasets are sometimes made public and sometimes they are not. Even if the datasets are public, understanding what they contain is cumbersome due to the sheer number of training examples which can be on the order of billions of data points. For this reason, evaluation and benchmarking of current state-of-the-art models becomes increasingly difficult and is the topic of Chapter 4.4.

Take-away

DNNs generalize worse than humans because of spurious shortcut features present in the training data. So far, the best solution has been to get a larger and more diverse training dataset which decorrelates the spurious features.

4.3 Can we predict OOD performance from ID performance in vision models?

In the domain of image classification, several large-scale empirical studies have shown that in-distribution (ID) performance is highly predictive of out-of-distribution (OOD) performance under a variety of dataset shifts. Taori et al. [81] conduct a systematic meta-analysis across model architectures, loss functions, and pretraining schemes, and find that ID accuracy is strongly correlated with OOD accuracy across diverse OOD benchmarks. Remarkably, the relationship is approximately linear in most settings. This finding implies that as a practitioner, one should choose the model with the lowest in-distribution validation loss which still fits one's computational requirements as it is the most likely candidate to perform well on the OOD task. Their study also shows that the only intervention that consistently improves OOD robustness is training on

more, and more diverse, data. Models pretrained on larger and more varied datasets exhibit noticeably better OOD generalization, suggesting that greater data diversity leads to more general, transferable representations.

The accuracy-on-the-line phenomenon which describes the linear relationship between ID and OOD accuracy in vision models has been investigated by Miller et al. [162] in more detail. They empirically show this effect for a wide range of models and distribution shifts. In contrast, Wenzel et al. [163] observe that ID and OOD accuracy only show a clear linear trend on specific tasks. On many other tasks, ID and OOD accuracy are only marginally positively correlated or not correlated at all. Wenzel et al. [163] do not report any trade-off between accuracy and robustness and therefore, conclude that ID accuracy can serve as a reasonable proxy for expected OOD generalization. Other works demonstrating positive correlation between gains on ImageNet and gains on OOD datasets, primarily achieved through better and bigger architectures as well as larger pretraining datasets, include Kornblith et al. [164], Kolesnikov et al. [165], Mahajan et al. [166], Xie et al. [167], Sun et al. [168].

A robust positive correlation between ID and OOD performance is surprising, given our discussion on shortcut solutions and spurious correlations in Chapter 4.2. Better ID performance may involve associating an object with its background which may not hold in a shifted distribution. Indeed, Teney et al. [169] report cases where inverse correlations between ID and OOD accuracy can be observed and explain them with the presence of spurious features. They argue that other researchers missed these inverse correlations due to a flawed experimental design and argue that the ID-OOD relationship should be studied per model over the course of its training. They show that models trained with a diversity-inducing method exhibit the ID-OOD trade-off throughout their training. Teney et al. [169] caution future robustness researchers to not only use ID accuracy as a proxy for generalization performance as has been advised by multiple previous studies. Similarly, Fang et al. [170] observe that better ImageNet performance does not translate to better performance on real-world datasets. They explain their finding with a higher distributional distance between their real-world datasets and ImageNet, as opposed to the previously used OOD datasets which have been more similar to ImageNet in comparison.

Take-away

In vision tasks, in-distribution accuracy is often (though not universally) a reasonable proxy for expected OOD performance. The strength of this relationship increases when the OOD data is drawn from a distribution that is closely related to the ID data.

4.4 Do foundation models generalize better than conventional ones?

Before the advent of *foundation models* [147], the world was a simpler place: When testing OOD generalization, the training and test distributions were strictly disjoint, and the training datasets generally had a more manageable size. If one wanted to test whether a model could generalize to sketches, sketches could be excluded from the training set. If one wanted to test whether a model could adapt to sketches, one

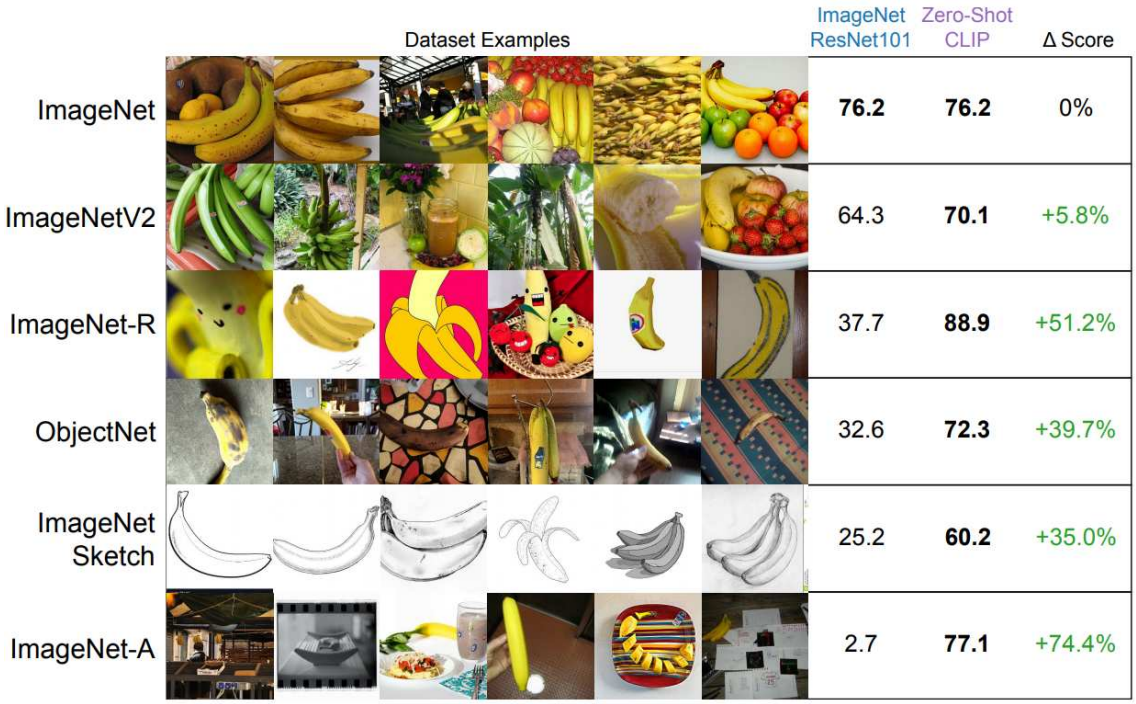


Figure 4.3: The CLIP-trained ViT-L is much more robust to distribution shifts compared to the ImageNet-trained ResNet101: Although both models have the same accuracy on ImageNet, the ImageNet-trained ResNet101 performs much worse on various shifted datasets compared to the ViT-L which was trained with CLIP. *Figure adopted from Radford et al. [55], Fig. 13*

could allow limited and well-controlled access to a few sketches at test time for model adaptation. Given that ImageNet (mostly) contains natural photographs, this clear distributional separation happened naturally. All studies described in this thesis were carried out using this paradigm. In P1 and P2, it was assumed that there is no access to the test distribution at training time and data augmentation techniques were used to learn a model which would be as general as possible. In P4 and P5, the model was allowed limited access to unlabeled test data to adapt a portion of the model's parameters. In all cases, the distinction between how the model interacts with the training versus the test data was defined in a very clear and predetermined way.

This distinction between the train and test distributions becomes rather ephemeral when dealing with today's foundation models [147]. A foundation model is a large-scale machine learning model (often a neural network, particularly a Transformer [127]) that is pre-trained on a vast, diverse, and often unlabeled dataset (frequently referred to as web-scale data). These models serve as the starting point for more advanced and specialized models. They can handle a wide variety of tasks, from translating text to analyzing medical images, and thereby provide the groundwork upon which more sophisticated models are built.

Contrastive Language-Image Pre-training (CLIP) [55] is a prominent foundation model whose efficacy stems from its exceptional performance in mitigating the effects of natural distribution shifts. This superior robustness, as measured across various OOD benchmarks (e.g., ImageNet-A, ImageNet-R, and ImageNet Sketch), significantly surpasses that of models trained exclusively on fixed, category-labeled datasets such as ImageNet, as demonstrated in Fig. 4.3.

CLIP's popularity can also be attributed to a well-maintained and stable open source implementation `open_clip` by Schuhmann et al. [153], Cherti et al. [154], Ilharco et al. [155], allowing for controlled and reproducible experimentation. Fang et al. [171] conduct a systematic attribution analysis to isolate the causal factors which might explain the superior robustness of CLIP models against various distribution shifts, compared to other pre-training methodologies.

The investigation systematically assesses four key axes of the CLIP paradigm:

1. **Scale:** The sheer size of the pre-training dataset.
2. **Distribution:** The inherent diversity of the image/concept distribution within the training data.
3. **Supervision:** The role of natural language supervision during pre-training and/or zero-shot inference.
4. **Objective:** The use of the contrastive loss function.

The authors conclude that the diversity of the training data distribution is the principal contributor to CLIP's notable robustness gains, with the other factors providing only marginal or negligible increases in generalization capability. Despite this critical finding, the precise nature of the causal factor of "diversity" remains ambiguous: The authors do not provide a quantitative metric or a clear operational definition for dataset diversity, leaving the distinction between a "diverse" and a "less diverse" dataset unformalized.

Furthermore, a significant methodological limitation persists concerning potential data leakage. Given the large-scale, uncurated nature of CLIP's training data, there are no inherent controls to definitively preclude the presence of test samples or highly similar variants from established OOD benchmarks within the pre-training dataset. Consequently, the observed robustness may, at least in part, be an artifact of the extensive data distribution *de facto* containing instances of the OOD domains being tested.

To better understand the role of the "diverse" pretraining dataset, as well as to understand whether CLIP's strong robustness to distribution shifts can be explained by data leakage, Mayilvahanan et al. [172] focus on restoring a less contaminated evaluation scenario. Specifically, they aim to determine whether CLIP's strong generalization against distribution shifts can be explained by the leakage of test-domain samples into the pre-training dataset. To achieve this, Mayilvahanan et al. [172] measure the perceptual similarity between all test images in ImageNet-Sketch [51] and LAION-400M [173]. This analysis reveals that LAION-400M contains a considerable number of ImageNet-Sketch images, thereby making the evaluation task on ImageNet-Sketch to an in-distribution rather than to an out-of-distribution evaluation. However, removing the exact test images as well as other highly similar images from LAION's training data only marginally impacts CLIP's generalization capabilities, implying that test data leakage cannot fully explain CLIP's strong robustness. A key limitation of this work is the partial nature of the data decontamination. By restricting the removal process only to images which are perceptually close to the ImageNet-Sketch test set, the study failed to purge *all* samples belonging to the sketch domain from the training data, leaving many images from the ImageNet-Sketch domain in the dataset, allowing for distributional leakage.

To account for domain contamination, Mayilvahanan et al. [174] train very precise

domain classifiers in order to remove all sketch images from LAION-400M in a follow-up study. After training a model on a pruned version of LAION-400M which contains only natural images, it is observed that CLIP trained on a single domain does not generalize to other domains. In particular, training on a subset containing only natural images, results in poor performance on renditions. The performance on renditions can be restored if random rendition images are added to the training dataset.

Thus, domain contamination, rather than an intrinsic ability to generalize, indeed explains CLIP's strong performance on different domains. In a similar fashion, Hammoud et al. [175] study the downstream performance of models depending on the diversity of the pretraining data, and observe that a larger dataset only helps improve performance if the pretraining data distribution is close to the data distribution of the downstream task.

Today's foundation models are trained "on the internet" which makes it impossible to truly assess the make-up of their training distribution. Thus, the issue of (distributional) leakage of test data into the training data remains since we cannot be certain which parts of which OOD benchmarks the large-scale training datasets contain. Further, it becomes increasingly unclear what "OOD" actually means for these models since they have likely seen every possible data domain in some form during the pretraining phase. On the other hand, it is not clear whether this constitutes an actual problem in the first place: If the models are trained on large portions of everything that is available online and are therefore in-distribution with respect to virtually anything, then there might be no point in trying to assess their OOD capabilities. After all, if some generalization gaps still remain in the end, we can simply add more training data.

Take-away

Foundation models appear to be more robust to different distribution shifts compared to conventional models which might be explained by (distributional) leakage of test data into the training data.

4.5 How could future research on OOD generalization look like?

We live in exciting and turbulent times. From the perspective of a seasoned robustness researcher interested in benchmarking, the increasingly clear lack of separation between common OOD benchmarks and the training data is concerning and confusing. During the ImageNet-era, reporting performance numbers on OOD benchmarks was meaningful to track progress in OOD generalization. Nowadays, interpreting numbers reported on those benchmarks for various foundation models is increasingly challenging. Udandarao et al. [176] show that CLIP models perform better on classes they see more frequently during training. Mayilvahanan et al. [174] demonstrate that the number of rendition samples in the pretraining dataset directly affects the model's performance on renditions. In the language domain, we observe a comparable pattern: in arithmetic tasks, model performance on test cases correlates with how frequently the terms appearing in those cases were represented in the pretraining dataset [177]. The researchers emphasize in their paper's discussion that evaluating reasoning abilities requires accounting for

the composition of the pretraining corpus. They argue that claims about reasoning capabilities are only valid if the model demonstrates robustness against pretraining effects—that is, if performance holds up even when controlling for the influence of the training data.

A higher score on a popular out-of-distribution benchmark might simply reflect that the model's training data contained more examples from that particular distribution, rather than indicating genuine progress in OOD generalization. This raises a fundamental question: how can we meaningfully assess improvements in OOD generalization capabilities when we cannot easily separate model performance gains from the influence of training data composition?

Explicitly remove a particular test distribution from the training data. This method was used by Mayilvahanan et al. [174] as discussed in the last section: They measured performance on rendition OOD benchmarks after fully removing image renditions from the training data. To track algorithmic or architectural progress in foundation models regarding the model's OOD generalization capabilities, this type of interventional analysis on the data level could be performed for every newly released model. The downside of this approach is that it requires training a new model for every distribution shift of interest. Further, domain classifiers are not perfect and we cannot exclude the possibility that distributional leakage does occasionally happen. Additionally, the filtered training dataset is necessarily smaller than the original one. Wei et al. [178] report emergent capabilities in very large language models which do not happen at smaller scales. These emergent capabilities could be missed when training the models on smaller filtered datasets. Finally, the proposed solution is hard to scale: In order to make this process fair and reproducible, there would need to be a standardized code base which all labs who train foundation models would have to use. Many training datasets and some training code are proprietary and it would be difficult to control for compliance as well as exclude human error.

Dynamic benchmarking In contrast to the *static* OOD benchmarks used in the studies in this thesis, an increasingly popular idea has been to devise *dynamic* benchmarks which are updated constantly. Finding adversarial examples can be considered as a form of dynamic benchmarking, since the optimal attack is usually adapted to the particular defense. Typically, dynamic benchmarks involve a human-in-the-loop. For example, AdaVision [179] is an interactive framework for testing vision models in which users can identify and fix model errors. To test an object detection model, the user provides a prompt, e.g. "stop sign", and AdaVision retrieves relevant images from LAION-5B. These images are passed through the object detector and can be labeled by the user for correctness. If the detector fails on some particular distribution shift, e.g. when the stop sign is covered in snow, AdaVision searches for more images with this distribution shift, thereby identifying high-error regions, and the user manually labels those images. Finally, the detection model can be finetuned on the corrected images. Prabhu et al. [180] propose ever-expanding large-scale benchmarks to test vision models. To address the challenge of evaluating an increasing number of models on a continually expanding sample set, the researchers propose an efficient evaluation framework called Sort & Search (S&S). S&S leverages dynamic programming algorithms to selectively rank and sub-select test samples, allowing cost-effective lifelong benchmarking by reusing previously evaluated models.

Prabhu et al. [180] also include a comprehensive overview over both static and dynamic benchmarks in Appendix 10.

ELO-based performance tracking The adoption of the ELO rating system [181] for tracking the performance of Large Language Models (LLMs) and Vision-Language Models (VLMs) represents a crucial evolution from static, single-metric evaluation. While inherently related to the philosophical goals of dynamic benchmarking, the ELO approach distinctively focuses on granular, individual user judgment and large-scale crowd-sourcing of preference data, moving beyond curated, fixed test sets. This methodology, exemplified by platforms such as the Chatbot Arena [182], establishes a continuously updated, pairwise comparison of model outputs under unconstrained, real-world conditions. Crucially, the organic and unpredictable nature of these human queries naturally generates data points that lie outside the typical model training distributions, making the ELO framework exceptionally well-suited for evaluating true OOD performance. By utilizing comparative wins and losses to iteratively adjust model scores, the ELO ranking provides a robust, zero-sum metric for assessing model generalization, robustness, and overall utility in practical deployment scenarios.

Test for creativity and curiosity In their extensive report, Bubeck et al. [183] test GPT-4 on a variety of different tasks, spanning mathematics, coding, vision, medicine, law, psychology and more. They argue that testing GPT-4 on conventional benchmarks only tests for memorization but not generalization. Therefore, they propose "a different approach to studying GPT-4 which is closer to traditional psychology rather than machine learning, leveraging human creativity and curiosity". For example, they query GPT-4 with a prompt "Draw a unicorn in TikZ" and demonstrate that GPT-4 produces valid code which indeed clearly resembles a unicorn. Other notable examples are GPT-4's responses to the prompts "Write a proof of the fact that there are infinitely many primes; do it in the style of a Shakespeare play through a dialogue between two parties arguing over the proof" and "Write a supporting letter to Kasturba Gandhi for Electron, a subatomic particle as a US presidential candidate by Mahatma Gandhi". The proof that there are infinitely many primes uses appropriate language and vocabulary to match the style of a Shakespeare play. Further, the rhyme and meter are consistent and the dialogue sounds very poetic. The letter advocating for electron, the subatomic particle, contains elaborate word-plays which demonstrate GPT-4 ability for abstraction and even humor. For example, the sentence "He is a peacemaker, who can balance forces, resolve conflicts, and harmonize systems" can be interpreted from a physical or a metaphorical viewpoint. In summary, Bubeck et al. [183] invent new tasks which are not part of GPT-4's training data.

Another prominent example of testing for creativity is the canonical astronaut riding on a horse on the moon, generated by Stable Diffusion (SD; 184), see Fig. 4.4. Clearly, this particular generated image has become popular due to its impossibility as it demonstrates SD's generalization capabilities. Overall, these kinds of analyses are interesting and certainly entertaining, but they also lack scientific rigor and allow for cherry-picking.

Figure 4.4: Demonstration of the capabilities of Stable Diffusion: Stable Diffusion can generate images which go beyond memorization. *Figure adopted from Wikipedia: Stable Diffusion [185].*

Take-away

Future research on OOD generalization will likely involve dynamic and ever-expanding benchmarks to counteract (distributional) leakage of static benchmarks into the training data, as well as crowd-sourced ELO based performance tracking.

5 Conclusion

In this dissertation, several methods for improving the generalization capabilities of deep neural networks beyond their training distribution were presented.

On the one hand, different kinds of data augmentation proved promising. In the first approach, noise patterns that were maximally confusing to a classifier were generated and then used as data augmentation to train the classifier against those noise patterns. Training the classifier and the noise generation network jointly improved the classifier's robustness to different kinds of image corruptions. In a different approach, it was demonstrated that object detection models are also vulnerable to image corruptions, but their resilience was shown to be improved with data augmentation in the form of stylization.

Next, a different learning paradigm was discussed: Self-supervised learning, which is a powerful way to train neural networks as it does not require labels. One popular self-supervised learning method is contrastive learning, where a model learns a low-dimensional representation of data by gathering similar samples close together and pushing dissimilar ones apart in the representation space. The theoretical foundations and understanding of contrastive learning and its connections to identifiability theory were extended and verified empirically.

When the distribution shift persisted over several samples, the model was allowed to adapt to the shift, focusing on the area of continual and life-long learning. It was shown that a very simple, gradient-free technique, namely adapting the batch normalization statistics to the shifted distribution, could strongly improve the classifier's robustness to image corruptions. The caveat here was that this technique could only rectify low-level shifts, such as additive noises, brightness, or contrast changes. However, high-level shifts, such as changes in the viewing perspective based on the camera position or strong distributional changes like shifting from natural images to renditions or comics, could not be fixed with this simple technique.

In a follow-up study, a more powerful technique of self-learning was then explored, where pseudo-labels were produced by the model itself and were used to finetune the model. It was observed that different variants of self-learning worked very well to adapt various models to different distribution shifts.

Throughout this dissertation, a data-centric view on model generalization was adopted: The distribution of the training data determined how well the trained model generalized to other distributions. Allowing the model to adapt to a shifted distribution was posited as a means of better aligning the model with the test distribution. While data augmentation did not necessarily bring the training data distribution closer to the test distribution, it provided a form of regularization by forcing the model to learn more general features which could be helpful for different shifts.

This data-centric view on generalization was taken a step further by abstracting away the model architecture and the training paradigm to focus solely on the training data. In Chapter 4.4, two studies which were not part of this dissertation were discussed where the generalization capabilities of CLIP models were analyzed. CLIP models differ from the ImageNet-trained ResNets used in most of the studies discussed in this thesis in various ways, but the most notable distinction is the difference in the training data distribution: CLIP models are trained on web-scale data of image-caption pairs and have become very popular due to their strong results on various Out-of-Distribution (OOD) benchmarks. It was shown that CLIP's impressive generalization capabilities could be explained by distributional leakage of test data into the training dataset. For example, once all sketches were removed from CLIP's training data, the trained model performed much worse on the ImageNet-Sketch benchmark. In summary, it is emphasized that the use of large-scale data masks the challenge of OOD generalization, which remains an unsolved issue.

Understanding, controlling, and improving the generalization capabilities of deep neural networks is a fundamental prerequisite for the large-scale deployment of AI-based systems in real-world applications. It is necessary to understand which distribution shifts models are vulnerable to and how a model's performance can be improved. This dissertation aims to offer a toolkit of potential solutions that users may find helpful.

References

[1] OpenAI. ChatGPT: A Large-Scale Generative Pre-trained Transformer. `https://openai.com/chatgpt`, 2022. Accessed: February 28, 2024.

[2] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[4] Paul Resnick and Hal R. Varian. Recommender systems. *Communications of the ACM*, 40(3):56–58, 1997.

[5] Tesla. A tragic loss. electric cars, solar & clean energy. retrieved december 2, 2023. `https://www.tesla.com/blog/tragic-loss`, 2016.

[6] John R. Zech, Marcus A. Badgeley, Manway Liu, Anthony B. Costa, Joseph J. Titano, and Eric Karl Oermann. Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs: A cross-sectional study. *PLOS Medicine*, 15:1–17, 11 2018. doi: 10.1371/journal.pmed.1002683. URL `https://doi.org/10.1371/journal.pmed.1002683`.

[7] Jiashuo Liu, Zheyan Shen, Yue He, Xingxuan Zhang, Renzhe Xu, Han Yu, and Peng Cui. Towards out-of-distribution generalization: A survey. *arXiv preprint arXiv:2108.13624*, 2021.

[8] Vladimir Vapnik. Principles of risk minimization for learning theory. *Advances in neural information processing systems*, 4, 1991.

[9] Masashi Sugiyama and Motoaki Kawanabe. *Machine learning in non-stationary environments: Introduction to covariate shift adaptation*. MIT press, 2012.

[10] Bernhard Schölkopf, Dominik Janzing, Jonas Peters, Eleni Sgouritsa, Kun Zhang, and Joris Mooij. On causal and anticausal learning. *arXiv preprint arXiv:1206.6471*, 2012.

[11] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255, 2009.

[12] Live Science. Octopus facts. `https://www.livescience.com/55478-octopus-facts.html`, 2024.

[13] Jérôme Dockès, Gaël Varoquaux, and Jean-Baptiste Poline. Preventing dataset shift from breaking machine-learning biomarkers. *GigaScience*, 10(9):giab055, 2021.

[14] Reaz Mahmud, Md Mujibur Rahman, Iftikher Alam, Kazi Gias Uddin Ahmed, AKM Humayon Kabir, SK Jakaria Been Sayeed, Mohammad Aftab Rassel, Farhana Binte Monayem, Md Shahidul Islam, Mohammad Monirul Islam, et al. Ivermectin in combination with doxycycline for treating covid-19 symptoms: a randomized trial. *Journal of International Medical Research*, 49(5): 03000605211013550, 2021.

[15] Susanna Naggie, David R Boulware, Christopher J Lindsell, Thomas G Stewart, Nina Gentile, Sean Collins, Matthew William McCarthy, Dushyantha Jayaweera, Mario Castro, Mark Sulkowski, et al. Effect of ivermectin vs placebo on time to sustained recovery in outpatients with mild to moderate covid-19: a randomized clinical trial. *Jama*, 328(16):1595–1603, 2022.

[16] Scott Alexander. Ivermectin: Much more than you wanted to know. `https://www.astralcodexten.com/p/ivermectin-much-more-than-you-wanted`, 2021.

[17] Kinga Głuchowska, Tomasz Dzieciątkowski, Aleksandra Sędzikowska, Anna Zawistowska-Deniziak, and Daniel Młocicki. The new status of parasitic diseases in the covid-19 pandemic—risk factors or protective agents? *Journal of Clinical Medicine*, 10(11):2533, 2021.

[18] Joaquin Quionero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D. Lawrence. *Dataset Shift in Machine Learning*. The MIT Press, 2009. ISBN 0262170051.

[19] Karin Stacke, Gabriel Eilertsen, Jonas Unger, and Claes F. Lundström. A closer look at domain shift for deep learning in histopathology. *ArXiv*, abs/1909.11575, 2019. URL `https://api.semanticscholar.org/CorpusID:202750093`.

[20] Wenshuai Zhao, Jorge Peña Queralta, and Tomi Westerlund. Sim-to-real transfer in deep reinforcement learning for robotics: a survey. In *2020 IEEE symposium series on computational intelligence (SSCI)*, pages 737–744. IEEE, 2020.

[21] Sebastian Huch, Luca Scalerandi, Esteban Rivera, and Markus Lienkamp. Quantifying the lidar sim-to-real domain shift: A detailed investigation using object detectors and analyzing point clouds at target-level. *IEEE Transactions on Intelligent Vehicles*, 2023.

[22] Emanuele Alberti, Antonio Tavera, Carlo Masone, and Barbara Caputo. Idda: A large-scale multi-domain dataset for autonomous driving. *IEEE Robotics and Automation Letters*, 5(4):5526–5533, 2020.

[23] Chen Chen, Qifeng Chen, Jia Xu, and Vladlen Koltun. Learning to see in the dark. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3291–3300, 2018.

[24] Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. A comprehensive survey of continual learning: Theory, method and application. *arXiv preprint arXiv:2302.00487*, 2023.

[25] Yuxiao Zhang, Alexander Carballo, Hanting Yang, and Kazuya Takeda. Perception and sensing for autonomous vehicles under adverse weather conditions: A survey. *ISPRS Journal of Photogrammetry and Remote Sensing*, 196:146–177, 2023.

[26] Georg Volk, Stefan Müller, Alexander von Bernuth, Dennis Hospach, and Oliver Bringmann. Towards robust cnn-based object detection through augmentation with synthetic rain variations. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 285–292, 2019.

[27] Alexander von Bernuth, Georg Volk, and Oliver Bringmann. Simulating photorealistic snow and fog on existing images for enhanced cnn training and evaluation. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 41–46, 2019. doi: 10.1109/ITSC.2019.8917367.

[28] Birgid Schömig-Markiefka, Alexey Pryalukhin, Wolfgang Hulla, Andrey Bychkov, Junya Fukuoka, Anant Madabhushi, Viktor Achter, Lech Nieroda, Reinhard Büttner, Alexander Quaas, et al. Quality control stress test for deep learning-based diagnostic model in digital pathology. *Modern Pathology*, 34(12):2098–2108, 2021.

[29] Theresa Roland, Carl Böck, Thomas Tschoellitsch, Alexander Maletzky, Sepp Hochreiter, Jens Meier, and Günter Klambauer. Domain shifts in machine learning based covid-19 diagnosis from blood tests. *Journal of Medical Systems*, 46(5):23, 2022.

[30] Stephen R Benoit, Israel Hora, Ann L Albright, and Edward W Gregg. New directions in incidence and prevalence of diagnosed diabetes in the usa. *BMJ Open Diabetes Research and Care*, 7(1):e000657, 2019.

[31] Alan C Geller, Richard W Clapp, Arthur J Sober, Lou Gonsalves, Lloyd Mueller, Cindy L Christiansen, Waqas Shaikh, and Donald R Miller. Melanoma epidemic: an analysis of six decades of data from the connecticut tumor registry. *Journal of clinical oncology*, 31(33):4172, 2013.

[32] Justin M. Johnson and Taghi M. Khoshgoftaar. Survey on deep learning with class imbalance. *Journal of Big Data*, 6:1–54, 2019. URL `https://api.semanticscholar.org/CorpusID:102354936`.

[33] Agostina J Larrazabal, Nicolás Nieto, Victoria Peterson, Diego H Milone, and Enzo Ferrante. Gender imbalance in medical imaging datasets produces biased classifiers for computer-aided diagnosis. *Proceedings of the National Academy of Sciences*, 117(23):12592–12594, 2020.

[34] Wei Wei, Jinjiu Li, Longbing Cao, Yuming Ou, and Jiahang Chen. Effective detection of sophisticated online banking fraud on extremely imbalanced data. *World Wide Web*, 16:449–475, 2013.

[35] Matthew Herland, Taghi M Khoshgoftaar, and Richard A Bauder. Big data fraud detection using multiple medicare data sources. *Journal of Big Data*, 5(1): 1–21, 2018.

[36] Eduard Martynov and Yuuichiroh Uematsu. Dealing with class imbalance in bird sound classification. *CLEF Working Notes*, 2022.

[37] Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion attacks against machine learning at test time. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2013, Prague, Czech Republic, September 23-27, 2013, Proceedings, Part III 13*, pages 387–402. Springer, 2013.

[38] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[39] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *CoRR*, abs/1412.6572, 2014. URL `https://api.semanticscholar.org/CorpusID:6706414`.

[40] Alexey Kurakin, Ian J Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *Artificial intelligence safety and security*, pages 99–112. Chapman and Hall/CRC, 2018.

[41] Wieland Brendel, Jonas Rauber, and Matthias Bethge. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. *arXiv preprint arXiv:1712.04248*, 2017.

[42] Cihang Xie, Jianyu Wang, Zhishuai Zhang, Yuyin Zhou, Lingxi Xie, and Alan Loddon Yuille. Adversarial examples for semantic segmentation and object detection. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 1378–1387, 2017.

[43] Anurag Arnab, Ondrej Miksik, and Philip H. S. Torr. On the robustness of semantic segmentation models to adversarial attacks. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 888–897, 2017.

[44] Chuanxing Geng, Sheng-Jun Huang, and Songcan Chen. Recent advances in open set recognition: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(10):3614–3631, 2021. doi: 10.1109/TPAMI.2020.2981604.

[45] The Merriam-Webster Dictionary. Definition of the word 'benchmark'. `https://www.merriam-webster.com/dictionary/benchmark`, 2024.

[46] Dan Hendrycks and Thomas G. Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *International Conference on Learning Representations (ICLR)*, 2019. URL `https://openreview.net/forum?id=HJz6tiCqYm`.

[47] Claudio Michaelis, Benjamin Mitzkus, Robert Geirhos, Evgenia Rusak, Oliver Bringmann, Alexander S Ecker, Matthias Bethge, and Wieland Brendel. Benchmarking robustness in object detection: Autonomous driving when winter is coming. In *Machine Learning for Autonomous Driving Workshop, NeurIPS*, 2019.

[48] Norman Mu and Justin Gilmer. MNIST-C: A robustness benchmark for computer vision. *arxiv*, 2019.

[49] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, Dawn Song, Jacob Steinhardt, and Justin Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 8340–8349, October 2021.

[50] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15262–15271, 2021.

[51] Haohan Wang, Songwei Ge, Zachary Lipton, and Eric P Xing. Learning robust global representations by penalizing local predictive power. In *Advances in Neural Information Processing Systems*, pages 10506–10518, 2019.

[52] Andrei Barbu, David Mayo, Julian Alverio, William Luo, Christopher Wang, Dan Gutfreund, Josh Tenenbaum, and Boris Katz. Objectnet: A large-scale bias-controlled dataset for pushing the limits of object recognition models. *Advances in neural information processing systems*, 32, 2019.

[53] Amanda's Cookin'. English trifle: Our family tradition. `https://amandascookin.com/traditional-english-trifle/`, 2024.

[54] Britannica. conch. `https://www.britannica.com/animal/conch`, 2024.

[55] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, 2021. URL `https://api.semanticscholar.org/CorpusID:231591445`.

[56] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *International conference on machine learning*, pages 4904–4916. PMLR, 2021.

[57] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*, pages 647–655, 2014.

[58] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian conference on computer vision, graphics & image processing*, pages 722–729. IEEE, 2008.

[59] Peter Welinder, Steve Branson, Takeshi Mita, Catherine Wah, Florian Schroff, Serge Belongie, and Pietro Perona. Caltech-UCSD Birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology, 2010.

[60] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Introducing eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. In *IGARSS 2018-2018 IEEE International Geoscience and Remote Sensing Symposium*, pages 204–207. IEEE, 2018.

[61] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, , and A. Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[62] Xiaohua Zhai, Joan Puigcerver, Alexander Kolesnikov, Pierre Ruyssen, Carlos Riquelme, Mario Lucic, Josip Djolonga, André Susano Pinto, Maxim Neumann, Alexey Dosovitskiy, Lucas Beyer, Olivier Bachem, Michael Tschannen, Marcin Michalski, Olivier Bousquet, Sylvain Gelly, and Neil Houlsby. A large-scale study of representation learning with the visual task adaptation benchmark. *arXiv: Computer Vision and Pattern Recognition*, 2019. URL `https://api.semanticscholar.org/CorpusID:214317405`.

[63] Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, et al. Wilds: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning*, pages 5637–5664. PMLR, 2021.

[64] Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. *International Conference on Learning Representations (ICLR)*, 2021.

[65] Ishaan Gulrajani, David Lopez-Paz. Domainbed. `https://github.com/facebookresearch/DomainBed`, 2020.

[66] Jonas Rauber, Wieland Brendel, and Matthias Bethge. Foolbox: A python toolbox to benchmark the robustness of machine learning models. In *Reliable Machine Learning in the Wild Workshop, 34th International Conference on Machine Learning*, 2017. URL `http://arxiv.org/abs/1707.04131`.

[67] Jonas Rauber, Roland Zimmermann, Matthias Bethge, and Wieland Brendel. Foolbox native: Fast adversarial attacks to benchmark the robustness of machine learning models in pytorch, tensorflow, and jax. *Journal of Open Source Software*, 5(53):2607, 2020. doi: 10.21105/joss.02607. URL `https://doi.org/10.21105/joss.02607`.

[68] Nicolas Papernot, Fartash Faghri, Nicholas Carlini, Ian Goodfellow, Reuben Feinman, Alexey Kurakin, Cihang Xie, Yash Sharma, Tom Brown, Aurko Roy, Alexander Matyasko, Vahid Behzadan, Karen Hambardzumyan, Zhishuai Zhang, Yi-Lin Juang, Zhi Li, Ryan Sheatsley, Abhibhav Garg, Jonathan Uesato, Willi Gierke, Yinpeng Dong, David Berthelot, Paul Hendricks, Jonas Rauber, and Rujun Long. Technical report on the cleverhans v2.1.0 adversarial examples library. *arXiv preprint arXiv:1610.00768*, 2018.

[69] Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *ICML*, 2020.

[70] Francesco Croce and Matthias Hein. Mind the box: l_1-apgd for sparse adversarial attacks on image classifiers. In *ICML*, 2021.

[71] Francesco Croce, Maksym Andriushchenko, Vikash Sehwag, Edoardo Debenedetti, Nicolas Flammarion, Mung Chiang, Prateek Mittal, and Matthias Hein. Robustbench: a standardized adversarial robustness benchmark. *arXiv preprint arXiv:2010.09670*, 2020.

[72] David H Wolpert and William G Macready. No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, 1(1):67–82, 1997.

[73] David H Wolpert. The lack of a priori distinctions between learning algorithms. *Neural computation*, 8(7):1341–1390, 1996.

[74] Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12104–12113, 2022.

[75] Joel Hestness, Sharan Narang, Newsha Ardalani, Gregory Diamos, Heewoo Jun, Hassan Kianinejad, Md Mostofa Ali Patwary, Yang Yang, and Yanqi Zhou. Deep learning scaling is predictable, empirically. *arXiv preprint arXiv:1712.00409*, 2017.

[76] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

[77] Tom Henighan, Jared Kaplan, Mor Katz, Mark Chen, Christopher Hesse, Jacob Jackson, Heewoo Jun, Tom B Brown, Prafulla Dhariwal, Scott Gray, et al. Scaling laws for autoregressive generative modeling. *arXiv preprint arXiv:2010.14701*, 2020.

[78] Jonathan S Rosenfeld, Amir Rosenfeld, Yonatan Belinkov, and Nir Shavit. A constructive prediction of the generalization error across scales. *arXiv preprint arXiv:1909.12673*, 2019.

[79] Mitchell A Gordon, Kevin Duh, and Jared Kaplan. Data and parameter scaling laws for neural machine translation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5915–5922, 2021.

[80] Danny Hernandez, Jared Kaplan, Tom Henighan, and Sam McCandlish. Scaling laws for transfer. *arXiv preprint arXiv:2102.01293*, 2021.

[81] Rohan Taori, Achal Dave, Vaishaal Shankar, Nicholas Carlini, Benjamin Recht, and Ludwig Schmidt. Measuring robustness to natural distribution shifts in image classification. *Advances in Neural Information Processing Systems*, 33: 18583–18599, 2020.

[82] Robert Geirhos, Kantharaju Narayanappa, Benjamin Mitzkus, Tizian Thieringer, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Partial success in closing the gap between human and machine vision. In *Advances in Neural Information Processing Systems 34*, 2021.

[83] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[84] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=Bygh9j09KX`.

[85] Philip TG Jackson, Amir Atapour Abarghouei, Stephen Bonner, Toby P Breckon, and Boguslaw Obara. Style augmentation: data augmentation via style randomization. In *CVPR workshops*, volume 6, pages 10–11, 2019.

[86] Chaithanya Kumar Mummadi, Ranjitha Subramaniam, Robin Hutmacher, Julien Vitay, Volker Fischer, and Jan Hendrik Metzen. Does enhanced shape bias improve neural network robustness to common corruptions? In *International Conference on Learning Representations*, 2021.

[87] Dan Hendrycks, Norman Mu, Ekin D. Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. AugMix: A simple data processing method to improve robustness and uncertainty. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020.

[88] Eric Mintun, Alexander Kirillov, and Saining Xie. On interaction between augmentations and corruptions in natural corruption robustness. *Advances in Neural Information Processing Systems*, 34:3571–3583, 2021.

[89] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations (ICLR)*, 2018.

[90] Suorong Yang, Weikang Xiao, Mengcheng Zhang, Suhan Guo, Jian Zhao, and Furao Shen. Image data augmentation for deep learning: A survey. *arXiv preprint arXiv:2204.08610*, 2022.

[91] Connor Shorten and Taghi M. Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6, 2019.

[92] Mingle Xu, Sook Yoon, Alvaro Fuentes, and Dong Sun Park. A comprehensive survey of image augmentation techniques for deep learning. *Pattern Recognition*, 137, 2023.

[93] Ilyes Khemakhem, Diederik P. Kingma, Ricardo Monti, and Aapo Hyvarinen. Variational autoencoders and nonlinear ica: A unifying framework. In *International Conference on Learning Representations (ICLR)*, 2020.

[94] Aapo Hyvarinen and Hiroshi Morioka. Unsupervised feature extraction by time-contrastive learning and nonlinear ica. *Advances in neural information processing systems*, 29, 2016.

[95] Aapo Hyvarinen, Juha Karhunen, and Erkki Oja. Independent component analysis. *John Wiley & Sons*, 2001.

[96] Christian Jutten and Juha Karhunen. Advances in blind source separation (bss) and independent component analysis (ica) for nonlinear mixtures. *International journal of neural systems*, 14(05):267–292, 2004.

[97] Sara Beery, Grant Van Horn, and Pietro Perona. Recognition in terra incognita. In *Proceedings of the European conference on computer vision (ECCV)*, pages 456–473, 2018.

[98] Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Raetsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. In *international conference on machine learning*, pages 4114–4124. PMLR, 2019.

[99] Roland S. Zimmermann, Yash Sharma, Steffen Schneider, Matthias Bethge, and Wieland Brendel. Contrastive learning inverts the data generating process. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 12979–12990. PMLR, 2021. URL `http://proceedings.mlr.press/v139/zimmermann21a.html`.

[100] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario March, and Victor Lempitsky. Domain-adversarial training of neural networks. *Journal of machine learning research*, 17(59):1–35, 2016.

[101] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*, pages 1180–1189. PMLR, 2015.

[102] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. Learning multiple visual domains with residual adapters. *Advances in neural information processing systems*, 30, 2017.

[103] Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7167–7176, 2017.

[104] Mei Wang and Weihong Deng. Deep visual domain adaptation: A survey. *Neurocomputing*, 312:135–153, 2018.

[105] Hao Guan and Mingxia Liu. Domain adaptation for medical image analysis: A survey. *IEEE Transactions on Biomedical Engineering*, 69(3), 2022.

[106] Bernhard Schölkopf. Causality for machine learning. In *Probabilistic and Causal Inference: The Works of Judea Pearl*, pages 765–804. 2022.

[107] Judea Pearl. *Causality*. Cambridge university press, 2009.

[108] Peter Spirtes, Clark Glymour, and Richard Scheines. *Causation, prediction, and search*. MIT press, 2001.

[109] Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019.

[110] Kartik Ahuja, Karthikeyan Shanmugam, Kush Varshney, and Amit Dhurandhar. Invariant risk minimization games. In *International Conference on Machine Learning*, pages 145–155. PMLR, 2020.

[111] Elliot Creager, Jörn-Henrik Jacobsen, and Richard Zemel. Environment inference for invariant learning. In *International Conference on Machine Learning*, pages 2189–2200. PMLR, 2021.

[112] Divyat Mahajan, Shruti Tople, and Amit Sharma. Domain generalization using causal matching. In *International Conference on Machine Learning*, pages 7313–7324. PMLR, 2021.

[113] Jianqing Fan, Cong Fang, Yihong Gu, and Tong Zhang. Environment invariant linear least squares. *arXiv preprint arXiv:2303.03092*, 2023.

[114] Bo Li, Yifei Shen, Yezhen Wang, Wenzhen Zhu, Dongsheng Li, Kurt Keutzer, and Han Zhao. Invariant information bottleneck for domain generalization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022.

[115] Dong Yin, Raphael Gontijo Lopes, Jon Shlens, Ekin Dogus Cubuk, and Justin Gilmer. A fourier perspective on model robustness in computer vision. *Advances in Neural Information Processing Systems*, 32, 2019.

[116] Evgenia Rusak, Lukas Schott, Roland S Zimmermann, Julian Bitterwolf, Oliver Bringmann, Matthias Bethge, and Wieland Brendel. A simple way to make neural networks robust against diverse image corruptions. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, pages 53–69. Springer, 2020.

[117] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2414–2423, 2016.

[118] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1765–1773, 2017.

[119] Dan A Calian, Florian Stimberg, Olivia Wiles, Sylvestre-Alvise Rebuffi, Andras Gyorgy, Timothy Mann, and Sven Gowal. Defending against image corruptions through adversarial augmentations. *arXiv preprint arXiv:2104.01086*, 2021.

[120] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 136–144, 2017.

[121] Lucas Theis, Wenzhe Shi, Andrew Cunningham, and Ferenc Huszár. Lossy image compression with compressive autoencoders. In *International conference on learning representations*, 2017.

[122] Oğuzhan Fatih Kar, Teresa Yeo, Andrei Atanov, and Amir Zamir. 3d common corruptions and data augmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18963–18974, 2022.

[123] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE international conference on computer vision*, pages 1501–1510, 2017.

[124] Kaggle. Painter by numbers dataset. `https://www.kaggle.com/c/painter-by-numbers/`, 2016.

[125] Fisher Yu, Wenqi Xian, Yingying Chen, Fangchen Liu, Mike Liao, Vashisht Madhavan, Trevor Darrell, et al. Bdd100k: A diverse driving video database with scalable annotation tooling. *arXiv preprint arXiv:1805.04687*, 2(5):6, 2018.

[126] Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Semantic foggy scene understanding with synthetic data. *International Journal of Computer Vision*, 126:973–992, 2018.

[127] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[128] Evgenia Rusak, Patrik Reizinger, Attila Juhos, Oliver Bringmann, Roland S. Zimmermann, and Wieland Brendel. Infonce: Identifying the gap between theory and practice. In *AISTATS*, 2025.

[129] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A Simple Framework for Contrastive Learning of Visual Representations. In *ICML*. PMLR, 2020.

[130] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation Learning with Contrastive Predictive Coding. *arXiv preprint*, 2018.

[131] Julius von Kügelgen, Yash Sharma, Luigi Gresele, Wieland Brendel, Bernhard Schölkopf, Michel Besserve, and Francesco Locatello. Self-supervised learning with data augmentations provably isolates content from style. In *NeurIPS*, 2021.

[132] Li Jing, Pascal Vincent, Yann LeCun, and Yuandong Tian. Understanding dimensional collapse in contrastive self-supervised learning. In *ICLR*. OpenReview.net, 2022.

[133] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. pmlr, 2015.

[134] Steffen Schneider, Evgenia Rusak, Luisa Eck, Oliver Bringmann, Wieland Brendel, and Matthias Bethge. Improving robustness against common corruptions by covariate shift adaptation. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, 2020.

[135] Evgenia Rusak, Steffen Schneider, George Pachitariu, Luisa Eck, Peter Vincent Gehler, Oliver Bringmann, Wieland Brendel, and Matthias Bethge. If your data distribution shifts, use self-learning. *Transactions on Machine Learning Research*, 2022. ISSN 2835-8856. URL `https://openreview.net/forum?id=vqRzLv6POg`. Expert Certification.

[136] Jogendra Nath Kundu, Naveen Venkat, R Venkatesh Babu, et al. Universal source-free domain adaptation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4544–4553, 2020.

[137] Ya Tsypkin. Self-learning–what is it? *IEEE Transactions on Automatic Control*, 13(6):608–612, 1968.

[138] Dong-Hyun Lee et al. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, page 896. Atlanta, 2013.

[139] Aram Galstyan and Paul R Cohen. Empirical comparison of "hard" and "soft" label propagation for relational classification. In *International Conference on Inductive Logic Programming*, pages 98–111. Springer, 2007.

[140] Zhilu Zhang and Mert Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. *Advances in neural information processing systems*, 31, 2018.

[141] Yves Grandvalet and Yoshua Bengio. Semi-supervised learning by entropy minimization. *Advances in neural information processing systems*, 17, 2004.

[142] Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=uXl3bZLkr3c`.

[143] Dian Chen, Dequan Wang, Trevor Darrell, and Sayna Ebrahimi. Contrastive test-time adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 295–305, 2022.

[144] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1406–1415, 2019.

[145] Ori Press, Steffen Schneider, Matthias Kümmerer, and Matthias Bethge. Rdumb: A simple approach that questions our progress in continual test-time adaptation. In *Advances in Neural Information Processing Systems*, 2023.

[146] Chaithanya Kumar Mummadi, Robin Hutmacher, Kilian Rambach, Evgeny Levinkov, Thomas Brox, and Jan Hendrik Metzen. Test-time adaptation to distribution shift by confidence maximization and input transformation. *arXiv preprint arXiv:2106.14999*, 2021.

[147] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

[148] Future of Life Institute. Pause giant ai experiments: An open letter. `https://futureoflife.org/open-letter/pause-giant-ai-experiments/`, 2023.

[149] Center for AI Safety. Statement on ai risk. `https://www.safe.ai/work/statement-on-ai-risk`, 2023.

[150] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. In *International Conference on Learning Representations (ICLR)*, 2018.

[151] Robert Geirhos, Kristof Meding, and Felix A Wichmann. Beyond accuracy: quantifying trial-by-trial behaviour of cnns and humans by measuring error consistency. *Advances in Neural Information Processing Systems*, 33:13890–13902, 2020.

[152] Max Wolff, Thomas Klein, Evgenia Rusak, Felix Wichmann, and Wieland Brendel. Low-pass filtering improves behavioral alignment of vision models. *submitted to ICLR*, 2026.

[153] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade W Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, Patrick Schramowski, Srivatsa R Kundurthy, Katherine Crowson, Ludwig Schmidt, Robert Kaczmarczyk, and Jenia Jitsev. LAION-5b: An open large-scale dataset for training next generation image-text models. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022. URL `https://openreview.net/forum?id=M3Y74vmsMcY`.

[154] Mehdi Cherti, Romain Beaumont, Ross Wightman, Mitchell Wortsman, Gabriel Ilharco, Cade Gordon, Christoph Schuhmann, Ludwig Schmidt, and Jenia Jitsev. Reproducible scaling laws for contrastive language-image learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2818–2829, 2023.

[155] Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. Openclip, July 2021. URL `https://doi.org/10.5281/zenodo.5143773`. If you use this software, please cite it as below.

[156] FW Campbell and DG Green. Optical and retinal factors affecting visual resolution. *The Journal of physiology*, 181(3):576, 1965.

[157] David R Williams, David H Brainard, Matthew J McMahon, and Rafael Navarro. Double-pass and interferometric measures of the optical quality of the eye. *Journal of the Optical Society of America A*, 11(12):3123–3135, 1994.

[158] Fergus W Campbell and John G Robson. Application of fourier analysis to the visibility of gratings. *The Journal of physiology*, 197(3):551, 1968.

[159] Donald H Kelly. Motion and vision. ii. stabilized spatio-temporal threshold surface. *Journal of the optical society of America*, 69(10):1340–1349, 1979.

[160] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020.

[161] Mert Yuksekgonul, Federico Bianchi, Pratyusha Kalluri, Dan Jurafsky, and James Zou. When and why vision-language models behave like bags-of-words, and what to do about it? In *International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=KRLUvxh8uaX`.

[162] John P Miller, Rohan Taori, Aditi Raghunathan, Shiori Sagawa, Pang Wei Koh, Vaishaal Shankar, Percy Liang, Yair Carmon, and Ludwig Schmidt. Accuracy on the line: on the strong correlation between out-of-distribution and in-distribution generalization. In *International conference on machine learning*, pages 7721–7735. PMLR, 2021.

[163] Florian Wenzel, Andrea Dittadi, Peter Gehler, Carl-Johann Simon-Gabriel, Max Horn, Dominik Zietlow, David Kernert, Chris Russell, Thomas Brox, Bernt Schiele, et al. Assaying out-of-distribution generalization in transfer learning. *Advances in Neural Information Processing Systems*, 35:7181–7198, 2022.

[164] Simon Kornblith, Jonathon Shlens, and Quoc V Le. Do better imagenet models transfer better? In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2661–2671, 2019.

[165] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*, pages 491–507. Springer, 2020.

[166] Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens Van Der Maaten. Exploring the limits of weakly supervised pretraining. In *Proceedings of the European conference on computer vision (ECCV)*, pages 181–196, 2018.

[167] Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. Self-training with noisy student improves imagenet classification. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10687–10698, 2020.

[168] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the IEEE international conference on computer vision*, pages 843–852, 2017.

[169] Damien Teney, Yong Lin, Seong Joon Oh, and Ehsan Abbasnejad. Id and ood performance are sometimes inversely correlated on real-world datasets. *Advances in Neural Information Processing Systems*, 36, 2023.

[170] Alex Fang, Simon Kornblith, and Ludwig Schmidt. Does progress on imagenet transfer to real-world datasets? *Advances in Neural Information Processing Systems*, 36, 2023.

[171] Alex Fang, Gabriel Ilharco, Mitchell Wortsman, Yuhao Wan, Vaishaal Shankar, Achal Dave, and Ludwig Schmidt. Data determines distributional robustness in contrastive language image pre-training (clip). In *International Conference on Machine Learning*, pages 6216–6234. PMLR, 2022.

[172] Prasanna Mayilvahanan, Thaddäus Wiedemer, Evgenia Rusak, Matthias Bethge, and Wieland Brendel. Does clip's generalization performance mainly stem from high train-test similarity? In *International Conference on Learning Representations (ICLR)*, 2024.

[173] Christoph Schuhmann, Richard Vencu, Romain Beaumont, Robert Kaczmarczyk, Clayton Mullis, Aarush Katta, Theo Coombes, Jenia Jitsev, and Aran

Komatsuzaki. Laion-400m: Open dataset of clip-filtered 400 million image-text pairs. *arXiv preprint arXiv:2111.02114*, 2021.

[174] Prasanna Mayilvahanan, Roland S. Zimmermann, Thaddäus Wiedemer, Evgenia Rusak, Attila Juhos, Matthias Bethge, and Wieland Brendel. In search of forgotten domain generalization. In *International Conference on Learning Representations (ICLR)*, 2025.

[175] Hasan Abed Al Kader Hammoud, Tuhin Das, Fabio Pizzati, Philip Torr, Adel Bibi, and Bernard Ghanem. On pretraining data diversity for self-supervised learning. *arXiv preprint arXiv:2403.13808*, 2024.

[176] Vishaal Udandarao, Ameya Prabhu, Adhiraj Ghosh, Yash Sharma, Philip HS Torr, Adel Bibi, Samuel Albanie, and Matthias Bethge. No" zero-shot" without exponential data: Pretraining concept frequency determines multimodal model performance. *arXiv preprint arXiv:2404.04125*, 2024.

[177] Yasaman Razeghi, Robert L Logan IV, Matt Gardner, and Sameer Singh. Impact of pretraining term frequencies on few-shot reasoning. *arXiv preprint arXiv:2202.07206*, 2022.

[178] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022.

[179] Irena Gao, Gabriel Ilharco, Scott Lundberg, and Marco Tulio Ribeiro. Adaptive testing of computer vision models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4003–4014, 2023.

[180] Ameya Prabhu, Vishaal Udandarao, Philip Torr, Matthias Bethge, Adel Bibi, and Samuel Albanie. Lifelong benchmarks: Efficient model evaluation in an era of rapid progress. *arXiv preprint arXiv:2402.19472*, 2024.

[181] Arpad E. Elo. *The Rating of Chess Players, Past and Present*. Arco Pub., New York, 1978. ISBN 0668047216 9780668047210.

[182] Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Banghua Zhu, Hao Zhang, Michael Jordan, Joseph E Gonzalez, et al. Chatbot arena: An open platform for evaluating llms by human preference. In *Forty-first International Conference on Machine Learning*, 2024.

[183] Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*, 2023.

[184] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[185] Wikipedia: Stable Diffusion. Stable diffusion. `https://en.wikipedia.org/wiki/Stable_Diffusion`, 2024.

A Publications

This chapter contains the publications discussed in Chapter 3.

A simple way to make neural networks robust against diverse image corruptions

Evgenia Rusak*[a,1,2], Lukas Schott*[a,1,2], Roland S. Zimmermann*[a,1], Julian Bitterwolf[b,1], Oliver Bringmann†[b,1], Matthias Bethge†[a,1], and Wieland Brendel†[a,1]

*equal contribution
†joint senior authors
[a]first.last@bethgelab.org
[b]first.last@uni-tuebingen.de
[1]University of Tübingen
[2]International Max Planck Research School for Intelligent Systems

Abstract

The human visual system is remarkably robust against a wide range of naturally occurring variations and corruptions like rain or snow. In contrast, the performance of modern image recognition models strongly degrades when evaluated on previously unseen corruptions. Here, we demonstrate that a simple but properly tuned training with additive Gaussian and Speckle noise generalizes surprisingly well to unseen corruptions, easily reaching the previous state of the art on the corruption benchmark ImageNet-C (with ResNet50) and on MNIST-C. We build on top of these strong baseline results and show that an adversarial training of the recognition model against uncorrelated worst-case noise distributions leads to an additional increase in performance. This regularization can be combined with previously proposed defense methods for further improvement.

1 Introduction

While Deep Neural Networks (DNNs) have surpassed the functional performance of humans in a range of complex cognitive tasks [He et al., 2016, Xiong et al., 2016, Silver et al., 2017, Campbell et al., 2002, OpenAI, 2018], they still lag behind humans in numerous other aspects. One fundamental shortcoming of machines is their lack of robustness against input perturbations. Even minimal perturbations that are hardly noticeable for humans can derail the predictions of high-performance neural networks.

For the purpose of this paper, we distinguish between two types of input perturbations. One type are minimal image-dependent perturbations specifically designed to fool a neural network with the smallest possible change to the input. These so-called *adversarial perturbations* have been the subject of hundreds of papers in the past five years, see e.g. [Szegedy et al., 2013, Madry et al., 2018, Schott et al., 2019, Gilmer et al., 2018]. Another, much less studied type are *common corruptions*. These perturbations occur naturally in many applications and include simple Gaussian or Salt and Pepper noise;

natural variations like rain, snow or fog; and compression artifacts such as those caused by JPEG encoding. All of these corruptions do not change the semantic content of the input, and thus, machine learning models should not change their decision-making behavior in their presence. Nonetheless, high-performance neural networks like ResNet50 [He et al., 2016] are easily confused by small local deformations [Geirhos et al., 2018]. The juxtaposition of adversarial examples and common corruptions was also explored in [Ford et al., 2019] where the authors discuss the relationship between both and encourage researchers working in the field of adversarial robustness to cross-evaluate the robustness of their models towards common corruptions.

We argue that in many practical applications robustness to common corruptions is often more relevant than robustness to artificially designed adversarial perturbations. Autonomous cars should not change their behavior in the face of unusual weather conditions such as hail or sand storms or small pixel defects in their sensors. Not-Safe-For-Work filters should not fail on images with unusual compression artifacts. Likewise, speech recognition algorithms should perform well regardless of the background music or sounds.

Besides its practical relevance, robustness to common corruptions is also an excellent target in its own right for researchers in the field of adversarial robustness and domain adaptation. Common corruptions can be seen as distributional shifts or as a weak form of adversarial examples that live in a smaller, constrained subspace.

Despite their importance, common corruptions have received relatively little attention so far. Only recently a modification of the ImageNet dataset [Russakovsky et al., 2014] to benchmark model robustness against common corruptions and perturbations has been published [Hendrycks and Dietterich, 2019] and is referred to as ImageNet-C. Now, this scheme has also been applied to other common datasets resulting in Pascal-C, Coco-C and Cityscapes-C [Michaelis et al., 2019] and MNIST-C [Mu and Gilmer, 2019].

Our contributions are as follows:

- We demonstrate that data augmentation with Gaussian or Speckle noise serves as a simple yet very strong baseline that is sufficient to surpass almost all previously proposed defenses against common corruptions on ImageNet-C for ResNet50. We further show that the magnitude of the additive noise is a crucial hyper-parameter to reach optimal robustness.

- Motivated by our strong results with baseline noise augmentations, we introduce a neural network-based *adversarial noise generator* that can learn arbitrary uncorrelated noise distributions that maximally fool a given recognition network when added to their inputs. We denote the resulting noise patterns as *adversarial noise*.

- We design and validate a constrained Adversarial Noise Training (ANT) scheme through which the recognition network learns to become robust against adversarial noise. We demonstrate that our ANT reaches state-of-the-art robustness on the corruption benchmark ImageNet-C for the commonly used ResNet50 architecture and on MNIST-C, even surpassing the already strong baseline noise augmentations. This result is not due to overfitting on the Noise categories of the respective benchmarks since we find equivalent results on the non-noise corruptions as well.

- We modify the adversarial noise generator to allow it to produce locally correlated noise thereby enabling it to learn more diverse noise distributions. Performing ANT

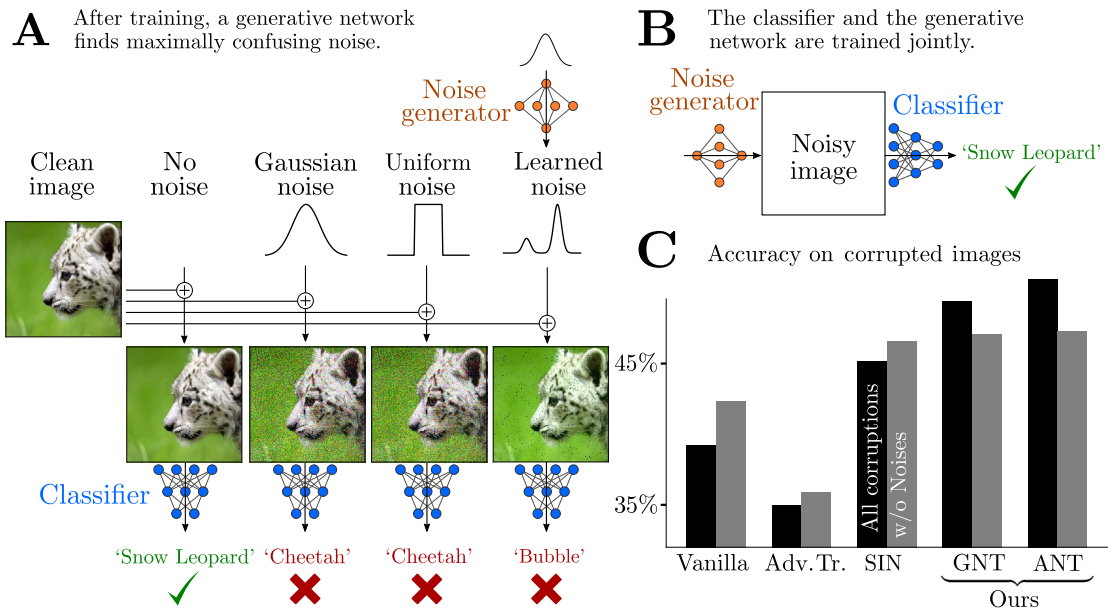


Figure 1. Outline of our approach. A: First, we train a generative network against a vanilla trained classifier to find the adversarial noise. B: To achieve robustness against adversarial noise, we train the classifier and the noise generator jointly. C: We measure the robustness against common corruptions for a vanilla, adversarially trained (Adv. Tr.), trained on Stylized ImageNet (SIN), trained via Gaussian data augmentation (GNT) and trained with the means of Adversarial Noise Training (ANT). With our methods, we achieve the highest accuracy on common corruptions, both on all and non-Noise categories.

with the modified noise generator, we observe an increase in robustness for the 'snow' corruption which is visually similar to our learned noise.

- We demonstrate a further increase in robustness when combining ANT with previous defense methods.
- We substantiate the claim that increased robustness against regular or universal adversarial perturbations does not imply increased robustness against common corruptions. This is not necessarily true vice-versa: Our noise trained recognition network has high accuracy on ImageNet-C and also slightly improved accuracy on adversarial attacks on clean ImageNet compared to a vanilla trained ResNet50.

We released our trained model weights along with evaluation code on github.com/bethgelab/game-of-noise.

2 Related work

Robustness against common corruptions Several recent publications study the vulnerability of DNNs to common corruptions. Dodge and Karam [2016] find that state-of-the-art image recognition networks are particularly vulnerable to blur and Gaussian

noise. Two recent studies compare humans and DNNs on recognizing corrupted images, showing that DNN performance drops much faster than human performance for increased perturbation sizes [Dodge and Karam, 2017a, Geirhos et al., 2018]. Yin et al. [2020] study the Fourier properties of common corruptions and link them to the robustness of differently trained classifiers.

Hendrycks and Dietterich [2019] introduce corrupted versions of standard datasets denoted as ImageNet-C, Tiny ImageNet-C and CIFAR10-C as standardized benchmarks for machine learning models and show that while state-of-the-art networks like ResNet50 are more accurate than outdated ones like AlexNet, their robustness is still negligible compared to humans. Similarly, common corruptions have been applied to and evaluated on COCO-C, Pascal-C, Cityscapes-C [Michaelis et al., 2019] and MNIST-C [Mu and Gilmer, 2019].

There have been attempts to increase robustness against common corruptions. Zhang [2019] integrate an anti-aliasing module from the signal processing domain in the ResNet50 architecture to restore the shift-equivariance which can get lost in deep CNNs. This results both in increased accuracy on clean data and increased generalization to corrupted image samples. Concurrent work to ours demonstrates that having more training data [Xie et al., 2019a, Mahajan et al., 2018] or using stronger backbones [Xie et al., 2019a, Michaelis et al., 2019] can significantly improve model performance on common corruptions.

A popular method to decrease overfitting and help the network generalize better to unseen data is to augment the training dataset by applying a set of (randomized) manipulations to the images [Mikołajczyk and Grochowski, 2018]. Furthermore, augmentation methods have also been applied to make the models more robust against image corruptions [Geirhos et al., 2019]. Geirhos et al. [2018] train ImageNet classifiers against a fixed set of corruptions but find no generalized robustness against unseen corruptions. However, they considered vastly higher noise severities than us. A similar observation is made by [Dodge and Karam, 2017b]. In a follow-up study, Geirhos et al. [2019] show that recognition models are biased towards texture and suggest this bias as one source of susceptibility for corruptions. They demonstrate that an increased shape bias also leads to increased accuracy on corrupted images. Hendrycks et al. [2020] is concurrent work to ours where the authors propose a data augmentation strategy which relies on combining and mixing augmentation chains. They also report strong robustness increases on ImageNet-C.

Augmentation with Gaussian noise has been used as a regularizer for smoothing the decision boundary of the classifier and was shown to be a provable adversarial defense [Cohen et al., 2019]. Conceptually, Ford et al. [2019] is the closest study to our work, since they also apply Gaussian noise to images to increase corruption robustness. They observe a low relative improvement in accuracy on corrupted images whereas we were able to outperform almost all previous baselines on the commonly used ResNet50 architecture.[1] They use a different architecture (InceptionV3 versus our ResNet50) and train a new model from scratch whereas we fine-tune a pretrained model. Another methodological difference is that we split every batch evenly in clean data and data augmented by Gaussian noise whereas they sample the standard deviation uniformly between 0 and one specific value and add noise to each image. Lopes et al. [2019] restrict the Gaussian noise to small image patches which improves accuracy but does not yield state-of-the-art performance on the

[1]To compare with Ford et al. [2019], we evaluate our approach for an InceptionV3 architecture, see our results in Appendix H.

ResNet50 architecture.

An alternative approach to data augmentation relies on stability training and shows robustness towards a range of distortions [Laermann et al., 2019].

Link between adversarial robustness and common corruptions There is currently no agreement on whether adversarial training increases robustness against common corruptions in the literature. Hendrycks and Dietterich [2019] report a robustness increase on common corruptions due to adversarial logit pairing on Tiny ImageNet-C. Ford et al. [2019] suggest a link between adversarial robustness and robustness against common corruptions, claim that increasing one robustness type should simultaneously increase the other, but report mixed results on MNIST and CIFAR10-C. Additionally, they also observe large drops in accuracy for adversarially trained networks and networks trained with Gaussian data augmentation compared to a vanilla classifier on certain corruptions. They do not evaluate adversarially robust classifiers on ImageNet. Fawzi et al. [2016] show that curvature constraints can both improve robustness against adversarial and random perturbations but they only present results on vanilla networks. On the other hand, Engstrom et al. [2019] report that increasing robustness against adversarial ℓ_∞ attacks does not increase robustness against translations and rotations, but they do not present results on noise. Kang et al. [2019] study robustness transfer between models trained against ℓ_1, ℓ_2, ℓ_∞ adversaries / elastic deformations and JPEG artifacts. They observe that adversarial training increases robustness against elastic and JPEG corruptions on a 100-class subset of ImageNet. This result contradicts our findings on full ImageNet as we see a slight decline in accuracy on those two classes for the adversarially trained model from [Xie et al., 2019b] and severe drops in accuracy on other corruptions. Jordan et al. [2019] show that adversarial robustness does not transfer easily between attack classes. Tramèr and Boneh [2019] also argue in favor of a trade-off between different robustness types. For a simple and natural classification task, they prove that adversarial robustness towards l_∞ perturbations does neither transfer to l_1 nor to input rotations and translations, and vice versa. They support their formal analysis with experiments on MNIST and CIFAR10.

Universal adversarial perturbations Universal adversarial perturbations (UAPs) [Moosavi-Dezfooli et al., 2017] are perturbations which, if added to any image, fool a given recognition model. This contrasts with regular adversarial perturbations, which need to be designed specifically for every single image. Hayes and Danezis [2017] generate UAPs by training so-called universal adversarial networks (UANs). They also train the classifier jointly with the UAN but manage to only slightly increase robustness against UAPs. Other defenses against UAPs are similarly based on adversarial training [Metzen, 2018, Shafahi et al., 2018, Mummadi et al., 2019, Pérolat et al., 2018].

UAPs are very different from our adversarial noise setting in that UAPs can learn perturbations with global, image-wide features while our adversarial noise is identically distributed over pixels and thus inherently local.

3 Methods

Our method section can be roughly split into two parts. In the first part, we revisit Gaussian data augmentation as a method to increase robustness against image corruptions. The second part contains two items (Fig. 1). First, we devise and train a generator neural network to produce spatially uncorrelated noise that is adversarial to a given recognition network (section 3.2.1 and Fig. 1A). Second, we formulate a constrained adversarial training scheme, which allows us to train the recognition model jointly with the noise generator (section 3.2.2 and Fig. 1B) that we call Adversarial Noise Training (ANT). Finally, in section 3.4 we explain our evaluation methods on corrupted images (Fig. 1C).

3.1 Training with Gaussian noise

As discussed in section 2, several researchers have tried using Gaussian noise as a method to increase robustness towards common corruptions with mixed results. In this work, we revisit the approach of Gaussian data augmentation and increase its efficacy. In contrast to previous work, we treat the standard deviation σ of the distribution as a hyper-parameter of the training and measure its influence on robustness.

To formally introduce the objective, let $\mathcal{D}$ be the data distribution over input pairs $(\boldsymbol{x}, y)$ with $\boldsymbol{x} \in \mathbb{R}^N$ and $y \in \{1, ..., k\}$. We train a differentiable classifier $f_\theta(\boldsymbol{x})$ by minimizing the risk on a dataset with additive Gaussian noise

$$\mathop{\mathbb{E}}_{\boldsymbol{x},y\sim\mathcal{D}} \mathop{\mathbb{E}}_{\boldsymbol{\delta}\sim\mathcal{N}(\mathbf{0},\sigma^2\mathbb{1})} \left[\mathcal{L}_{\text{CE}}\left(f_\theta(\text{clip}((\boldsymbol{x}+\boldsymbol{\delta})), y)\right], \tag{1}$$

where σ is the standard deviation of the Gaussian noise and $\boldsymbol{x} + \boldsymbol{\delta}$ is clipped to the input range $[0, 1]^N$. The standard deviation is either kept fixed or is chosen uniformly from a fixed set of standard deviations. In both cases, the possible standard deviations are chosen from a small set of nine values inspired by the noise variance in the ImageNet-C dataset (cf. section 3.4). To maintain high accuracy on clean data, we only perturb 50% of the training data with Gaussian noise within each batch.

3.2 Adversarial noise

3.2.1 Learning Adversarial Noise

Our goal is to find a noise distribution $p_\phi(\boldsymbol{\delta})$, $\boldsymbol{\delta} \in \mathbb{R}^N$ such that noise samples added to $\boldsymbol{x}$ maximally confuse the classifier f_θ. More concisely, we optimize

$$\max_\phi \mathop{\mathbb{E}}_{\boldsymbol{x},y\sim\mathcal{D}} \mathop{\mathbb{E}}_{\boldsymbol{\delta}\sim p_\phi(\boldsymbol{\delta})} \left[\mathcal{L}_{\text{CE}}\left(f_\theta(\text{clip}_\epsilon(\boldsymbol{x}+\boldsymbol{\delta})), y\right)\right], \tag{2}$$

where clip is an operator that clips all values to the valid interval (i.e. $\text{clip}(\boldsymbol{x}+\boldsymbol{\delta}) \in [0, 1]^N$) and $||\boldsymbol{\delta}||_2 = \epsilon$. [2]

We do not have to explicitly model the probability density function $p_\phi(\boldsymbol{\delta})$ since optimizing Eq. (2) only involves samples drawn from $p_\phi(\boldsymbol{\delta})$. We model the samples from

[2]We apply the method derived in Rauber and Bethge [2020] and rescale the perturbation by a factor γ to obtain the desired ℓ_2 norm; despite the clipping, the squared ℓ_2 norm is a piece-wise linear function of γ^2 that can be inverted to find the correct scaling factor γ.

$p_\phi(\boldsymbol{\delta})$ as the output of a neural network $g_\phi : \mathbb{R}^N \to \mathbb{R}^N$ which gets its input from a normal distribution $\boldsymbol{\delta} = g_\phi(\boldsymbol{z})$ where $\boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, \mathbb{1})$. We enforce the independence property of $p_\phi(\boldsymbol{\delta}) = \prod_n p_\phi(\delta_n)$ by constraining the network architecture of the noise generator g_ϕ to only consist of convolutions with 1x1 kernels. Lastly, the projection onto a sphere $||\boldsymbol{\delta}||_2 = \epsilon$ is achieved by scaling the generator output with a scalar while clipping $\boldsymbol{x} + \boldsymbol{\delta}$ to the valid range $[0, 1]^N$. This fixed size projection (hyper-parameter) is motivated by the fact that Gaussian noise training with a single, fixed σ achieved the highest accuracy. [3]

The noise generator g_ϕ has four 1x1 convolutional layers with ReLU activations and one residual connection from input to output. The convolutional weights are initialized such that the noise generator outputs a Gaussian distribution.

3.2.2 Adversarial Noise Training

To increase robustness, we now train the classifier f_θ to minimize the risk under adversarial noise distributions jointly with the noise generator

$$\min_\theta \max_\phi \mathop{\mathbb{E}}_{\boldsymbol{x},y\sim\mathcal{D}} \mathop{\mathbb{E}}_{\boldsymbol{\delta}\sim p_\phi(\boldsymbol{\delta})} \left[\mathcal{L}_{\text{CE}}\left(f_\theta(\text{clip}_\epsilon(\boldsymbol{x} + \boldsymbol{\delta})), y\right)\right], \tag{3}$$

where again $\boldsymbol{x} + \boldsymbol{\delta} \in [0, 1]^N$ and $||\boldsymbol{\delta}||_2 = \epsilon$. For a joint adversarial training, we alternate between an outer loop of classifier update steps and an inner loop of generator update steps. This is also depicted schematically in Fig. 1B. Note that in regular adversarial training, e.g. [Madry et al., 2018], $\boldsymbol{\delta}$ is optimized directly whereas we optimize a constrained distribution over $\boldsymbol{\delta}$.

To maintain high classification accuracy on clean samples, we sample every mini-batch so that they contain 50% clean data and perturb the rest. The current state of the noise generator is used to perturb 30% of this data and the remaining 20% are augmented with samples chosen randomly from previous distributions. For this, the noise generator states are saved at regular intervals. The latter method is inspired by experience replay from reinforcement learning [Mnih et al., 2015] and is used to keep the classifier from forgetting previous adversarial noise patterns.

To prevent the noise generator from being stuck in a local minimum, we halt the Adversarial Noise Training (ANT) at regular intervals and train a new noise generator from scratch. This noise generator is trained against the current state of the classifier to find a current optimum. The new noise generator replaces the former noise generator in the ANT. This technique has proven crucial to train a robust classifier.

Learning locally correlated adversarial noise We modify the architecture of the noise generator defined in Eq. 2 to allow for local spatial correlations and thereby enable the generator to learn more diverse distributions. Since we seek to increase model robustness towards image corruptions such as rain or snow that produce locally correlated patterns, it is natural to include local patterns in the manifold of learnable distributions. We replace the 1x1 kernels in one network layer with 3x3 kernels limiting the maximum correlation length of the output noise sample to 3x3 pixels. We indicate the correlation length of noise generator used for the constrained adversarial noise training as $\text{ANT}^{1\text{x}1}$ or $\text{ANT}^{3\text{x}3}$.

[3] We also experimented with an adaptive sphere radius ϵ which grows with the classifier's accuracy. However, we did not see any improvements and followed Occam's razor.

3.3 Combining Adversarial Noise Training with stylization

As demonstrated by Geirhos et al. [2019], using random stylization as data augmentation increases the accuracy on ImageNet-C. The robustness gains are attributed to a stronger shape bias of the classifier. We combine our ANT and the stylization approach to achieve robustness gains from both. To incorporate stylized data into the training scheme described in the previous section, we change the way we sample every mini-batch: we split the batches into clean data (25%), stylized data (30%) and data perturbed by the noise generator (45%). Like before, we choose the latter samples with roughly equal probability from the current state of the noise generator and from any previous distribution.

3.4 Evaluation on corrupted images

Evaluation of noise robustness We evaluate the robustness of a model by sampling a Gaussian noise vector $\boldsymbol{\delta}$. We then do a line search along the direction $\boldsymbol{\delta}$ starting from the original image $\boldsymbol{x}$ until it is misclassified. We denote the resulting minimal perturbation as $\boldsymbol{\delta}_{\text{min}}$. The robustness of a model is then denoted by the median[4] over the test set

$$\epsilon^* = \underset{\boldsymbol{x},y\sim\mathcal{D}}{\text{median}}\ ||\boldsymbol{\delta}_{\text{min}}||_2, \tag{4}$$

with $f_\theta(\boldsymbol{x} + \boldsymbol{\delta}_{\text{min}}) \neq y$ and $\boldsymbol{x} + \boldsymbol{\delta}_{\text{min}} \in [0,1]^N$.

Note that a higher ϵ^* denotes a more robust classifier. To test the robustness against adversarial noise, we train a new noise generator at the end of the Adversarial Noise Training until convergence and evaluate it according to Eq. (4).

ImageNet-C The ImageNet-C benchmark[5] [Hendrycks and Dietterich, 2019] is a conglomerate of 15 diverse corruption types that were applied to the validation set of ImageNet. The corruptions are organized into four main categories: Noise, Blur, Weather, and Digital and have five levels of severities to reflect the varying intensities of common corruptions. The MNIST-C benchmark is created similarly to ImageNet-C [Mu and Gilmer, 2019] with a slightly different set of corruptions. Our main evaluation metric for both benchmarks is the Top-1 accuracy on corrupted images for each Noise category averaged over the severities; we also report the Top-5 accuracy on ImageNet-C. Since some works report the 'mean Corruption Error' (mCE) instead of accuracy, we also include results on mCE in Appendix D.

We evaluate all proposed methods for ImageNet-C on the ResNet50 architecture for better comparability to previous methods, e.g. [Geirhos et al., 2019, Lopes et al., 2019, Zhang, 2019]. The clean ImageNet accuracy of the used architecture highly influences the results and could be seen as an upper bound for the accuracy on ImageNet-C. Note that our approach is independent of the used architecture and could in principle be applied to any differentiable network.

[4]Samples for which no ℓ_2-distance allows us to manipulate the classifier's decision contribute a value of ∞ to the median.

[5]For the evaluation, we use the JPEG compressed images from github.com/hendrycks/robustness as is advised by the authors to ensure reproducibility. We note that Ford et al. [2019] report a decrease in performance when the compressed JPEG files are used as opposed to applying the corruptions directly in memory without compression artifacts.

4 Results

For our experiments on ImageNet, we use a classifier that was pretrained on ImageNet. For the experiments on MNIST, we use the architecture from [Madry et al., 2018] for comparability. We use PyTorch [Paszke et al., 2017] for all of our experiments. All technical details, hyper-parameters and the architecture of the noise generator can be found in Appendix A-B.

4.1 (In-)Effectiveness of regular adversarial training to increase robustness towards common corruptions

As our first experiment, we evaluate whether robustness against regular adversarial examples generalizes to robustness against common corruptions. We display the Top-1 accuracy of vanilla and adversarially trained models in Table 1; detailed results on individual corruptions can be found in Appendix C. For all tested models, we find that regular ℓ_∞ adversarial training can strongly decrease the robustness towards common corruptions, especially for the corruption types Fog and Contrast. Universal adversarial training [Shafahi et al., 2018], on the other hand, leads to severe drops on some corruptions but the overall accuracy on ImageNet-C is slightly increased relative to the vanilla baseline model (AlexNet). Nonetheless, the absolute ImageNet-C accuracy of 22.2% is still very low. These results disagree with two previous studies which reported that (1) adversarial logit pairing[6] (ALP) increases robustness against common corruptions on Tiny ImageNet-C [Hendrycks and Dietterich, 2019], and that (2) adversarial training can increase robustness on the CIFAR10-C data set [Ford et al., 2019].

We evaluate adversarially trained models on MNIST-C and present the results and their discussion in Appendix E. The results on MNIST-C show the same tendency as on ImageNet-C: adversarially trained models have lower accuracy on MNIST-C and thus indicate that adversarial robustness does not transfer to robustness against common corruptions. This corroborates the results of [Ford et al., 2019] on MNIST who also found that an adversarially robust model had decreased robustness towards a set of common corruptions.

model	IN-C	IN-C w/o Noises
Vanilla RN50	39.2%	42.3%
Adv. Training [Shafahi et al., 2019]	29.1%	32.0%
Vanilla RN152	45.0%	47.9%
Adv. Training [Xie et al., 2019b]	35.0%	35.9%
Vanilla AlexNet	21.1%	23.9%
Universal Adv. Training [Shafahi et al., 2018]	22.2%	23.1%

Table 1: Top-1 accuracy on ImageNet-C and ImageNet-C without the Noise categories (higher is better). Regular adversarial training decreases robustness towards common corruptions; universal adversarial training seems to slightly increase it.

[6]Note that ALP was later found to not increase adversarial robustness [Engstrom et al., 2018].

4.2 Effectiveness of Gaussian data augmentation to increase robustness towards common corruptions

We fine-tune a pretrained image classifier with Gaussian data augmentation from the distribution $\mathcal{N}(\mathbf{0}, \sigma^2 \mathbb{1})$ and vary σ. We try two different settings: in one we choose a single noise level σ while in the second we sample σ uniformly from a set of multiple possible values. The Top-1 accuracy of the fine-tuned models on ImageNet-C and a comparison to a vanilla trained model is shown in Fig. 2. Each black point shows the performance of one model fine-tuned with one specific σ; the vanilla trained model is marked by the point at $\sigma = 0$. The horizontal lines indicate that the model is fine-tuned with Gaussian noise where σ is sampled from a set for each image. For example, for the dark green line, as indicated by the stars, we sample σ from the set $\{0.08, 0.12, 0.18, 0.26, 0.38\}$, which corresponds to the Gaussian corruption of ImageNet-C. We show both the results on the full ImageNet-C evaluation set and the results on ImageNet-C without Noises (namely Blur, Weather and Digital) since Gaussian noise is part of the test set. To give a feeling of how the different σ-levels manifest themselves in an image, we include example images for all σ-levels in Appendix G.

There are three important results evident from Fig. 2:

1. Gaussian noise generalizes well to the non-noise corruptions of the ImageNet-C evaluation dataset and is a powerful baseline. This is a surprising result as it was shown in several recent works that training on Gaussian or uniform noise does not generalize to other corruption types [Geirhos et al., 2018, Lopes et al., 2019] or that the effect is very weak [Ford et al., 2019].

2. The standard deviation σ is a crucial hyper-parameter and has an optimal value of about $\sigma = 0.5$ for ResNet50.

3. If σ is chosen well, using a single σ is enough, sampling from a set of σ values is not necessary and even detrimental for robustness against non-noise corruptions.

In the following Results sections, we will compare Gaussian data augmentation to our Adversarial Noise Training approach and baselines from the literature. For this, we will use the models with the overall best-performance: The model $\text{GN}_{0.5}$ that was trained with Gaussian data augmentation with a single $\sigma = 0.5$ and the model GN_{mult} where σ was sampled from the set $\{0.08, 0.12, 0.18, 0.26, 0.38\}$.

4.3 Evaluation of the severity of adversarial noise as an attack

In this section, we try to answer the question: Can we learn the most severe uncorrelated additive noise distribution for a classifier?

Following the success of simple uncorrelated Gaussian noise data augmentation (section 4.2) and the ineffectiveness of regular adversarial training (section 4.1) which allows for highly correlated patterns, we restrict our learned noise distribution to be sampled independently for each pixel. We denote this learned adversarial noise distribution $p_\phi(\boldsymbol{\delta})$ as adversarial noise (AN, section 3.2.1).

To measure the effectiveness of our adversarial noise, we report the median perturbation size ϵ^* that is necessary for a misclassification for each image in the test set.

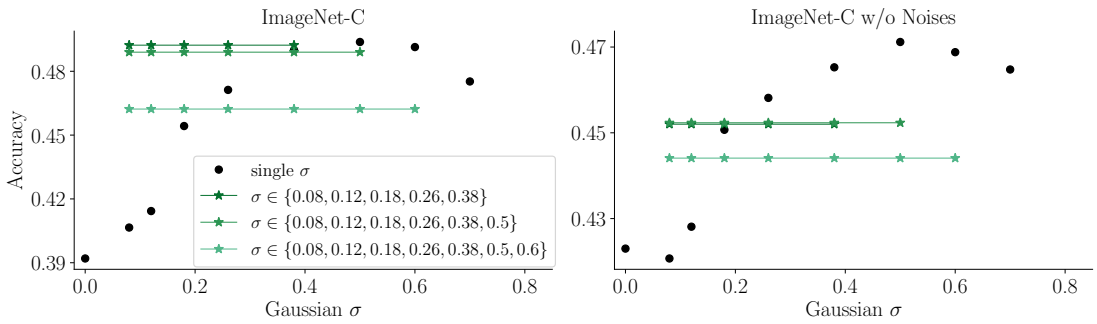


Figure 2. Top-1 accuracy on ImageNet-C (left) and ImageNet-C without the noise corruptions (right) of a ResNet50 architecture fine-tuned with Gaussian data augmentation of varying standard deviation σ. We train on Gaussian noise sampled from a distribution with a single σ (black dots) and on distributions where σ is sampled from different sets (green lines with stars). We also compare to a vanilla trained model at $\sigma = 0$.

In Table 2, we see that our AN is much more effective at fooling the classifier compared to Gaussian and uniform noise. This is also reflected qualitatively in the noisy images in Fig. 1 where we show images at the decision boundary: The amount of noise to fool the classifier is smaller in the right-most image produced by the generative network than in the central images (Gaussian and uniform noise).

model	ϵ^*_{GN}	ϵ^*_{UN}	ϵ^*_{AN}
Vanilla RN50	39.0	39.1	16.2

Table 2: Median ℓ_2 perturbation size ϵ^* that is required to misclassify an image for Gaussian (GN), uniform (UN) and adversarial noise (AN). A lower ϵ^* indicates a more severe noise, since on average, a smaller perturbation size is sufficient to fool a classifier.

4.4 Evaluation of Adversarial Noise Training as a defense

In the previous section, we established a method for learning the most adversarial noise distribution for a classifier. Now, we utilize it for a joint Adversarial Noise Training ($\text{ANT}^{1\text{x}1}$) where we simultaneously train the noise generator and classifier (see section 3.2.2). This leads to substantially increased robustness against Gaussian, uniform and adversarial noise, see Table 3.

The robustness of models that were trained via Gaussian data augmentation also increases, but on average much less compared to the model trained with $\text{ANT}^{1\text{x}1}$. To evaluate the robustness against adversarial noise, we train four noise generators from scratch with different random seeds and measure $\epsilon^*_{\text{AN1x1}}$. We report the mean value and the standard deviation over the four runs.

To better understand this effect, we visualize the temporal evolution of the probability density function $p_\phi(\delta_n)$ of uncorrelated noise during $\text{ANT}^{1\text{x}1}$ and samples of correlated

ϵ =15.0 ϵ =30.0 ϵ =60.0 ϵ =120.0

Figure 3. Example images for different perturbation sizes ϵ^* for additive Gaussian noise on ImageNet.

noise during ANT$^{3\text{x}3}$ in Fig. 4. This shows that the generator converges to different distributions during ANT. Therefore, the classifier has been trained against a rich variety of distributions.

For ANT$^{1\text{x}1}$, we have also tested how the classifier's performance changes if it is trained against adversarial noise sampled randomly from $p_\phi(\delta_n)$. The accuracy on ImageNet-C decreases slightly compared to regular ANT$^{1\text{x}1}$: 51.1%/ 71.9% (Top-1/ Top-5) on full ImageNet-C and 47.3%/ 68.3% (Top-1/ Top-5) on ImageNet-C without the Noise categories.

model	clean acc.	ϵ^*_{GN}	ϵ^*_{UN}	$\epsilon^*_{\text{AN1x1}}$
Vanilla RN50	76.1%	39.0	39.1	15.7 $\pm$ 0.6
GNT$\sigma_{0.5}$	75.9%	74.8	74.9	31.8 $\pm$ 3.9
ANT$^{1\text{x}1}$	76.0%	136.7	137.0	95.4 $\pm$ 5.7

Table 3: Accuracy on clean data and robustness of differently trained models as measured by the median perturbation size ϵ^*. A higher ϵ^* indicates a more robust model. To provide an intuition for the perturbation sizes indicated by ϵ^*, we show example images for Gaussian noise below in Fig. 3 and a larger Figure for different noise types in Appendix I.

4.5 Comparison of different methods to increase robustness towards common corruptions

We now revisit common corruptions on ImageNet-C and compare the robustness of differently trained models. Since Gaussian noise is part of ImageNet-C, we train another baseline model with data augmentation using the Speckle noise corruption from the ImageNet-C holdout set. We later denote the cases where the corruptions present during training are part of the test set by putting corresponding accuracy values in brackets.

The Top-1 accuracies on the full ImageNet-C dataset and ImageNet-C without the Noise corruptions are displayed in Table 4; detailed results on individual corruptions in terms of accuracy and mCE are shown in Tables 3 and 4, Appendix D. We also calculate the accuracy on corruptions without the Noise category as our approaches to add Gaussian noise or to perform ANT$^{1\text{x}1}$ overfits to the Noise categories. By allowing the noise generator to learn local correlations, we lift the independence constraint and argue that the learned noise distributions are now different enough to not cause overfitting to

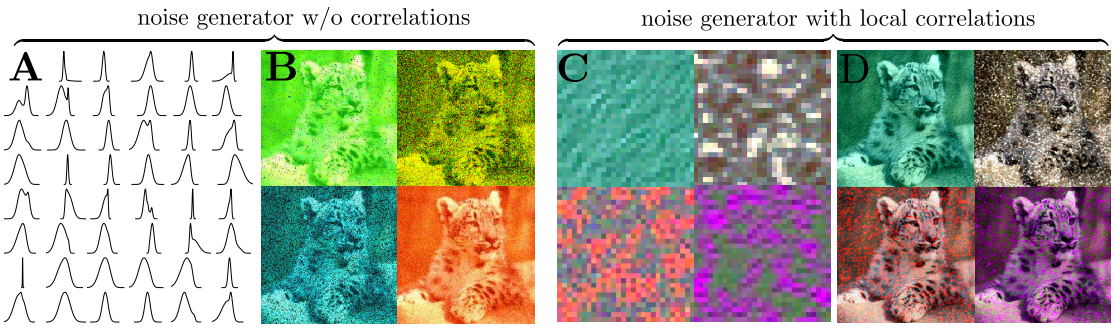


Figure 4. A: Examples of learned probability densities over the grayscale version of the noise $\boldsymbol{\delta}_n$ during ANT^{1x1} where each density corresponds to one local minimum; B: Example images with sampled uncorrelated adversarial noise; C: Example patches of locally correlated noise with a size of 28x28 pixels learned during ANT^{3x3}; D: Example images with sampled correlated adversarial noise.

the Noise categories (see Fig. 4C+D). For this reason, we do not put the accuracy values of the classifier obtained via ANT^{3x3} in brackets.

Additionally, we compare our results with several baseline models from the literature. The ImageNet-C benchmark has been published recently and we use all baselines we could find for a ResNet50 architecture:

1. Shift Inv: The model is modified to enhance shift-equivariance using anti-aliasing [Zhang, 2019].[7]

2. Patch GN: The model was trained on Gaussian patches [Lopes et al., 2019]. Since no model weights are released, we can only include their Top-1 ImageNet-C accuracy values from their paper (and not the Top-5).

3. SIN+IN: The model was trained on a stylized version of ImageNet [Geirhos et al., 2019].[8]

4. AugMix: Hendrycks et al. [2020] trained their model using diverse augmentations.[9] They use image augmentations from AutoAugment [Cubuk et al., 2018] and exclude the contrast, color, brightness, sharpness, and Cutout operations to make sure that the test set of ImageNet-C is disjoint from the training set. We would like to highlight the difficulty in clearly distinguishing between the augmentations used during training and testing as there might be a certain overlap. This can be seen by the visual similarity between the Posterize operation and the JPEG corruption (see Appendix J).

The results on full ImageNet-C are striking (see Table 4): a very simple baseline, namely a model trained with Speckle noise data augmentation, beats almost all previous baselines reaching an accuracy of 46.4% which is larger than the accuracy of SIN+IN

[7]Weights were taken from github.com/adobe/antialiased-cnns.
[8]Weights were taken from github.com/rgeirhos/texture-vs-shape.
[9]Weights were taken from github.com/google-research/augmix.

model	IN clean acc. [%]	IN-C Top-1 [%]	IN-C Top-5 [%]	IN-C w/o Noises Top-1 [%]	IN-C w/o Noises Top-5 [%]
Vanilla RN50	76.1	39.2	59.3	42.3	63.2
Shift Inv [Zhang, 2019]	77.0	41.4	61.8	44.2	65.1
Patch GN [Lopes et al., 2019]	76.0	(43.6)	(n.a.)	43.7	n.a.
SIN+IN [Geirhos et al., 2019]	74.6	45.2	66.6	46.6	68.2
AugMix [Hendrycks et al., 2020]	77.5	48.3	69.2	50.4	71.8
GNT_{mult}	76.1	(49.2)	(70.2)	45.2	66.2
$\text{GNT}\sigma_{0.5}$	75.9	(49.4)	(70.6)	47.1	68.3
$\text{ANT}^{1\text{x}1}$	76.0	(51.1)	(72.2)	47.7	68.8
$\text{ANT}^{1\text{x}1}$+SIN	74.9	(52.2)	(73.6)	49.2	70.6
$\text{ANT}^{1\text{x}1}$ w/o EP	75.7	(48.9)	(70.2)	46.5	67.7
$\text{ANT}^{3\text{x}3}$	76.1	50.4	71.5	47.0	68.1
$\text{ANT}^{3\text{x}3}$+SIN	74.1	**52.6**	**74.4**	**50.6**	**72.5**

Table 4: Average accuracy on clean data, average Top-1 and Top-5 accuracies on ImageNet-C and ImageNet-C without the Noise categories (higher is better). We compare the results obtained by the means of Gaussian (GNT) and Speckle noise data augmentation and with Adversarial Noise Training (ANT) to several baselines. Gray numbers in brackets indicate scenarios where a corruption from the test set was used during training.

(45.2%) and close to AugMix (48.3%). The $\text{GN}\sigma_{0.5}$ surpasses SIN+IN not only on the Noise categories but also on almost all other corruptions, see a more detailed breakdown in Table 9, Appendix D. The $\text{ANT}^{3\text{x}3}$+SIN model produces the best results on ImageNet-C both with and without Noises. Thus, it is slightly superior to Gaussian data augmentation and pure $\text{ANT}^{3\text{x}3}$. Comparing $\text{ANT}^{1\text{x}1}$ and $\text{ANT}^{3\text{x}3}$, we observe that $\text{ANT}^{3\text{x}3}$ performs better than $\text{ANT}^{1\text{x}1}$ in the 'snow' corruption. We attribute this to the successful modeling capabilities of locally correlated patterns resembling snow of the 3x3 noise generator. We perform an ablation study to investigate the necessity of experience replay and note that we lose roughly 2% without it ($\text{ANT}^{1\text{x}1}$ w/o EP vs $\text{ANT}^{1\text{x}1}$). We also test how the classifier's performance changes if it is trained against adversarial noise sampled randomly from $p_\phi(\delta_n)$. The accuracy on ImageNet-C decreases slightly compared to regular $\text{ANT}^{1\text{x}1}$: 51.1%/ 71.9% (Top-1/ Top-5) on full ImageNet-C and 47.3%/ 68.3% (Top-1/ Top-5) on ImageNet-C without the noise category. We include additional results for $\text{ANT}^{1\text{x}1}$ with a DenseNet121 architecture **?** and for varying parameter counts of the noise generator in Appendix K.

For MNIST, we train a model with Gaussian data augmentation and via $\text{ANT}^{1\text{x}1}$. We achieve similar results with both approaches and report a new state-of-the-art accuracy on MNIST-C: 92.4%. The results on MNIST-C can be found in Appendix E.

4.6 Robustness towards adversarial perturbations

As regular adversarial training can decrease the accuracy on common corruptions, it is also interesting to check what happens vice-versa: How does a model which is robust on common corruptions behave under adversarial attacks?

Both our $\text{ANT}^{1\text{x}1}$ and GNT models have slightly increased ℓ_2 and ℓ_∞ robustness scores compared to a vanilla trained model, see Table 5. We tested this using the white-box attacks PGD Madry et al. [2017] and DDN Rony et al. [2019]. Expectedly, an adversarially trained model has higher adversarial robustness compared to $\text{ANT}^{1\text{x}1}$ or GNT. In this experiment, we only verify that we do not unintentionally reduce adversarial robustness compared to a vanilla ResNet50. For details, see Appendix E for MNIST and Appendix F for ImageNet.

model	clean acc. [%]	ℓ_2 acc. [%]	ℓ_∞ acc. [%]
Vanilla RN50	75.2	41.1	18.1
$\text{GNT}\sigma_{0.5}$	75.3	49.0	28.1
$\text{ANT}^{1\text{x}1}$	75.7	50.1	28.6
Adv. Training Shafahi et al. [2019]	60.5	**58.1**	**58.5**

Table 5: Adversarial robustness on ℓ_2 ($\epsilon = 0.12$) and ℓ_∞ ($\epsilon = 0.001$) compared to a Vanilla ResNet50.

5 Discussion & Conclusion

So far, attempts to use simple noise augmentations for general robustness against common corruptions have produced mixed results, ranging from no generalization from one noise to other noise types [Geirhos et al., 2018] to only marginal robustness increases [Ford et al., 2019, Lopes et al., 2019]. In this work, we demonstrate that carefully tuned additive noise patterns in conjunction with training on clean samples can surpass almost all current state-of-the-art defense methods against common corruptions. By drawing inspiration from adversarial training and experience replay, we additionally show that training against simple uncorrelated worst-case noise patterns outperforms our already strong baseline defense, with additional gains to be made in combination with previous defense methods like stylization training [Geirhos et al., 2019].

There are still a few corruption types (e.g. Motion or Zoom blurs) on which our method is not state of the art, suggesting that additional gains are possible. Future extensions of this work may combine noise generators with varying correlation lengths, add additional interactions between noise and image (e.g. multiplicative interactions or local deformations) or take into account local image information in the noise generation process to further boost robustness across many types of image corruptions.

6 Acknowledgements

The authors thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting Evgenia Rusak and Lukas Schott. The authors thank Yash Sharma for helpful discussions and Alexander Ecker, Robert Geirhos and Dylan Paiton for helpful feed-back while writing the manuscript. The authors thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting Evgenia Rusak, Lukas Schott and Roland S. Zimmermann. Oliver Bringmann and Evgenia Rusak have been partially supported by the Deutsche Forschungsgemeinschaft (DFG) in the priority program 1835 "Cooperatively Interacting Automobiles" under grant BR2321/5-1 and BR2321/5-2. This work was supported by the German Federal Ministry of Education and Research (BMBF): Tübingen AI Center, FKZ: 01IS18039A, by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation): Germany's Excellence Strategy – EXC 2064/1 – 390727645 and SFB 1233, Robust Vision: Inference Principles and Neural Mechanisms, TP XX, project number: 276693517 and additionally by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior/Interior Business Center (DoI/IBC) contract number D16PC00003. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoI/IBC, or the U.S. Government.

References

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Wayne Xiong, Jasha Droppo, Xuedong Huang, Frank Seide, Mike Seltzer, Andreas Stolcke, Dong Yu, and Geoffrey Zweig. Achieving human parity in conversational speech recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2016.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, L Robert Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy P. Lillicrap, Fan Fong Celine Hui, Laurent Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis. Mastering the game of go without human knowledge. *Nature*, 550:354–359, 2017.

Murray Campbell, A. Joseph Hoane, Jr., and Feng-hsiung Hsu. Deep blue. *Artif. Intell.*, 134(1-2):57–83, January 2002. ISSN 0004-3702. doi: 10.1016/S0004-3702(01)00129-1. URL `http://dx.doi.org/10.1016/S0004-3702(01)00129-1`.

OpenAI. Openai five. `https://blog.openai.com/openai-five/`, 2018.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJzIBfZAb.

Lukas Schott, Jonas Rauber, Matthias Bethge, and Wieland Brendel. Towards the first adversarially robust neural network model on MNIST. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=S1EHOsC9tX.

Justin Gilmer, Luke Metz, Fartash Faghri, Samuel S. Schoenholz, Maithra Raghu, Martin Wattenberg, and Ian J. Goodfellow. Adversarial spheres. *CoRR*, abs/1801.02774, 2018. URL http://arxiv.org/abs/1801.02774.

Robert Geirhos, Carlos R. M. Temme, Jonas Rauber, Heiko H. Schütt, Matthias Bethge, and Felix A. Wichmann. Generalisation in humans and deep neural networks. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 7538–7550. Curran Associates, Inc., 2018. URL http://papers.nips.cc/paper/7982-generalisation-in-humans-and-deep-neural-networks.pdf.

Nic Ford, Justin Gilmer, Nicolas Carlini, and Dogus Cubuk. Adversarial examples are a natural consequence of test error in noise. *ICML*, 2019.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Fei-Fei Li. Imagenet large scale visual recognition challenge. *CoRR*, abs/1409.0575, 2014. URL http://arxiv.org/abs/1409.0575.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=HJz6tiCqYm.

Claudio Michaelis, Benjamin Mitzkus, Robert Geirhos, Evgenia Rusak, Oliver Bringmann, Alexander S Ecker, Matthias Bethge, and Wieland Brendel. Benchmarking robustness in object detection: Autonomous driving when winter is coming. *arXiv preprint arXiv:1907.07484*, 2019.

Norman Mu and Justin Gilmer. MNIST-C: A robustness benchmark for computer vision. *arXiv preprint arXiv:1906.02337*, 2019.

Samuel Fuller Dodge and Lina J. Karam. Understanding how image quality affects deep neural networks. *2016 Eighth International Conference on Quality of Multimedia Experience (QoMEX)*, pages 1–6, 2016.

Samuel F. Dodge and Lina J. Karam. A study and comparison of human and deep learning recognition performance under visual distortions. *CoRR*, abs/1705.02498, 2017a. URL http://arxiv.org/abs/1705.02498.

Dong Yin, Raphael Gontijo Lopes, Jonathon Shlens, Ekin D. Cubuk, and Justin Gilmer. A Fourier perspective on model robustness in computer vision. *NeurIPS*, 2020.

Richard Zhang. Making convolutional networks shift-invariant again. *ICML*, 2019.

Qizhe Xie, Eduard Hovy, Minh-Thang Luong, and Quoc V. Le. Self-training with noisy student improves imagenet classification. *arXiv preprint arXiv:1911.04252*, 2019a.

Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens van der Maaten. Exploring the limits of weakly supervised pretraining. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 181–196, 2018.

Agnieszka Mikołajczyk and Michał Grochowski. Data augmentation for improving deep learning in image classification problem. *2018 International Interdisciplinary PhD Workshop (IIPhDW)*, pages 117–122, 2018.

Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A. Wichmann, and Wieland Brendel. Imagenet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=Bygh9j09KX`.

Samuel F. Dodge and Lina J. Karam. Quality resilient deep neural networks. *CoRR*, abs/1703.08119, 2017b. URL `http://arxiv.org/abs/1703.08119`.

Dan Hendrycks, Norman Mu, Ekin Dogus Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=S1gmrxHFvB`.

Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 1310–1320, Long Beach, California, USA, 09–15 Jun 2019. PMLR.

Raphael Gontijo Lopes, Dong Yin, Ben Poole, Justin Gilmer, and Ekin D. Cubuk. Improving robustness without sacrificing accuracy with patch gaussian augmentation. *CoRR*, abs/1906.02611, 2019. URL `http://arxiv.org/abs/1906.02611`.

Jan Laermann, Wojciech Samek, and Nils Strodthoff. Achieving generalizable robustness of deep neural networks by stability training. In *German Conference on Pattern Recognition*, pages 360–373. Springer, 2019.

Alhussein Fawzi, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. Robustness of classifiers: from adversarial to random noise. *NIPS*, 2016.

Logan Engstrom, Dimitris Tsipras, Ludwig Schmidt, and Aleksander Madry. A rotation and a translation suffice: Fooling cnns with simple transformations. *ICML*, 2019.

Daniel Kang, Yi Sun, Tom Brown, Dan Hendrycks, and Jacob Steinhardt. Transfer of adversarial robustness between perturbation types. *CoRR*, abs/1905.01034, 2019. URL `http://arxiv.org/abs/1905.01034`.

Cihang Xie, Yuxin Wu, Laurens van der Maaten, Alan L. Yuille, and Kaiming He. Feature denoising for improving adversarial robustness. *CVPR*, 2019b.

Matt Jordan, Naren Manoj, Surbhi Goel, and Alexandros G Dimakis. Quantifying perceptual distortion of adversarial examples. *arXiv preprint arXiv:1902.08265*, 2019.

Florian Tramèr and Dan Boneh. Adversarial training and robustness for multiple perturbations. *NeurIPS*, 2019. URL `http://arxiv.org/abs/1904.13000`.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. *CVPR*, 2017.

Jamie Hayes and George Danezis. Machine learning as an adversarial service: Learning black-box adversarial examples. *CoRR*, abs/1708.05207, 2017. URL `http://arxiv.org/abs/1708.05207`.

Jan Hendrik Metzen. Universality, robustness, and detectability of adversarial perturbations under adversarial training. *https://openreview.net/forum?id=SyjsLqxR-*, 2018.

Ali Shafahi, Mahyar Najibi, Zheng Xu, John P. Dickerson, Larry S. Davis, and Tom Goldstein. Universal adversarial training. *CoRR*, abs/1811.11304, 2018. URL `http://arxiv.org/abs/1811.11304`.

Chaithanya Kumar Mummadi, Thomas Brox, and Jan Hendrik Metzen. Defending against universal perturbations with shared adversarial training. *ICCV*, 2019.

Julien Pérolat, Mateusz Malinowski, Bilal Piot, and Olivier Pietquin. Playing the game of universal adversarial perturbations. *CoRR*, abs/1809.07802, 2018. URL `http://arxiv.org/abs/1809.07802`.

Jonas Rauber and Matthias Bethge. Fast differentiable clipping-aware normalization and rescaling. *arXiv preprint arXiv:2007.07677*, 2020. URL `https://github.com/jonasrauber/clipping-aware-rescaling`.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in PyTorch. In *NIPS Autodiff Workshop*, 2017.

Logan Engstrom, Andrew Ilyas, and Anish Athalye. Evaluating and understanding the robustness of adversarial logit pairing. *CoRR*, abs/1807.10272, 2018. URL `https://arxiv.org/abs/1807.10272`.

Ali Shafahi, Mahyar Najibi, Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! *arXiv preprint arXiv:1904.12843*, 2019.

Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*, 2018.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Jérôme Rony, Luiz G Hafemann, Luiz S Oliveira, Ismail Ben Ayed, Robert Sabourin, and Eric Granger. Decoupling direction and norm for efficient gradient-based l2 adversarial attacks and defenses. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4322–4330, 2019.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Eric Wong, Frank R. Schmidt, Jan Hendrik Metzen, and J. Zico Kolter. Scaling provable adversarial defenses. *CoRR*, abs/1805.12514, 2018. URL `http://arxiv.org/abs/1805.12514`.

Appendix

A Architectures of the noise generators

The architectures of the noise generators are displayed in Tables 6 6 and 7. The number of color channels is indicated by C. The noise generator displayed in Table 6 only uses kernels with a size of 1 and thus produces spatially uncorrelated noise. With the stride being 1 and no padding, the spatial dimensions are preserved in each layer. The noise generator displayed in Table 7 has one layer with 3x3 convolutions and thus produces noise samples with a correlation length of 3x3 pixels.

Layer	Shape
Conv + ReLU	$20 \times 1 \times 1$
Conv + ReLU	$20 \times 1 \times 1$
Conv + ReLU	$20 \times 1 \times 1$
Conv	C $\times 1 \times 1$

Layer	Shape
Conv + ReLU	$20 \times 1 \times 1$
Conv + ReLU	$20 \times 3 \times 3$
Conv + ReLU	$20 \times 1 \times 1$
Conv	C $\times 1 \times 1$

Table 7: Architecture of the noise generator producing locally correlated noise.

B Implementation details and hyper-parameters

Preprocessing MNIST images are preprocessed such that their pixel values lie in the range $[0, 1]$. Preprocessing for ImageNet is performed in the standard way for PyTorch ImageNet models from the model zoo by subtracting the mean $[0.485, 0.456, 0.406]$ and dividing by the standard deviation $[0.229, 0.224, 0.225]$. We add Gaussian, adversarial and Speckle noise before the preprocessing step, so the noisy images are first clipped to the range $[0, 1]$ of the raw images and then preprocessed before being fed into the model.

B.1 ImageNet experiments

For all ImageNet experiments, we used a pretrained ResNet50 architecture from `https://pytorch.org/docs/stable/torchvision/models.html`. We fine-tuned the model with SGD-M using an initial learning rate of 0.001, which corresponds to the last learning rate of the PyTorch model training, and a momentum of 0.9. After convergence, we decayed the learning rate once by a factor of 10 and continued the training. Decaying the learning rate was highly beneficial for the model performance. We tried decaying the learning rate a second time, but this did not bring any benefits in any of our experiments. For GNT, we also tried training from scratch, i.e. starting with a large learning rate of 0.1 and random weights, and trained for 120 epochs, but we got worse results compared to merely fine-tuning the model provided by torchvision. We used a batch size of 70 for all our experiments. We have also tried to use the batch sizes 50 and 100, but did not see major effects.

Gaussian noise We trained the models until convergence. The total number of training epochs varied between 30 and 90 epochs.

Speckle noise We used the Speckle noise implementation from `https://github.com/hendrycks/robustness/blob/master/ImageNet-C/create_c/make_imagenet_c.py`, line 270. The model trained with Speckle noise converged faster than with Gaussian data augmentation and therefore, we only trained the model for 10 epochs.

Adversarial Noise Training The adversarial noise generator was trained with the Adam optimizer with a learning rate of 0.0001. We have replaced the noise generator every 0.33 epochs. For $\text{ANT}^{1\text{x}1}$, we set the ϵ-sphere to control the size of the perturbation to 135.0 which on average corresponds to the ℓ_2-size of a perturbation caused by additive Gaussian noise sampled from $\mathcal{N}(0, 0.5^2 \cdot \mathbb{1})$. We have trained the classifier until convergence for 80 epochs. For $\text{ANT}^{3\text{x}3}$, we set the ϵ-sphere to 70.0 and trained the classifier for 80 epochs. We decreased the ϵ-sphere for $\text{ANT}^{3\text{x}3}$ to counteract giving the noise generator more degrees of freedom to fool the classifier to maintain a similar training losses and accuracies for $\text{ANT}^{1\text{x}1}$ and $\text{ANT}^{3\text{x}3}$.

B.2 MNIST experiments

For the MNIST experiments, we used the same model architecture as Madry et al. [2017] for our $\text{ANT}^{1\text{x}1}$ and GNT. For $\text{ANT}^{1\text{x}1}$, our learning rate for the generator was between 10^{-4} and 10^{-5}, and equal to 10^{-3} for the classifier. We used a batch size of 300. As an optimizer, we used SGD-M with a momentum of 0.9 for the classifier and Adam [Kingma and Ba, 2014] for the generator. The splitting of batches in clean, noisy and history was equivalent to the ImageNet experiments. The optimal ϵ hyper-parameter was determined with a line search similar to the optimal σ of the Gaussian noise; we found $\epsilon = 10$ to be optimal. The parameters for the Gaussian noise experiments were equivalent. Both models were trained until convergence (around 500-600 epochs). GNT and $\text{ANT}^{1\text{x}1}$ were performed on a pretrained network.

C Detailed results on the evaluation of robustness due to regular adversarial training

We find that standard adversarial training against minimal adversarial perturbations in general does not increase robustness against common corruptions. While some early results on CIFAR-10 by Ford et al. [2019] and Tiny ImageNet-C by Hendrycks and Dietterich [2019] suggest that standard adversarial training might increase robustness to common corruptions, we here observe the opposite: Adversarially trained models have lower robustness against common corruptions. An adversarially trained ResNet152 with an additional denoising layer[10] from Xie et al. [2019b] has lower accuracy across almost all corruptions except Snow and Pixelations. On some corruptions, the accuracy of the adversarially trained model decreases drastically, e.g. from 49.1% to 4.6% on Fog or 42.8% to 9.3% on Contrast. Similarly, the adversarially trained ResNet50[11] from [Shafahi et al.,

[10]Model weights from `https://github.com/facebookresearch/ImageNet-Adversarial-Training`

[11]Model weights were kindly provided by the authors.

2019] shows a substantial decrease in performance on common corruptions compared with a vanilla trained model.

An evaluation of a robustified version of AlexNet[11] [Shafahi et al., 2018] that was trained with the Universal Adversarial Training scheme on ImageNet-C shows that achieving robustness against universal adversarial perturbations does not noticeably increase robustness towards common corruptions (22.2%) compared with a vanilla trained model (21.1%).

		Noise (Compressed)			Blur (Compressed)			
Model	All	Gaussian	Shot	Impulse	Defocus	Glass	Motion	Zoom
Vanilla RN50	39.2	29.3	27.0	23.8	38.7	26.8	38.7	36.2
AT [Shafahi et al., 2019]	29.1	20.5	19.1	12.4	21.4	30.8	30.4	31.4
Vanilla RN152	45.0	35.7	34.3	29.6	45.1	32.8	48.4	40.5
AT [Xie et al., 2019b]	35.0	35.2	34.4	24.8	22.1	31.7	30.9	32.0
Vanilla AlexNet	21.1	11.4	10.6	7.7	18.0	17.4	21.4	20.2
UAT [Shafahi et al., 2018]	22.2	20.1	19.1	16.2	13.1	21.6	19.7	19.2

	Weather (Compressed)				Digital (Compressed)			
Model	Snow	Frost	Fog	Brightness	Contrast	Elastic	Pixelate	JPEG
Vanilla RN50	32.5	38.1	45.8	68.0	39.1	45.2	44.8	53.4
AT [Shafahi et al., 2019]	24.4	25.6	5.8	51.1	7.8	45.4	53.4	56.3
Vanilla RN152	38.7	43.9	49.1	71.2	42.8	51.1	50.5	60.5
AT [Xie et al., 2019b]	42.0	40.4	4.6	58.8	9.3	47.2	54.1	58.0
Vanilla AlexNet	13.3	17.3	18.1	43.5	14.7	35.4	28.2	39.4
UAT [Shafahi et al., 2018]	13.8	18.3	4.3	36.5	4.8	36.8	42.3	47.1

Table 8: Average Top-1 accuracy over 5 severities of common corruptions on ImageNet-C in percent. A high accuracy on a certain corruption type indicates high robustness of a classifier on this corruption type, so higher accuracy is better. Adversarial training (AT) decreases the accuracy on common corruptions, especially on the corruptions Fog and Contrast. Universal Adversarial Training (UAT) slightly increases the overall performance.

D Detailed ImageNet-C results

We show detailed results on individual corruptions in Table 9 in accuracy and in Table 10 in mCE for differently trained models. In Fig. 5, we show the degradation of accuracy for different severity levels. To avoid clutter, we only show results for a vanilla trained model, for the previous state of the art SIN+IN [Geirhos et al., 2019], for several Gaussian trained models and for the overall best model ANT$^{3\text{x}3}$+SIN.

The Corruption Error [Hendrycks and Dietterich, 2019] is defined as

$$\mathrm{CE}_c^f = \left(\sum_{s=1}^{5} E_{s,c}^f\right) \Big/ \left(\sum_{s=1}^{5} E_{s,c}^{\mathrm{AlexNet}}\right), \tag{5}$$

where $E_{s,c}^f$ is the Top-1 error of a classifier f for a corruption c with severity s. The mean Corruption error (mCE) is taken by averaging over all corruptions.

		Noise			Blur				Weather				Digital			
model	mean	Gauss	Shot	Impulse	Defocus	Glass	Motion	Zoom	Snow	Frost	Fog	Bright	Contrast	Elastic	Pixel	Jpeg
Vanilla RN50	39	29	27	24	39	27	39	36	33	38	46	68	39	45	45	53
Shift Inv	42	36	34	30	40	29	38	39	33	40	48	68	42	45	49	57
Patch GN	44	45	43	42	38	26	39	38	30	39	54	67	39	52	47	56
SIN+IN	45	41	40	37	43	32	45	36	41	42	47	67	43	50	56	58
AugMix	48	41	41	38	**48**	35	**54**	**49**	40	44	47	69	**51**	52	57	60
Speckle	46	55	58	49	43	32	40	36	34	41	46	68	41	47	49	58
GNT$_{\text{mult}}$	49	**67**	65	**64**	43	33	41	37	34	42	45	68	41	48	50	60
GNT$\sigma_{0.5}$	49	58	59	57	47	38	43	42	35	44	44	68	39	50	55	**62**
GNT$\sigma_{0.5}$+SIN	52	57	58	54	41	39	47	36	48	49	57	67	53	54	57	58
ANT$^{1\text{x}1}$	51	65	**66**	**64**	47	37	43	40	36	46	44	**70**	43	49	55	**62**
ANT$^{1\text{x}1}$+SIN	52	64	65	63	46	38	46	39	42	47	49	69	47	50	**57**	60
ANT$^{1\text{x}1}$ w/o EP	49	59	59	57	46	37	43	40	34	43	43	68	39	49	55	61
ANT$^{3\text{x}3}$	50	65	64	**64**	44	36	42	38	39	46	44	69	41	49	55	61
ANT$^{3\text{x}3}$+SIN	**53**	62	61	60	41	**39**	46	37	**48**	**52**	**55**	68	49	**53**	**59**	59

Table 9: Average Top-1 accuracy over 5 severities of common corruptions on ImageNet-C in percent obtained by different models; higher is better.

		Noise			Blur				Weather				Digital			
model	mCE	Gauss	Shot	Impulse	Defocus	Glass	Motion	Zoom	Snow	Frost	Fog	Bright	Contrast	Elastic	Pixel	Jpeg
Vanilla	77	80	82	83	75	89	78	80	78	75	66	57	71	85	77	77
SIN	69	66	67	68	70	82	69	80	68	71	65	58	66	78	62	70
Patch GN	71	62	63	62	75	90	78	78	81	74	57	59	71	74	74	72
Shift Inv.	73	73	74	76	74	86	78	77	77	72	63	56	68	86	71	71
AugMix	65	67	66	68	**64**	79	**59**	**64**	69	68	65	54	**57**	74	60	65
Speckle	68	51	47	55	70	83	77	80	76	71	66	57	70	82	71	69
GNT$_{\text{mult}}$	65	**37**	39	**39**	69	81	76	79	76	70	67	56	69	81	69	66
GNT$\sigma_{0.5}$	64	46	46	47	65	75	72	74	75	68	69	57	71	78	63	63
ANT$^{1\text{x}1}$	62	39	**38**	**39**	65	77	72	75	74	66	68	**53**	67	78	62	**62**
ANT$^{1\text{x}1}$+SIN	**61**	40	39	40	65	76	69	76	67	64	62	55	63	77	59	66
ANT$^{1\text{x}1}$ w/o EP	65	46	46	47	66	76	73	75	76	69	70	57	72	79	63	64
ANT$^{3\text{x}3}$	63	39	40	**39**	68	78	73	77	71	66	68	55	69	79	63	64
ANT$^{3\text{x}3}$+SIN	**61**	43	44	43	71	**74**	69	79	**60**	**58**	**55**	56	59	**73**	**57**	67

Table 10: Average mean Corruption Error (mCE) obtained by different models on common corruptions from ImageNet-C; lower is better.

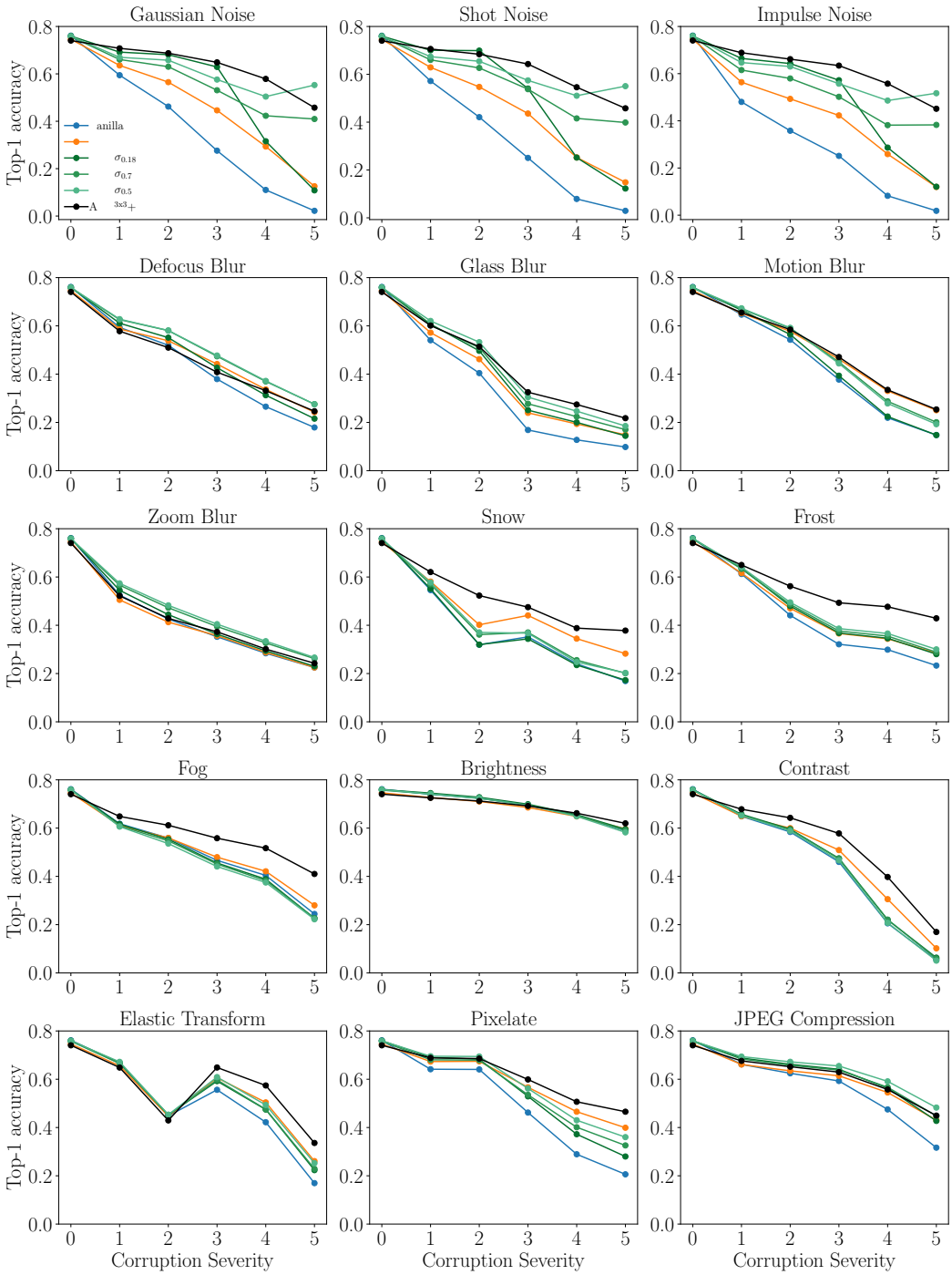


Figure 5. Top-1 accuracy for each corruption type and severity on ImageNet-C.

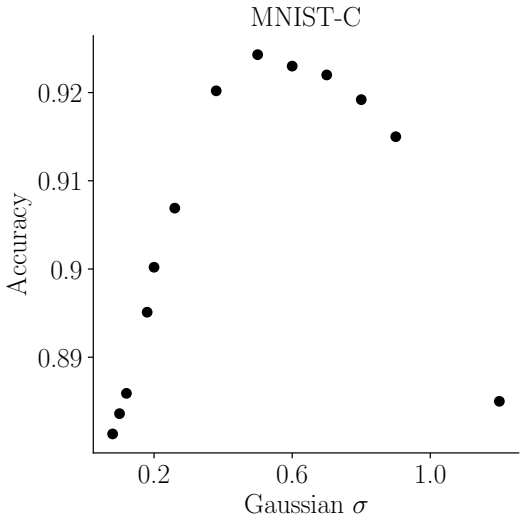


Figure 6. Average accuracy on MNIST-C over all severties and corruptions for different values of sigma σ of the Gaussian noise training (GNT) during training. Each point corresponds to one converged training.

E MNIST-C results

Similar to the ImageNet-C experiments, we are interested how vanilla, adversarially and noise trained models perform on MNIST-C.

The adversarially robust MNIST model by Wong et al. [2018] was trained with a robust loss function and is among the state of the art in certified adversarial robustness. The other baseline models were trained with Adversarial Training in ℓ_2 (DDN) by Rony et al. [2019] and ℓ_∞ (PGD) by Madry et al. [2017]. Our GNT and ANT$^{1\text{x}1}$ trained versions are trained as described in the main paper and Appendix B.2. The results are shown in Table 11. Similar to ImageNet-C, the models trained with GNT and ANT$^{1\text{x}1}$ are significantly better than our vanilla trained baseline. Also, regular adversarial training has severe drops and does not lead to significant robustness improvements.

As for ImageNet and GNT, we have treated σ as a hyper-parameter. The accuracy on MNIST-C for different values of σ is displayed in Fig. 6 and has a maximum around $\sigma = 0.5$ like for ImageNet.

model	clean acc	mean	Shot	Impulse	Glass Blur	Motion Blur	Shear	Scale	Rotate	Brightness	Translate	Stripe	Fog	Splatter	Dotted Line	Zig Zag	Canny Edges
Vanilla	99.1	86.9	98	96	96	94	98	95	92	88	57	88	50	97	96	86	72
[Madry et al., 2017]	98.5	75.6	98	55	94	94	97	88	92	27	53	40	63	96	78	74	84
Vanilla	98.8	74.3	98	91	96	88	95	80	89	34	45	41	23	96	96	80	63
[Wong et al., 2018]	98.2	68.6	97	65	93	93	94	87	89	11	40	20	25	96	89	61	68
Vanilla	99.5	89.8	98	96	95	97	98	96	94	95	61	89	79	98	98	90	63
DDN Tr [Rony et al., 2019]	99.0	87.0	99	97	96	94	98	91	93	72	55	92	64	99	98	91	66
Vanilla	99.1	86.9	98	96	96	94	98	95	92	88	57	88	50	97	96	86	72
GNT$\sigma_{0.5}$	99.3	92.4	99	99	98	97	98	95	93	98	56	91	91	99	99	96	78
ANT$^{1\text{x}1}$	99.4	92.4	99	99	98	97	98	95	93	98	55	89	91	99	99	96	80

Table 11: Accuracy in percent for the MNIST-C dataset for adversarially robust ([Wong et al., 2018], [Madry et al., 2017], DDN [Rony et al., 2019]) and our noise trained models (GNT and ANT$^{1\text{x}1}$). Vanilla always denotes the same network architecture as its adversarially or noise trained counterpart but with standard training. Note that we used the same network architecture as Madry et al. [2017].

F Evaluation of adversarial robustness of models trained via GNT and ANT^{1x1}

ImageNet To evaluate adversarial robustness on ImageNet, we used PGD [Madry et al., 2017] and DDN [Rony et al., 2019]. For the ℓ_∞ PGD attack, we allowed for 200 iterations with a step size of 0.0001 and a maximum sphere size of 0.001. For the DDN ℓ_2 attack, we also allowed for 200 iterations, set the sphere adjustment parameter γ to 0.02 and the maximum epsilon to 0.125. We note that for both attacks increasing the number of iterations from 100 to 200 did not make a significant difference in robustness of our tested models. The results on adversarial robustness on ImageNet can be found in the main paper in Table 5.

MNIST To evaluate adversarial robustness on MNIST, we also used PGD [Madry et al., 2017] and DDN [Rony et al., 2019]. For the ℓ_∞ PGD attack, we allowed for 100 iterations with a step size of 0.01 and a maximum sphere size of 0.1. For the DDN ℓ_2 attack, we also allowed for 100 iterations, set the sphere adjustment parameter γ to 0.05 and the maximum epsilon to 1.5. All models have the same architecture as Madry et al. [2017]. The results on adversarial robustness on MNIST can be found in Table 12.

model	clean acc. [%]	ℓ_2 acc. [%]	ℓ_∞ acc. [%]
Vanilla	99.1	73.2	55.8
GNT$_{\sigma_{0.5}}$	99.3	89.2	73.6
ANT^{1x1}	99.4	90.4	76.3

Table 12: Adversarial robustness on MNIST on ℓ_2 ($\epsilon = 1.5$) and ℓ_∞ ($\epsilon = 0.1$) compared to a Vanilla CNN.

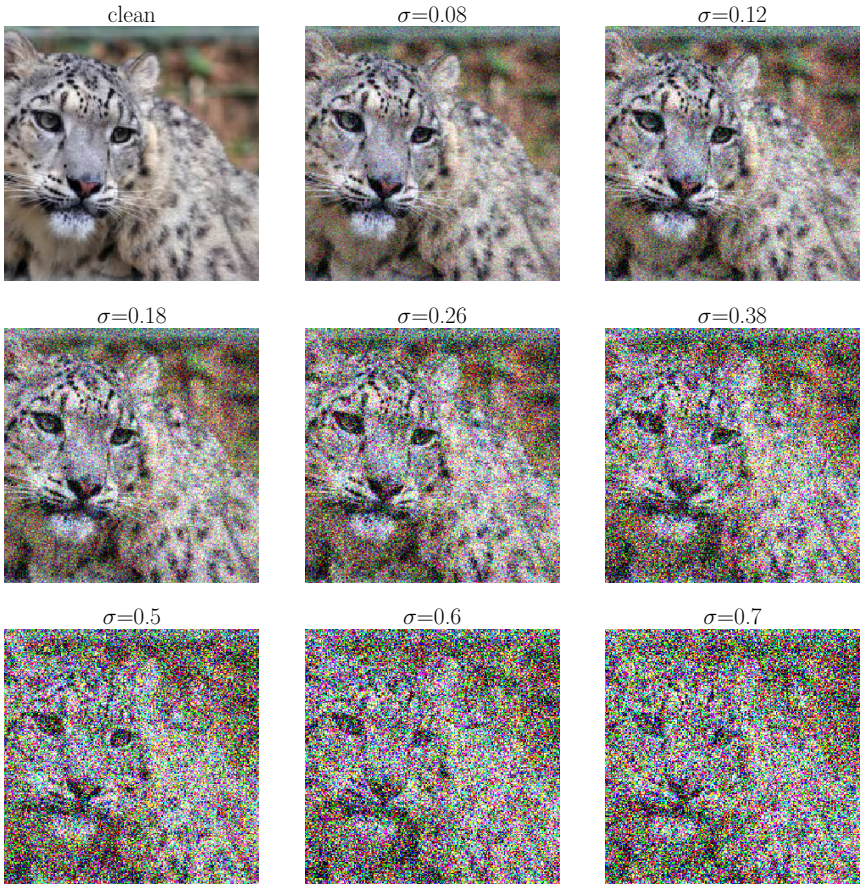


Figure 7. Example images with different σ-levels of additive Gaussian noise on ImageNet.

G Example images for additive Gaussian noise

Example images with additive Gaussian noise of varying standard deviation σ are displayed in Fig. 7. The considered σ-levels correspond to those studied in section 4.2. in the main paper.

H Comparison to Ford et al.

Ford et al. trained an InceptionV3 model from scratch both on clean data from the ImageNet dataset and on data augmented with Gaussian noise [Ford et al., 2019]. Since we use a very similar approach, we compare our approach to theirs directly. The results for comparison on ImageNet both for the vanilla and the Gaussian noise trained model can be found in Table 13. Since we use a pretrained model provided by PyTorch and fine-tune it instead of training a new one, the performance of our vanilla trained model differs from the performance of their vanilla trained model, both on clean data and on ImageNet-C. The accuracy on clean data is displayed in Table 14. Another difference between our training and theirs is that we split every batch evenly in clean and data augmented by Gaussian noise with one standard deviation whereas they sample σ uniformly between 0 and one specific value. With our training scheme, we were able to outperform their model significantly on all corruptions except for Elastic, Fog and Brightness.

		Noise (Compressed)			Blur (Compressed)			
model	All	Gaussian	Shot	Impulse	Defocus	Glass	Motion	Zoom
Vanilla InceptionV3 [Ford et al., 2019]	38.8	36.6	34.3	34.7	31.1	19.3	35.3	30.1
Gaussian ($\sigma = 0.4$) [Ford et al., 2019]	42.7	40.3	38.8	37.7	32.9	29.8	35.3	33.1
Vanilla InceptionV3 [ours]	41.6	42.0	40.3	38.5	33.5	27.1	36.1	28.8
GNT$\sigma_{0.4}$ [ours]	49.5	60.8	59.6	59.4	43.8	37.0	42.8	38.4
GNT$\sigma_{0.5}$ [ours]	**50.2**	**61.6**	**60.9**	**60.8**	**44.6**	**37.3**	**44.0**	**39.3**

	Weather (Compressed)				Digital (Compressed)			
model	Snow	Frost	Fog	Brightness	Contrast	Elastic	Pixelate	JPEG
Vanilla InceptionV3 [Ford et al., 2019]	33.1	34.0	52.4	66.0	35.9	47.8	38.2	50.0
Gaussian ($\sigma = 0.4$) [Ford et al., 2019]	36.6	43.5	**52.3**	**67.1**	35.8	**52.2**	47.0	55.5
Vanilla InceptionV3 [ours]	33.5	39.6	42.2	64.2	41.0	43.5	57.4	56.9
GNT$\sigma_{0.4}$ [ours]	35.6	43.7	43.3	64.8	43.0	49.0	59.3	61.7
GNT$\sigma_{0.5}$ [ours]	**37.1**	**44.2**	43.6	64.6	**43.3**	49.4	**59.6**	**61.9**

Table 13: ImageNet-C accuracy for InceptionV3.

model	clean accuracy [%]
Vanilla InceptionV3 [Ford et al., 2019]	75.9
Gaussian ($\sigma = 0.4$) [Ford et al., 2019]	74.2
Vanilla InceptionV3 [ours]	77.2
GNT$\sigma_{0.4}$ [ours]	78.1
GNT$\sigma_{0.5}$ [ours]	77.9

Table 14: Accuracy on clean data for differently trained models.

I Visualization of images with different perturbation sizes

In the main paper, we measure model robustness by calculating the median perturbation size ϵ^* and report the results in Table 3. To provide a better intuition for the noise level in an image for a particular ϵ^*, we display example images in Fig. 8.

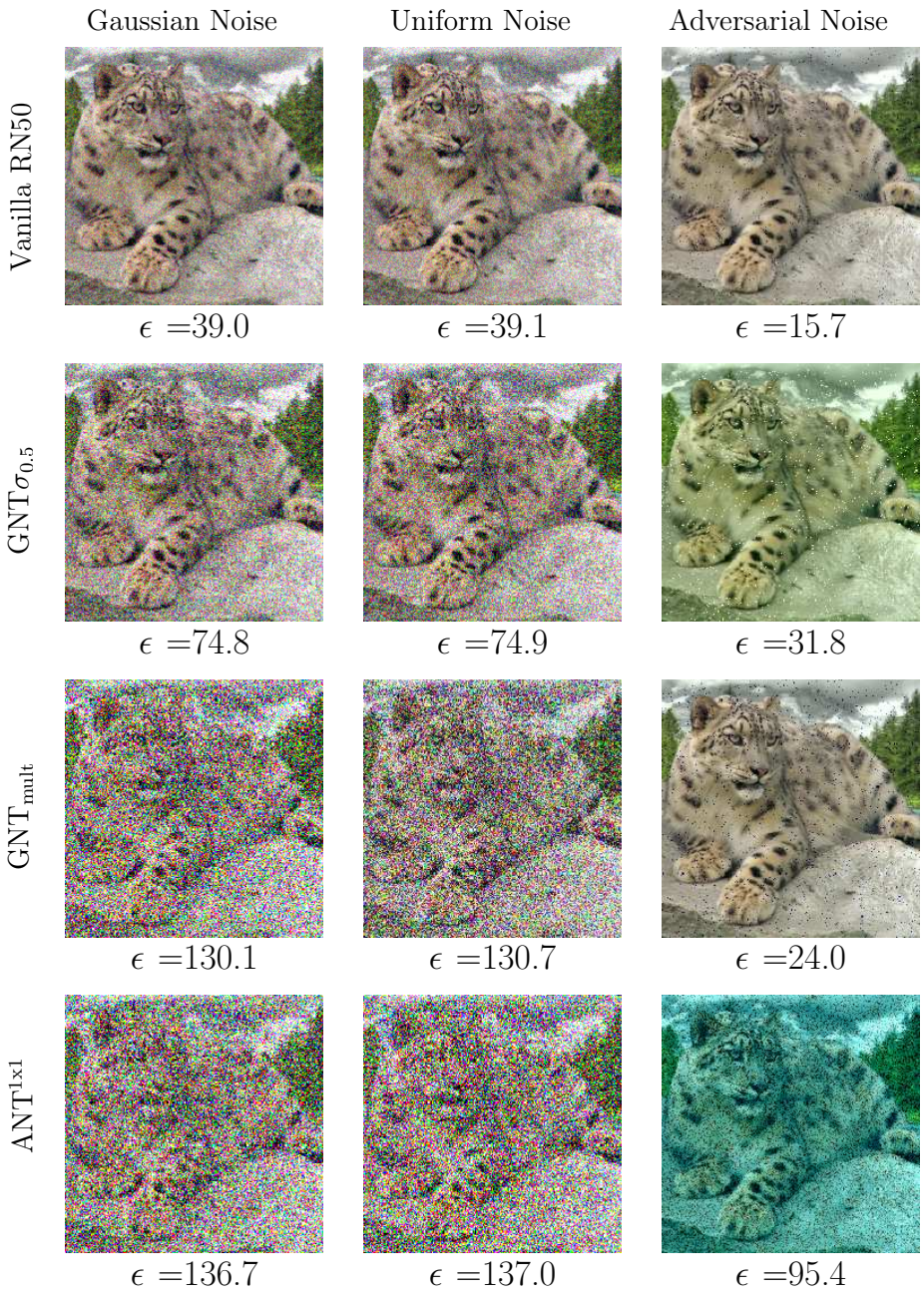


Figure 8. Example images for the different perturbation sizes ϵ^* and different noise types on ImageNet corresponding to the ϵ^* values in Table 3 in the main paper.

Figure 9. Example images for the JPEG compression from ImageNet-C and the `PIL.ImageOps.Posterize` operation.

	IN	IN-C		IN-C w/o noises	
model	clean acc.	Top-1	Top-5	Top-1	Top-5
Vanilla RN50	76.1	39.2	59.3	42.3	63.2
ANT^{1x1} RN50	76.0	(51.1)	(72.2)	47.7	68.8
Vanilla DN121	74.4	42.1	63.4	44.0	65.5
ANT^{1x1} DN121	74.3	50.3	71.6	46.8	68.3

Table 15: Average accuracy on clean data, average Top-1 and Top-5 accuracies on full ImageNet-C and ImageNet-C without the noise category (higher is better); all values in percent. We compare the results obtained by ANT^{1x1} for a ResNet50 (RN50) architecture to a DenseNet121 (DN121) architecture.

J Visualization of Posterize vs JPEG

AugMix [Hendrycks et al., 2020] uses Posterize as one of their operations for data augmentation during training. In Fig. 9, we show the visual similarity between the Posterize operation and the JPEG corruption from ImageNet-C.

K Additional results

K.1 Adversarial Noise Training with a DenseNet121 architecture

To test in how far our results generalize to other backbones, we have trained a DenseNet121 model with ANT^{1x1}. The DenseNet121 model was finetuned from the checkpoint provided by torchvision. A DenseNet121 has $7.97886 \cdot 10^6$ trainable parameters whereas a ResNet50 has $2.5557 \cdot 10^7$. Our results and a comparison to ANT^{1x1} with a ResNet50 model is shown in Table 15: ANT^{1x1} increases robustness on full ImageNet-C and ImageNet-C without noises for the DenseNet121 model, showing that adversarial noise training generalizes to other backbones.

model	Number of parameters	IN clean acc.	IN-C Top-1	IN-C w/o noises Top-1
Vanilla RN50	-	76.1	39.2	42.3
ANT^{1x1} RN50 NG_1	12	75.1	(49.5)	46.6
ANT^{1x1} RN50 NG_2	143	75.5	(50.8)	47.2
ANT^{1x1} RN50 NG_3	563	75.3	(50.7)	47.2
ANT^{1x1} RN50 NG_4	983	76.0	(51.1)	47.7
ANT^{1x1} RN50 NG_5	1403	74.0	(50.7)	47.0

Table 16: Number of trainable parameters of different noise generators, average accuracy on clean data, ImageNet-C and ImageNet-C without the noise category (higher is better); all values in percent. We compare the results obtained by ANT^{1x1} with noise generators of different depth. Note that a depth of 4 layers was used in all experiments in this paper apart from this ablation study.

K.2 Results for different parameter counts of the noise generator

Here, we study the effect of different parameter counts of the adversarial noise generator on ANT^{1x1}. We provide the results in Table 16. We indicate the depth of the noise generator with a subscript. All experiments in this paper apart from this ablation study were performed with a default depth of 4 layers. We observe that while depth is a tunable hyper-parameter, the performances of ANT^{1x1} with the studied noise generators do not differ by a lot. Only the most shallow noise generator with a depth of one layer and only 12 trainable parameters results in a roughly 1% lower accuracy than its deeper counterparts. We note that a $GNT\sigma_{0.5}$ model has an accuracy of 49.4% on full ImageNet-C and an accuracy of 47.1% on ImageNet-C without noises which roughly corresponds to the respective accuracies of ANT^{1x1} with the most shallow noise generator.

arXiv:1907.07484v2 [cs.CV] 31 Mar 2020

Benchmarking Robustness in Object Detection: Autonomous Driving when Winter is Coming

Claudio Michaelis* **Benjamin Mitzkus*** **Robert Geirhos*** **Evgenia Rusak***

Oliver Bringmann† **Alexander S. Ecker**† **Matthias Bethge**†

Wieland Brendel†
University of Tübingen
first.last@uni-tuebingen.de

Abstract

The ability to detect objects regardless of image distortions or weather conditions is crucial for real-world applications of deep learning like autonomous driving. We here provide an easy-to-use benchmark to assess how object detection models perform when image quality degrades. The three resulting benchmark datasets, termed Pascal-C, Coco-C and Cityscapes-C, contain a large variety of image corruptions. We show that a range of standard object detection models suffer a severe performance loss on corrupted images (down to 30–60% of the original performance). However, a simple data augmentation trick—stylizing the training images—leads to a substantial increase in robustness across corruption type, severity and dataset. We envision our comprehensive benchmark to track future progress towards building robust object detection models. Benchmark, code and data are publicly available.

Figure 1: Mistaking a dragon for a bird (left) may be dangerous but missing it altogether because of snow (right) means playing with fire. Sadly, this is exactly the fate that an autonomous agent relying on a state-of-the-art object detection system would suffer. Predictions generated using Faster R-CNN; best viewed on screen.

1 Introduction

A day in the near future: Autonomous vehicles are swarming the streets all over the world, tirelessly collecting data. But on this cold November afternoon traffic comes to an abrupt halt as it suddenly begins to snow: winter is coming. Huge snowflakes are falling from the sky and the cameras of autonomous vehicles are no longer able to make sense of their surroundings, triggering immediate emergency brakes. A day later, an investigation of this traffic disaster reveals that the unexpectedly large size of

Machine Learning for Autonomous Driving Workshop at the 33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

Figure 2: Expect the unexpected: To ensure safety, an autonomous vehicle must be able to recognize objects even in challenging outdoor conditions such as fog, rain, snow and at night.[1]

> *the snowflakes was the cause of the chaos: While state-of-the-art vision systems had been trained on a variety of common weather types, their training data contained hardly any snowflakes of this size...*

This fictional example highlights the problems that arise when Convolutional Neural Networks (CNNs) encounter settings that were not explicitly part of their training regime. For example, state-of-the-art object detection algorithms such as Faster R-CNN [Ren et al., 2015] fail to recognize objects when snow is added to an image (as shown in Figure 1), even though the objects are still clearly visible to a human eye. At the same time, augmenting the training data with several types of distortions is not a sufficient solution to achieve general robustness against previously unknown corruptions: It has recently been demonstrated that CNNs generalize poorly to novel distortion types, despite being trained on a variety of other distortions [Geirhos et al., 2018].

On a more general level, CNNs often fail to generalize outside of the training domain or training data distribution. Examples include the failure to generalize to images with uncommon poses of objects [Alcorn et al., 2019] or to cope with small distributional changes [e.g. Zech et al., 2018, Touvron et al., 2019]. One of the most extreme cases are adversarial examples [Szegedy et al., 2013]: images with a domain shift so small that it is imperceptible for humans yet sufficient to fool a DNN. We here focus on the less extreme but far more common problem of perceptible image distortions like blurry images, noise or natural distortions like snow.

As an example, autonomous vehicles need to be able to cope with wildly varying outdoor conditions such as fog, frost, snow, sand storms, or falling leaves, just to name a few (as visualized in Figure 2). One of the major reasons why autonomous cars have not yet gone mainstream is the inability of their recognition models to function well in adverse weather conditions [Dai and Van Gool, 2018]. Getting data for unusual weather conditions is hard and while many common environmental conditions can (and have been) modelled, including fog [Sakaridis et al., 2018a], rain [Hospach et al., 2016], snow [Bernuth et al., 2019] and daytime to nighttime transitions [Dai and Van Gool, 2018], it is impossible to foresee all potential conditions that might occur "in the wild".

If we could build models that are robust to every possible image corruption, it is to be expected that weather changes would not be an issue. However, in order to assess the robustness of models one first needs to define a measure. While testing models on the set of all possible corruption types is impossible. We therefore propose to evaluate models on a diverse range of corruption types that were not part of the training data and demonstrate that this is a useful approximation for predicting performance under natural distortions like rain, snow, fog or the transition between day and night.

More specifically we propose three easy-to-use benchmark datasets termed PASCAL-C, COCO-C and Cityscapes-C to assess distortion robustness in object detection. Each dataset contains versions of the original object detection dataset which are corrupted with 15 distortions, each spanning five levels of severity. This approach follows Hendrycks and Dietterich [2019], who introduced corrupted versions of commonly used *classification* datasets (ImageNet-C, CIFAR10-C) as standardized benchmarks. After evaluating standard object detection algorithms on these benchmark datasets, we show how a simple data augmentation technique—stylizing the training images—can strongly improve robustness across corruption type, severity and dataset.

1.1 Contributions

Our contributions can be summarized as follows:

[1] Outdoor hazards have been directly linked to increased mortality rates [Lystad and Brown, 2018].

1. We demonstrate that a broad range of object detection and instance segmentation models suffer severe performance impairments on corrupted images.
2. To quantify this behaviour and to enable tracking future progress, we propose the `Robust Detection Benchmark`, consisting of three benchmark datasets termed PASCAL-C, COCO-C & Cityscapes-C.
3. We demonstrate that improved performance on this benchmark of synthetic corruptions corresponds to increased robustness towards real-world "natural" distortions like rain, snow and fog.
4. We use the benchmark to show that corruption robustness scales with performance on clean data and that a simple data augmentation technique—stylizing the training data—leads to large robustness improvements for all evaluated corruptions without any additional labelling costs or architectural changes.
5. We make our benchmark, corruption and stylization code openly available in an easy-to-use fashion:
   - Benchmark, [2] data and data analysis are available at `https://github.com/bethgelab/robust-detection-benchmark`.
   - Our pip installable image corruption library is available at `https://github.com/bethgelab/imagecorruptions`.
   - Code to stylize arbitrary datasets is provided at `https://github.com/bethgelab/stylize-datasets`.

1.2 Related Work

Benchmarking corruption robustness Several studies investigate the vulnerability of CNNs to common corruptions. Dodge and Karam [2016] measure the performance of four state-of-the-art image recognition models on out-of-distribution data and show that CNNs are in particular vulnerable to blur and Gaussian noise. Geirhos et al. [2018] show that CNN performance drops much faster than human performance for the task of recognizing corrupted images when the perturbation level increases across a broad range of corruption types. Azulay and Weiss [2018] investigate the lack of invariance of several state-of-the-art CNNs to small translations. A benchmark to evaluate the robustness of recognition models against common corruptions was recently introduced by Hendrycks and Dietterich [2019].

Improving corruption robustness One way to restore the performance drop on corrupted data is to preprocess the data in order to remove the corruption. Mukherjee et al. [2018] propose a DNN-based approach to restore image quality of rainy and foggy images. Bahnsen and Moeslund [2018] and Bahnsen et al. [2019] propose algorithms to remove rain from images as a preprocessing step and report a subsequent increase in recognition rate. A challenge for these approaches is that noise removal is currently specific to a certain distortion type and thus does not generalize to other types of distortions. Another line of work seeks to enhance the classifier performance by the means of data augmentation, i.e. by directly including corrupted data into the training. Vasiljevic et al. [2016] study the vulnerability of a classifier to blurred images and enhance the performance on blurred images by fine-tuning on them. Geirhos et al. [2018] examine the generalization between different corruption types and find that fine-tuning on one corruption type does not enhance performance on other corruption types. In a different study, Geirhos et al. [2019] train a recognition model on a stylized version of the ImageNet dataset [Russakovsky et al., 2015], reporting increased general robustness against different corruptions as a result of a stronger bias towards ignoring textures and focusing on object shape. Hendrycks and Dietterich [2019] report several methods leading to enhanced performance on their corruption benchmark: Histogram Equalization, Multiscale Networks, Adversarial Logit Pairing, Feature Aggregating and Larger Networks.

Evaluating robustness to environmental changes in autonomous driving In recent years, weather conditions turned out to be a central limitation for state-of-the art autonomous driving systems [Sakaridis et al., 2018a, Volk et al., 2019, Dai and Van Gool, 2018, Chen et al., 2018,

[2] Our evaluation code to assess performance under corruption has been integrated into one of the most widely used detection toolboxes. The code can be found here: `https://github.com/bethgelab/mmdetection`

Lee et al., 2018]. While many specific approaches like modelling weather conditions [Sakaridis et al., 2018a,b, Volk et al., 2019, Bernuth et al., 2019, Hospach et al., 2016, Bernuth et al., 2018] or collecting real [Wen et al., 2015, Yu et al., 2018, Che et al., 2019, Caesar et al., 2019] and artificial [Gaidon et al., 2016, Ros et al., 2016, Richter et al., 2017, Johnson-Roberson et al., 2017] datasets with varying weather conditions, no general solution towards the problem has yet emerged. Radecki et al. [2016] experimentally test the performance of various sensors and object recognition and classification models in adverse weather and lighting conditions. Bernuth et al. [2018] report a drop in the performance of a Recurrent Rolling Convolution network trained on the KITTI dataset when the camera images are modified by simulated raindrops on the windshield. Pei et al. [2017] introduce VeriVis, a framework to evaluate the security and robustness of different object recognition models using real-world image corruptions such as brightness, contrast, rotations, smoothing, blurring and others. Machiraju and Channappayya [2018] propose a metric to evaluate the degradation of object detection performance of an autonomous vehicle in several adverse weather conditions evaluated on the Virtual KITTI dataset. Building upon Hospach et al. [2016], Volk et al. [2019] study the fragility of an object detection model against rainy images, identify corner cases where the model fails and include images with synthetic rain variations into the training set. They report enhanced performance on real rain images. Bernuth et al. [2019] model photo-realistic snow and fog conditions to augment real and virtual video streams. They report a significant performance drop of an object detection model when evaluated on corrupted data.

2 Methods

2.1 Robust Detection Benchmark

We introduce the `Robust Detection Benchmark` inspired by the ImageNet-C benchmark for object classification [Hendrycks and Dietterich, 2019] to assess object detection robustness on corrupted images.

Corruption types Following Hendrycks and Dietterich [2019], we provide 15 corruptions on five severity levels each (visualized in Figure 3) to assess the effect of a broad range of different corruption types on object detection models.[3] The corruptions are sorted into four groups: noise, blur, digital and weather groups (as defined by Hendrycks and Dietterich [2019]). It is important to note that the corruption types are *not* meant to be used as a training data augmentation toolbox, but rather to measure a model's robustness against *previously unseen* corruptions. Thus, training should be done without using any of the provided corruptions. For model validation, four separate corruptions are provided (Speckle Noise, Gaussian Blur, Spatter, Saturate). The 15 corruptions described above should only be used to test the final model performance.

Benchmark datasets The `Robust Detection Benchmark` consists of three benchmark datasets: PASCAL-C, COCO-C and Cityscapes-C. Among the vast number of available object detection datasets [Everingham et al., 2010, Geiger et al., 2012, Lin et al., 2014, Cordts et al., 2016, Zhou et al., 2017, Neuhold et al., 2017, Krasin et al., 2017], we chose to use PASCAL VOC [Everingham et al., 2010], MS COCO [Lin et al., 2014] and Cityscapes [Cordts et al., 2016] as they are the most commonly used datasets for general object detection (PASCAL & COCO) and street scenes (Cityscapes). We follow common conventions to select the tests splits: VOC2007 test set for PASCAL-C, the COCO 2017 validation set for COCO-C and the Cityscapes validation set for Cityscapes-C.

Metrics Since performance measures differ between the original datasets, the dataset-specific performance (P) measures are adopted as defined below:

$$\mathrm{P} := \begin{cases} \mathrm{AP}^{50}(\%) & \text{PASCAL VOC} \\ \mathrm{AP}(\%) & \text{MS COCO } \& \text{ Cityscapes} \end{cases}$$

where AP^{50} stands for the PASCAL 'Average Precision' metric at 50% Intersection over Union (IoU) and AP stands for the COCO 'Average Precision' metric which averages over IoUs between 50% and

[3] These corruption types were introduced by Hendrycks and Dietterich [2019] and modified by us to work with images of arbitrary dimensions. Our generalized corruptions can be found at `https://github.com/bethgelab/imagecorruptions` and installed via `pip3 install imagecorruptions`.

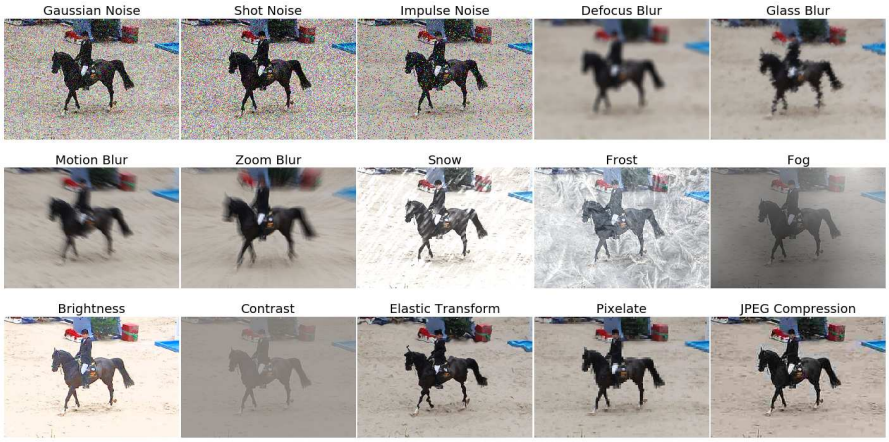


Figure 3: 15 corruption types from Hendrycks and Dietterich [2019], adapted to corrupt arbitrary images (example: randomly selected PASCAL VOC image, center crop, severity 3). Best viewed on screen.

95%. On the corrupted data, the benchmark performance is measured in terms of mean performance under corruption (mPC):

$$\mathrm{mPC} = \frac{1}{\mathrm{N}_c} \sum_{c=1}^{\mathrm{N}_c} \frac{1}{\mathrm{N}_s} \sum_{s=1}^{\mathrm{N}_s} \mathrm{P}_{c,s} \tag{1}$$

Here, $\mathrm{P}_{c,s}$ is the dataset-specific performance measure evaluated on test data corrupted with corruption c under severity level s while $\mathrm{N}_c = 15$ and $\mathrm{N}_s = 5$ indicate the number of corruptions and severity levels, respectively. In order to measure relative performance degradation under corruption, the relative performance under corruption (rPC) is introduced as defined below:

$$\mathrm{rPC} = \frac{\mathrm{mPC}}{\mathrm{P}_{\mathrm{clean}}} \tag{2}$$

rPC measures the relative degradation of performance on corrupted data compared to clean data.

Submissions Submissions to the benchmark should be handed in as a simple pull request to the `Robust Detection Benchmark`[4] and need to include all three performance measures: clean performance ($\mathrm{P}_{\mathrm{clean}}$), mean performance under corruption (mPC) and relative performance under corruption (rPC). While mPC is the metric used to rank models on the `Robust Detection Benchmark`, the other measures provide additional insights, as they disentangle gains from higher clean performance (as measured by $\mathrm{P}_{\mathrm{clean}}$) and gains from better generalization performance to corrupted data (as measured by rPC).

Baseline models We provide baseline results for a set of common object detection models including Faster R-CNN [Ren et al., 2015], Mask R-CNN [He et al., 2017], Cascade R-CNN [Cai and Vasconcelos, 2018], Cascade Mask R-CNN [Chen et al., 2019a], RetinaNet [Lin et al., 2017a] and Hybrid Task Cascade [Chen et al., 2019a]. We use a ResNet50 [He et al., 2016] with Feature Pyramid Networks [Lin et al., 2017b] as backbone for all models except for Faster R-CNN where we additionally test ResNet101 [He et al., 2016], ResNeXt101-32x4d [Xie et al., 2017] and ResNeXt-64x4d [Xie et al., 2017] backbones. We additionally provide results for Faster R-CNN and Mask R-CNN models with deformable convolutions [Dai et al., 2017, Zhu et al., 2018] in Appendix D. Models were evaluated using the `mmdetection toolbox` [Chen et al., 2019b]; all models were trained and tested with standard hyperparameters. The details can be found in Appendix A.

2.2 Style transfer as data augmentation

For image classification, style transfer [Gatys et al., 2016]—the method of combining the content of an image with the style of another image—has been shown to strongly improve corruption robustness

[4] `https://github.com/bethgelab/robust-detection-benchmark`

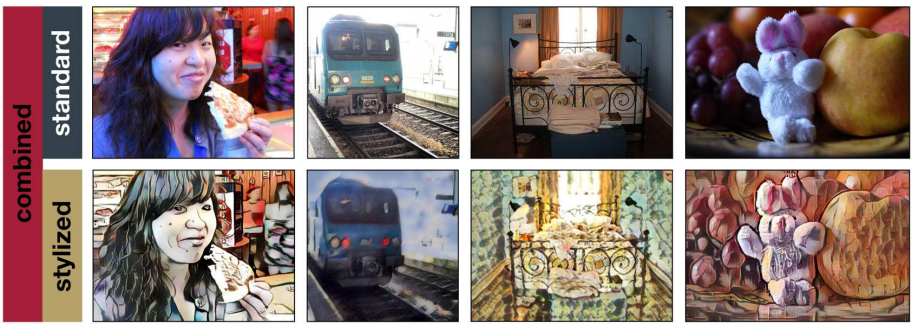


Figure 4: Training data visualization for COCO and Stylized-COCO. The three different training settings are: standard data (top row), stylized data (bottom row) and the concatenation of both (termed 'combined' in plots).

[Geirhos et al., 2019]. We here transfer this method to object detection datasets testing two settings: (1) Replacing each training image with a stylized version and (2) adding a stylized version of each image to the existing dataset. We apply the fast style transfer method AdaIN [Huang and Belongie, 2017] with hyperparameter $\alpha = 1$ to the training data, replacing the original texture with the randomly chosen texture information of Kaggle's `Painter by Numbers`[5] dataset. Examples for the stylization of COCO images are given in Figure 4. We provide ready-to-use code for the stylization of arbitrary datasets at `https://github.com/bethgelab/stylize-datasets`.

2.3 Natural Distortions

Foggy Cityscapes Foggy Cityscapes Sakaridis et al. [2018a] is a version of Cityscapes with synthetic fog in three severity levels (given byt he attenuation coefficient $\beta = 0.005m^{-1}$, $0.01m^{-1}$ and $0.02m^{-1}$), that was carefully designed to look as realistic as possible. We use Fogy Cityscapes only at test time, testing the same models as used for our experiments with the original Cityscapes dataset and report results in the same AP metric.

BDD100k BDD100k Yu et al. [2018] is a driving dataset consisting of 100 thousand videos of driving scenes recorded in varying conditions including weather changes and different times of the day[6]. We use these annotations to perform experiments, on different weather conditions ("clear", "rainy" and "snowy") and on the transition from day to night. Training is performed on what we would consider "clean" data - clear for weather and daytime for time - and evaluation is performed on all three splits. We use Faster R-CNN with the same hyper-parameters as in our experiments on COCO. Details of the dataset preparation can be found in Appendix C.

3 Results

3.1 Image corruptions reduce model performance

In order to assess the effect of image corruptions, we evaluated a set of common object detection models on the three benchmark datasets defined in Section 2. Performance is heavily degraded on corrupted images (compare Table 1). While Faster R-CNN can retain roughly 60% relative performance (rPC) on the rather simple images in PASCAL VOC, the same model suffers a dramatic reduction to 33% rPC on the Cityscapes dataset, which contains many small objects. With some variations, this effect is present in all tested models and also holds for instance segmentation tasks (for instance segmentation results, please see Appendix D).

[5] `https://www.kaggle.com/c/painter-by-numbers/`

[6] The frame at the 10th second of each video is annotated with additional information including bounding boxes which we use for our experiments

PASCAL VOC

model	backbone	clean P [AP^{50}]	corrupted mPC [AP^{50}]	relative rPC [%]
Faster	r50	80.5	48.6	60.4

MS COCO

model	backbone	clean P [AP]	corrupted mPC [AP]	relative rPC [%]
Faster	r50	36.3	18.2	50.2
Faster	r101	38.5	20.9	54.2
Faster	x101-32x4d	40.1	22.3	55.5
Faster	x101-64x4d	41.3	23.4	56.6
Mask	r50	37.3	18.7	50.1
Cascade	r50	40.4	20.1	49.7
Cascade Mask	r50	41.2	20.7	50.2
RetinaNet	r50	35.6	17.8	50.1
HTC	x101-64x4d	50.6	32.7	64.7

Cityscapes

model	backbone	clean P [AP]	corrupted mPC [AP]	relative rPC [%]
Faster	r50	36.4	12.2	33.4
Mask	r50	37.5	11.7	31.1

Table 1: Object detection performance of various models. Backbones indicated with r are ResNet and x ResNeXt. All model names except for RetinaNet and HTC indicate the corresponding model from the R-CNN family. All COCO models were downloaded from the `mmdetection` modelzoo. For all reported quantities: higher is better; square brackets denote metric.

3.2 Robustness increases with backbone capacity

We test variants of Faster R-CNN with different backbones (top of Table 1) and different head architectures (bottom of Table 1) on COCO. For the models with different backbones, we find that all image corruptions—except for the blur types—induce a fixed penalty to model performance, independent of the baseline performance on clean data: $\Delta\,\mathrm{mPC} \approx \Delta\,\mathrm{P}$ (compare Table 1 and Appendix Figure 10). Therefore, models with more powerful backbones show a relative performance improvement under corruption.[7] In comparison, Mask R-CNN, Cascade R-CNN and Cascade Mask R-CNN which draw their performance increase from more sophisticated head architectures all have roughly the same rPC of $\approx 50\%$. The current state-of-the-art model Hybrid Task Cascade [Chen et al., 2019a] is in so far an exception as it employs a combination of a stronger backbone, improved head architecture and additional training data to not only outperform the strongest baseline model by 9% AP on clean data but distances itself on corrupted data by a similar margin, achieving a leading relative performance under corruption (rPC) of 64.7%. These results indicate that robustness in the tested regime can be improved primarily through a better image encoding, and better head architectures cannot extract more information if the primary encoding is already sufficiently impaired.

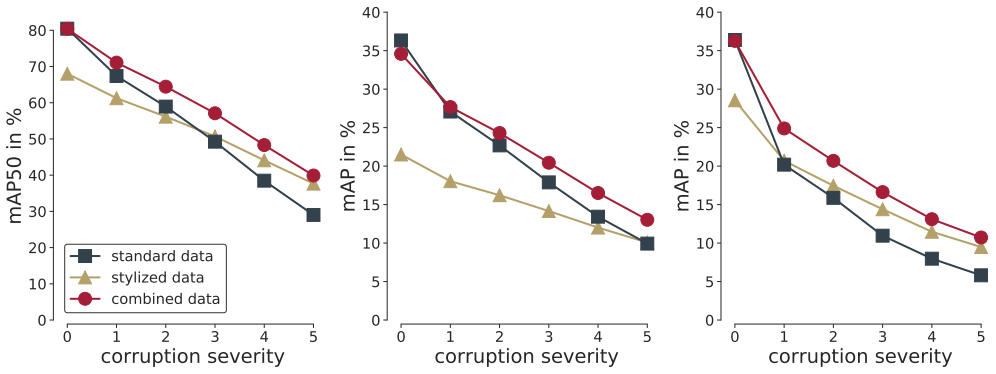
3.3 Training on stylized data improves robustness

In order to reduce the strong effect of corruptions on model performance observed above, we tested whether a simple approach (stylizing the training data) leads to a robustness improvement. We evaluate the exact same model (Faster R-CNN) with three different training data schemes (visualized in Figure 4):

standard: the unmodified training data of the respective dataset
stylized: the training data is stylized completely
combined: concatenation of standard and stylized training data

The results across our three datasets PASCAL-C, COCO-C and Cityscapes-C are visualized in Figure 5. We observe a similar pattern as reported by Geirhos et al. [2019] for object classification on ImageNet—a model trained on stylized data suffers less from corruptions than the model trained

[7] This finding is further supported by investigating models with deformable convolutions (see Appendix D).



(a) PASCAL-C (b) COCO-C (c) Cityscapes-C

Figure 5: Figure 5: Training on stylized data improves test performance of Faster R-CNN on corrupted versions of PASCAL VOC, MS COCO and Cityscapes which include all 15 types of corruptions shown in Figure 3. Corruption severity 0 denotes clean data. Corruption specific performances are shown in the appendix (Figures 7, 8, 9).

	PASCAL VOC [AP^{50}]			MS COCO [AP]			Cityscapes [AP]		
	clean	corr.	rel.	clean	corr.	rel.	clean	corr.	rel.
train data	P	mPC	rPC [%]	P	mPC	rPC [%]	P	mPC	rPC [%]
standard	**80.5**	48.6	60.4	**36.3**	18.2	50.2	**36.4**	12.2	33.4
stylized	68.0	50.0	**73.5**	21.5	14.1	**65.6**	28.5	14.7	**51.5**
combined	80.4	**56.2**	69.9	34.6	**20.4**	58.9	36.3	**17.2**	47.4

Table 2: Object detection performance of Faster R-CNN trained on standard images, stylized images and the combination of both evaluated on standard test sets (test 2007 for PASCAL VOC; val 2017 for MS COCO, val for Cityscapes); higher is better.

only on the original clean data. However, its performance on clean data is much lower. Combining stylized and clean data seems to achieve the best of both worlds: high performance on clean data as well as strongly improved performance under corruption. From the results in Table 2, it can be seen that both stylized and combined training improve the relative performance under corruption (rPC). Combined training yields the highest absolute performance under corruption (mPC) for all three datasets. This pattern is fairly consistent. Detailed results across corruption types are reported in the Appendix (Figure 7, Figure 8 and Figure 9).

3.4 Training directly on stylized data is better than using stylized data only during pre-training

For comparison reasons, we reimplemented the object detection models from Geirhos et al. [2019] and tested them for corruption robustness. Those models use backbones which are pre-trained with Stylized-ImageNet, but the object detection models are trained on the standard clean training sets of Pascal VOC and COCO. In contrast, we here use backbones trained on standard "clean" ImageNet and train using stylized Pascal VOC and COCO. We find that stylized pre-training helps not only on clean data (as reported by Geirhos et al. [2019]) but also for corruption robustness (Table 3), albeit less than our approach of performing the final training on stylized data (compare to Table 2)[8].

3.5 Robustness to natural distortions is connected to synthetic corruption robustness

A central question is whether results on the robust detection benchmark generalize to real-world natural distortions like rain, snow or fog as illustrated in Figure 2. We test this using BDD100k [Yu et al., 2018], a driving scene dataset with annotations for weather conditions. For our first experiment, we train a model only on images that are taken in "clear" weather. We also train models on a stylized

[8]Note that Geirhos et al. [2019] use Faster R-CNN without Feature Pyramids (FPN), which is why the baseline performance of these models is different from ours

	PASCAL VOC [AP^{50}]			MS COCO [AP]		
train data	clean P	corr. mPC	rel. rPC [%]	clean P	corr. mPC	rel. rPC [%]
IN	78.9	45.7	57.4	31.8	15.5	48.7
SIN	75.1	48.2	63.6	29.8	15.3	51.3
SIN+IN	78.0	**50.6**	**64.2**	31.1	16.0	**51.4**
SIN+IN ft IN	**79.0**	48.9	61.4	**32.3**	**16.2**	50.1

Table 3: Object detection performance of Faster R-CNN pre-trained on ImageNet (IN), Stylized ImageNet (SIN) and the combination of both evaluated on standard test sets (test 2007 for PASCAL VOC; val 2017 for MS COCO); higher is better.

BDD100k [AP]	Weather					Day/Night		
train data	clear P	rainy mPC	rel. rPC [%]	snowy mPC	rel. rPC [%]	day P	night mPC	rel. rPC [%]
clean	**27.8**	27.6	99.3	23.6	84.9	**30.0**	21.5	71.7
stylized	20.9	21.0	100.5	18.7	**89.5**	24.0	16.8	70.0
combined	27.7	**28.0**	**101.1**	**24.2**	87.4	**30.0**	**22.5**	**75.0**

Table 4: Performance of Faster R-CNN across different weather conditions and time changes when trained on standard images, stylized images and the combination of both evaluated on BDD100k (see Appendix C for dataset details); higher is better.

version of the same images as well as the combination of both following the protocol from Section 3.3. We then test these models on images which are annotated to be "clear", "rainy" or "snowy" (see Appendix C for details). We find that these weather changes have little effect on performance on all three models, but that combined training improves the generalization to "rainy" and "snowy" images (Table 4 Weather). It may be important to note that the weather changes of this dataset are often relatively benign (e.g., images annotated as rainy often show only wet roads instead of rain).

A stronger test is generalization of a model trained on images taken during daytime to images taken at night which exhibit a strong appearance change. We find that a model trained on images taken during the day performs much worse at night but combined training improves nighttime performance (Table 4 Day/Night and Appendix C).

As a third test of real-world distortions, we test our approach on Foggy Cityscapes Sakaridis et al. [2018a] which uses fog in three different strengths (given by the attenuation factor β = 0.005, 0.01 or $0.2m^{-1}$) as a highly realistic model of natural fog. Fog drastically reduces the performance of standard models trained on Cityscapes which was collected in clear conditions. The reduction is almost 50% for the strongest corruption, see Table 5. In this strong test for OOD (out-of-distribution) robustness, stylized training increases relative performance substantially from about 50% to over 70% (Table 5).

Taken together, these results suggest that there is a connection between performance on synthetic and natural corruptions. Our approach of combined training with stylized data improves performance in every single case with increasing gains in harder conditions.

3.6 Performance degradation does not simply scale with perturbation size

We investigated whether there is a direct relationship between the impact of a corruption on the pixel values of an image and the impact of a corruption on model performance. The left of Figure 6 shows the relative performance of Faster R-CNN on the corruptions in PASCAL-C dependent on the perturbation size of each corruption measured in Root Mean Square Error (RMSE). It can be seen that no simple relationship exists, counterintuitively robustness increases to corruption types with higher perturbation size (there is a weak positive correlation between rPC and RMSE, $r = 0.45$). This stems from the fact that corruptions like Fog or Brightness alter the image globally (resulting in high RMSE) while leaving local structure unchanged. Corruptions like Impulse Noise alter only a few pixels (resulting in low RMSE) but have a drastic impact on model performance.

To investigate further if classical perceptual image metrics are more predictive, we look at the relationship between the perceived image quality of the original and corrupted images measured in

Foggy Cityscapes [AP]		$\beta = 0.005$		$\beta = 0.01$		$\beta = 0.02$	
train data	clean P	corr. mPC	rel. rPC [%]	corr. mPC	rel. rPC [%]	corr. mPC	rel. rPC [%]
standard	**36.4**	30.2	83.0	25.1	69.0	18.7	51.4
stylized	28.5	26.2	**91.9**	24.7	**86.7**	22.5	**78.9**
combined	36.3	**32.2**	88.7	**29.9**	82.4	**26.2**	72.2

Table 5: Object detection performance of Faster R-CNN on Foggy Cityscapes when trained on Cityscapes with standard images, stylized images and the combination of both evaluated on the validation set; higher is better; β is the attenuation coefficient in m^{-1}

structural similarity (SSIM, higher value means more similar, Figure 6 on the right). There is a weak correlation between rPC and SSIM ($r = 0.48$). This analysis shows that SSIM better captures the effect of the corruptions on model performance.

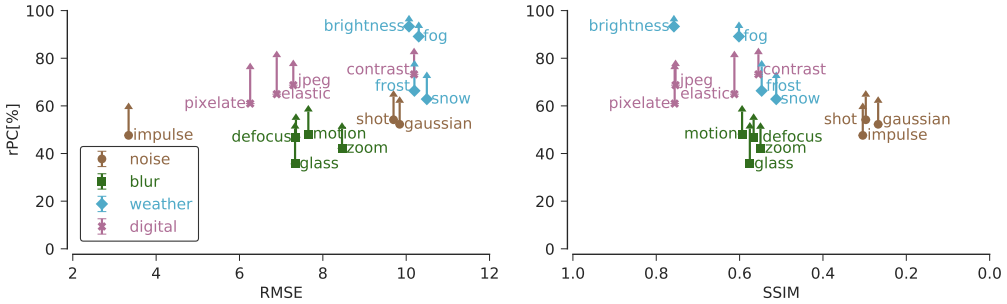


Figure 6: Relative performance under corruption (rPC) as a function of corruption RMSE (left, higher value=greater change in pixel space) and SSIM (right, higher value=higher perceived image quality) evaluated on PASCAL VOC. The dots indicate the rPC of Faster R-CNN trained on standard data; the arrows show the performance gained via training on 'combined' data. Corruptions are grouped into four corruption types: noise, blur, weather and digital.

4 Discussion

We here showed that object detection and instance segmentation models suffer severe performance impairments on corrupted images. This drop in performance has previously been observed in image recognition models [e.g. Geirhos et al., 2018, Hendrycks and Dietterich, 2019]. In order to track future progress on this important issue, we propose the `Robust Detection Benchmark` containing three easy-to-use benchmark datasets PASCAL-C, COCO-C and Cityscapes-C. We provide evidence that performance on our benchmarks predicts performance on natural distortions and show that robustness corresponds to model performance on clean data. Apart from providing baselines, we demonstrate how a simple data augmentation technique, namely adding a stylized copy of the training data in order to reduce a model's focus on textural information, leads to strong robustness improvements. On corrupted images, we consistently observe a performance increase (about 16% for PASCAL, 12% for COCO, and 41% for Cityscapes) with small losses on clean data (0–2%). This approach has the benefit that it can be applied to any image dataset, requires no additional labelling or model tuning and, thus, comes basically for free. At the same time, our benchmark data shows that there is still space for improvement and it is yet to be determined whether the most promising robustness enhancement techniques will require architectural modifications, data augmentation schemes, modifications to the loss function, or a combination of these.

We encourage readers to expand the benchmark with novel corruption types. In order to achieve robust models, testing against a wide variety of different image corruptions is necessary—there is no 'too much'. Since our benchmark is open source, we welcome new corruption types and look forward to your pull requests to `https://github.com/bethgelab/imagecorruptions`! We envision our comprehensive benchmark to track future progress towards building robust object detection models

that can be reliably deployed 'in the wild', eventually enabling them to cope with unexpected weather changes, corruptions of all kinds and, if necessary, even the occasional dragonfire.

Author contributions

The initial project idea for improving detection robustness was developed by E.R., R.G. and C.M. The initial idea of benchmarking detection robustness was developed by C.M., B.M., R.G., E.R. & W.B. The overall research focus on robustness was collaboratively developed in the Bethge, Bringmann and Wichmann labs. The `Robust Detection Benchmark` was jointly designed by C.M., B.M., R.G. & E.R.; including selecting datasets, corruptions, metrics and models. B.M. and E.R. jointly developed the pip-installable package to corrupt arbitrary images. B.M. developed code to stylize arbitrary datasets with input from R.G. and C.M.; C.M. and B.M. developed code to evaluate the robustness of arbitrary object detection models. B.M. prototyped the core experiments; C.M. ran the reported experiments. The results were jointly analysed and visualized by C.M., R.G. and B.M. with input from E.R., M.B. and W.B.; C.M., B.M., R.G. & E.R. worked towards making our work reproducible, i.e. making data, code and benchmark openly accessible and (hopefully) user-friendly. Senior support, funding acquisition and infrastructure were provided by O.B., A.S.E., M.B. and W.B. The illustratory figures were designed by E.R., C.M. and R.G. with input from B.M. and W.B. The paper was jointly written by R.G., C.M., E.R. and B.M. with input from all other authors.

Acknowledgement

We would like to thank Alexander von Bernuth for help with Figure 1; Marissa Weis for help with the Cityscapes dataset; Andreas Geiger for helpful discussions on the topic of autonomous driving in bad weather; Mackenzie Mathis for helpful contributions to the stylization code as well as Eshed Ohn-Bar and Jan Lause for pointing us to important references. R.G. would like to acknowledge Felix Wichmann for senior support, funding acquisition and providing infrastructure. C.M., R.G. and E.R. graciously acknowledge support by the International Max Planck Research School for Intelligent Systems (IMPRS-IS) and the Iron Bank of Braavos. C.M. was supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) via grant EC 479/1-1 to A.S.E.. A.S.E., M.B. and W.B. acknowledge support from the BMBF competence center for machine learning (FKZ 01IS18039A) and the Collaborative Research Center Robust Vision (DFG Projektnummer 276693517 – SFB 1233: Robust Vision). M.B. acknowledges support by the Centre for Integrative Neuroscience Tübingen (EXC 307). O.B. and E.R. have been partially supported by the Deutsche Forschungsgemeinschaft (DFG) in the priority program 1835 "Cooperatively Interacting Automobiles" under grant BR2321/5-1 and BR2321/5-2. A.S.E., M.B. and W.B. were supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior / Interior Business Center (DoI/IBC) contract number D16PC00003.

References

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NIPS*, 2015.

Robert Geirhos, Carlos RM Temme, Jonas Rauber, Heiko H Schütt, Matthias Bethge, and Felix A Wichmann. Generalisation in humans and deep neural networks. In *NeurIPS*, 2018.

Reidar P Lystad and Benjamin T Brown. "Death is certain, the time is not": mortality and survival in Game of Thrones. *Injury epidemiology*, 5(1):44, 2018.

Michael A Alcorn, Qi Li, Zhitao Gong, Chengfei Wang, Long Mai, Wei-Shinn Ku, and Anh Nguyen. Strike (with) a pose: Neural networks are easily fooled by strange poses of familiar objects. In *CVPR*, 2019.

John R Zech, Marcus A Badgeley, Manway Liu, Anthony B Costa, Joseph J Titano, and Eric Karl Oermann. Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs: A cross-sectional study. *PLoS medicine*, 15(11):e1002683, 2018.

Hugo Touvron, Andrea Vedaldi, Matthijs Douze, and Hervé Jégou. Fixing the train-test resolution discrepancy. *arXiv:1906.06423*, 2019.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv:1312.6199*, 2013.

Dengxin Dai and Luc Van Gool. Dark model adaptation: Semantic image segmentation from daytime to nighttime. In *ITSC*, 2018.

Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Semantic foggy scene understanding with synthetic data. *IJCV*, 2018a.

Dennis Hospach, Stefan Müller, Wolfgang Rosenstiel, and Oliver Bringmann. Simulating photo-realistic snow and fog on existing images for enhanced CNN training and evaluation. In *DATE*, 2016.

Alexander Von Bernuth, Georg Volk, and Oliver Bringmann. Simulating photo-realistic snow and fog on existing images for enhanced CNN training and evaluation. In *ITSC*, 2019.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *ICLR*, 2019.

Samuel Fuller Dodge and Lina J. Karam. Understanding how image quality affects deep neural networks. *QoMEX*, 2016.

Aharon Azulay and Yair Weiss. Why do deep convolutional networks generalize so poorly to small image transformations? *arXiv:1805.12177*, 2018.

Jashojit Mukherjee, K Praveen, and Venugopala Madumbu. Visual quality enhancement of images under adverse weather conditions. In *ITSC*, 2018.

Chris H. Bahnsen and Thomas B. Moeslund. Rain removal in traffic surveillance: Does it matter? *arXiv:1810.12574*, 2018.

Chris H. Bahnsen, David Vázquez, Antonio M. López, and Thomas B. Moeslund. Learning to remove rain in traffic surveillance by using synthetic data. In *VISIGRAPP*, 2019.

Igor Vasiljevic, Ayan Chakrabarti, and Gregory Shakhnarovich. Examining the impact of blur on recognition by convolutional networks. *arXiv:1611.05760*, 2016.

Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A. Wichmann, and Wieland Brendel. ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In *ICLR*, 2019.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115(3): 211–252, 2015.

Georg Volk, Stefan Müller, Alexander von Bernuth, Dennis Hospach, and Oliver Bringmann. Towards robust CNN-based object detection through augmentation with synthetic rain variations. In *ITSC*, 2019.

Yuhua Chen, Wen Li, Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Domain adaptive faster R-CNN for object detection in the wild. In *CVPR*, 2018.

Unghui Lee, Jiwon Jung, Seokwoo Jung, and David Hyunchul Shim. Development of a self-driving car that can handle the adverse weather. *International journal of automotive technology*, 2018.

Christos Sakaridis, Dengxin Dai, Simon Hecker, and Luc Van Gool. Model adaptation with synthetic and real data for semantic dense foggy scene understanding. In *ECCV*, 2018b.

Alexander Von Bernuth, Georg Volk, and Oliver Bringmann. Rendering physically correct raindrops on windshields for robustness verification of camera-based object recognition. *Intelligent Vehicles Symposium (IV)*, pages 922–927, 2018.

Longyin Wen, Dawei Du, Zhaowei Cai, Zhen Lei, Ming-Ching Chang, Honggang Qi, Jongwoo Lim, Ming-Hsuan Yang, and Siwei Lyu. UA-DETRAC: A new benchmark and protocol for multi-object detection and tracking. *arXiv:1511.04136*, 2015.

Fisher Yu, Wenqi Xian, Yingying Chen, Fangchen Liu, Mike Liao, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving video database with scalable annotation tooling. *arXiv:1805.04687*, 2018.

Zhengping Che, Guangyu Li, Tracy Li, Bo Jiang, Xuefeng Shi, Xinsheng Zhang, Ying Lu, Guobin Wu, Yan Liu, and Jieping Ye. D2-city: A large-scale dashcam video dataset of diverse traffic scenarios. *arXiv:1904.01975*, 2019.

Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. *arXiv:1903.11027*, 2019.

Adrien Gaidon, Qiao Wang, Yohann Cabon, and Eleonora Vig. Virtual worlds as proxy for multi-object tracking analysis. In *CVPR*, 2016.

German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio M. Lopez. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *CVPR*, 2016.

Stephan R. Richter, Zeeshan Hayder, and Vladlen Koltun. Playing for benchmarks. In *ICCV*, 2017.

M. Johnson-Roberson, Charles Barto, Rounak Mehta, Sharath Nittur Sridhar, Karl Rosaen, and Ram Vasudevan. Driving in the matrix: Can virtual worlds replace human-generated annotations for real world tasks? In *ICRA*, 2017.

Peter Radecki, Mark Campbell, and Kevin Matzen. All weather perception: Joint data association, tracking, and classification for autonomous ground vehicles. *CoRR*, abs/1605.02196, 2016. URL `http://arxiv.org/abs/1605.02196`.

Kexin Pei, Yinzhi Cao, Junfeng Yang, and Suman Jana. Towards practical verification of machine learning: The case of computer vision systems. *arXiv:1712.01785*, 2017.

Harshitha Machiraju and Sumohana Channappayya. An evaluation metric for object detection algorithms in autonomous navigation systems and its application to a real-time alerting system. In *25th IEEE International Conference on Image Processing (ICIP)*, 2018.

Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. The Pascal Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision*, 2010.

Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *CVPR*, 2012.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common Objects in Context. In *ECCV*, 2014.

Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016.

Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ADE20K dataset. In *CVPR*, 2017.

Gerhard Neuhold, Tobias Ollmann, Samuel Rota Bulò, and Peter Kontschieder. The mapillary vistas dataset for semantic understanding of street scenes. In *ICCV*, 2017.

Ivan Krasin, Tom Duerig, Neil Alldrin, Vittorio Ferrari, Sami Abu-El-Haija, Alina Kuznetsova, Hassan Rom, Jasper Uijlings, Stefan Popov, Shahab Kamali, Matteo Malloci, Jordi Pont-Tuset, Andreas Veit, Serge Belongie, Victor Gomes, Abhinav Gupta, Chen Sun, Gal Chechik, David Cai, Zheyun Feng, Dhyanesh Narayanan, and Kevin Murphy. Openimages: A public dataset for large-scale multi-label and multi-class image classification. *Dataset available from https://storage.googleapis.com/openimages/web/index.html*, 2017.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *ICCV*, 2017.

Zhaowei Cai and Nuno Vasconcelos. Cascade R-CNN: Delving into high quality object detection. In *CVPR*, 2018.

Kai Chen, Jiangmiao Pang, Jiaqi Wang, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. Hybrid task cascade for instance segmentation. In *CVPR*, 2019a.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal Loss for Dense Object Detection. *ICCV*, 2017a.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature Pyramid Networks for Object Detection. In *CVPR*, 2017b.

Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *CVPR*, 2017.

Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *ICCV*, 2017.

Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai. Deformable convnets v2: More deformable, better results. *arXiv:1811.11168*, 2018.

Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, et al. Mmdetection: Open mmlab detection toolbox and benchmark. *arXiv:1906.07155*, 2019b.

Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2414–2423, 2016.

Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *ICCV*, pages 1501–1510, 2017.

Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch SGD: Training ImageNet in 1 hour. *arXiv:1706.02677*, 2017.

Agrim Gupta, Piotr Dollar, and Ross Girshick. LVIS: A dataset for large vocabulary instance segmentation. In *CVPR*, 2019.

Appendix

A Implementation details: Model training

We train all our models with two images per GPU which corresponds to a batch size of 16 on eight GPUs. On COCO, we resize images so that their short edge is 800 pixels and train for twelve epochs with a starting learning rate of 0.01 which is decreased by a factor of ten after eight and eleven epochs. On PASCAL VOC, images are resized so that their short edge is 600 pixels. Training is done for twelve epochs with a starting learning rate of 0.00125 with a decay step of factor ten after nine epochs. For Cityscapes, we stayed as close as possible to the procedure described in [He et al., 2017], rescaling images to a shorter edge size between 800 and 1024 pixels and train for 64 epochs (to match 24k steps at a batch size of eight) with an initial learning rate of 0.0025 and a decay step of factor ten after 48 epochs. For evaluation, only one scale (1024 pixels) is used. Specifically, we used four GPUs to train the COCO models and one GPU for all other models[9] Training with stylized data is done by simply exchanging the dataset folder or adding it to the list of dataset folders to consider. For all further details please refer to the config files in our implementation (which we will make available after the end of the anonymous review period).

B Corrupting arbitrary images

In the original corruption benchmark of ImageNet-C [Hendrycks and Dietterich, 2019], two technical aspects are hard-coded: The image-dimensions and the number of channels. To allow for different data sets with different image dimensions, several corruption functions are defined independently of each other, such as `make_cifar_c`, `make_tinyimagenet_c`, `make_imagenet_c` and `make_imagenet_c_inception`. Additionally, many corruptions expect quadratic images. We have modified the code to resolve these constraints and now all corruptions can be applied to non-quadratic images with varying sizes, which is a necessary prerequisite for adapting the corruption benchmark to the PASCAL VOC and COCO datasets. For the corruption type Frost, crops from provided images of frost are added to the input images. Since images in PASCAL VOC and COCO have arbitrarily large dimensions, we resize the frost images to fit the largest input image dimension if necessary. The original corruption benchmark also expects RGB images. Our code now allows for grayscale images.[10] Both `motion_blur` and `snow` relied on the motion-blur functionality of Imagemagick, resulting in an external dependency that could not be resolved by standard Python package managers. For convenience, we reimplemented the motion-blur functionality in Python and removed the dependency on non-Python software.

C BDD100k

We use the weather annotations present in the BDD100k dataset Yu et al. [2018] to split it in images with clear, rainy and snowy conditions. We disregard all images which are annotated to have any other weather condition (foggy, partly cloudy, overcast and undefined) to make the separation easier[11]. We use all images from the training set which are labeled having clear weather conditions for training. For testing, we created 3 subsets of the validation set each containing 725 images in clear, rainy or snowy conditions[12]. The sets were created to have the same size which was determined by the category with the least images (rainy). Having same sized test sets is important because evaluation under the AP metric leads to lower scores with increasing sequence length [Gupta et al., 2019].

[9] In all our experiments, we employ the linear scaling rule [Goyal et al., 2017] to select the appropriate learning rate.

[10] There are approximately 2–3% grayscale images in PASCAL VOC/MS COCO.

[11] It would have been great to combine the performance on natural fog with the results from Foggy Cityscapes but as there are only 13 foggy images in the validation set the results cannot be seen as representative in any way

[12] We will release the datasets splits at `https://github.com/bethgelab/robust-detection-benchmark`

MS COCO

model	backbone	clean P [AP]	corr. mPC [AP]	rel. rPC [%]
Mask	r50	34.2	16.8	49.1
Cascade Mask	r50	35.7	17.6	49.3
HTC	x101-64x4d	43.8	28.1	64.0

Cityscapes

model	backbone	clean P [AP]	corr. mPC [AP]	rel. rPC [%]
Mask	r50	32.7	10.0	30.5

Table 6: **Instance segmentation** performance of various models. Backbones indicated with r: ResNet. All model names indicate the corresponding model from the R-CNN family. All models were downloaded from the `mmdetection` modelzoo.

	MS COCO			Cityscapes		
train data	clean [P]	corr. [mPC]	rel. [rPC]	clean [P]	corr. [mPC]	rel. [rPC]
standard	**34.2**	16.9	49.4	**32.7**	10.0	30.5
stylized	20.5	13.2	**64.1**	23.0	11.3	**49.2**
combined	32.9	**19.0**	57.7	32.1	**14.9**	46.3

Table 7: **Instance segmentation** performance of Mask R-CNN trained on standard images, stylized images and the combination of both evaluated on standard test sets (test 2007 for PASCAL VOC; val 2017 for MS COCO, val for Cityscapes).

D Additional Results

D.1 Instance Segmentation Results

We evaluated Mask R-CNN and Cascade Mask R-CNN on instance segmentation. The results are very similar to those on the object detection task with a slightly lower relative performance (1%, see Table 6). We also trained Mask R-CNN on the stylized datasets finding again very similar trends for the instance segmentation task as for the object detection task (Table 7). On the one hand, this is not very surprising as Mask R-CNN and Faster R-CNN are very similar. On the other hand, the contours of objects can change due to the stylization process, which would expectedly lead to poor segmentation performance when training only on stylized images. We do not see such an effect but rather find the instance segmentation performance of Mask R-CNN to mirror the object detection performance of Faster R-CNN when trained on stylized images.

D.2 Deformable Convolutional Networks

We tested the effect of deformable convolutions [Dai et al., 2017, Zhu et al., 2018] on corruption robustness. Deformable convolutions are a modification of the backbone architecture exchanging some standard convolutions with convolutions that have adaptive filters in the last stages of the encoder. It has been shown that deformable convolutions can help on a range of tasks like object detection and instance segmentation. This is the case here too as networks with deformable convolutions do not only perform better on clean but also on corrupted images improving relative performance by 6-7% compared to the baselines with standard backbones (See Tables 8 and 9). The effect appears to be the same as for other backbone modifications such as using deeper architectures (See Section 3 in the main paper).

Image rights & attribution

Figure 1: Home Box Office, Inc. (HBO).

		MS COCO		
model	backbone	clean P [AP]	corr. mPC [AP]	rel. rPC [%]
Faster	r50-dcn	40.0	22.4	56.1
Faster	x101-64x4d-dcn	43.4	26.7	61.6
Mask	r50-dcn	41.1	23.3	56.7

Table 8: **Object detection** performance of models with deformable convolutions Dai et al. [2017]. Backbones indicated with r are ResNet, the addition dcn signifies deformable convolutions in stages c3-c5. All model names indicate the corresponding model from the R-CNN family. All models were downloaded from the `mmdetection` modelzoo.

		MS COCO		
model	backbone	clean P [AP]	corr. mPC [AP]	rel. rPC [%]
Mask	r50-dcn	37.2	20.7	55.7

Table 9: **Instance segmentation** performance of Mask R-CNN with deformable convolutions [Dai et al., 2017]. The backbone indicated with r is a ResNet 50, the addition dcn signifies deformable convolutions in stages c3-c5. The model was downloaded from the `mmdetection` modelzoo.

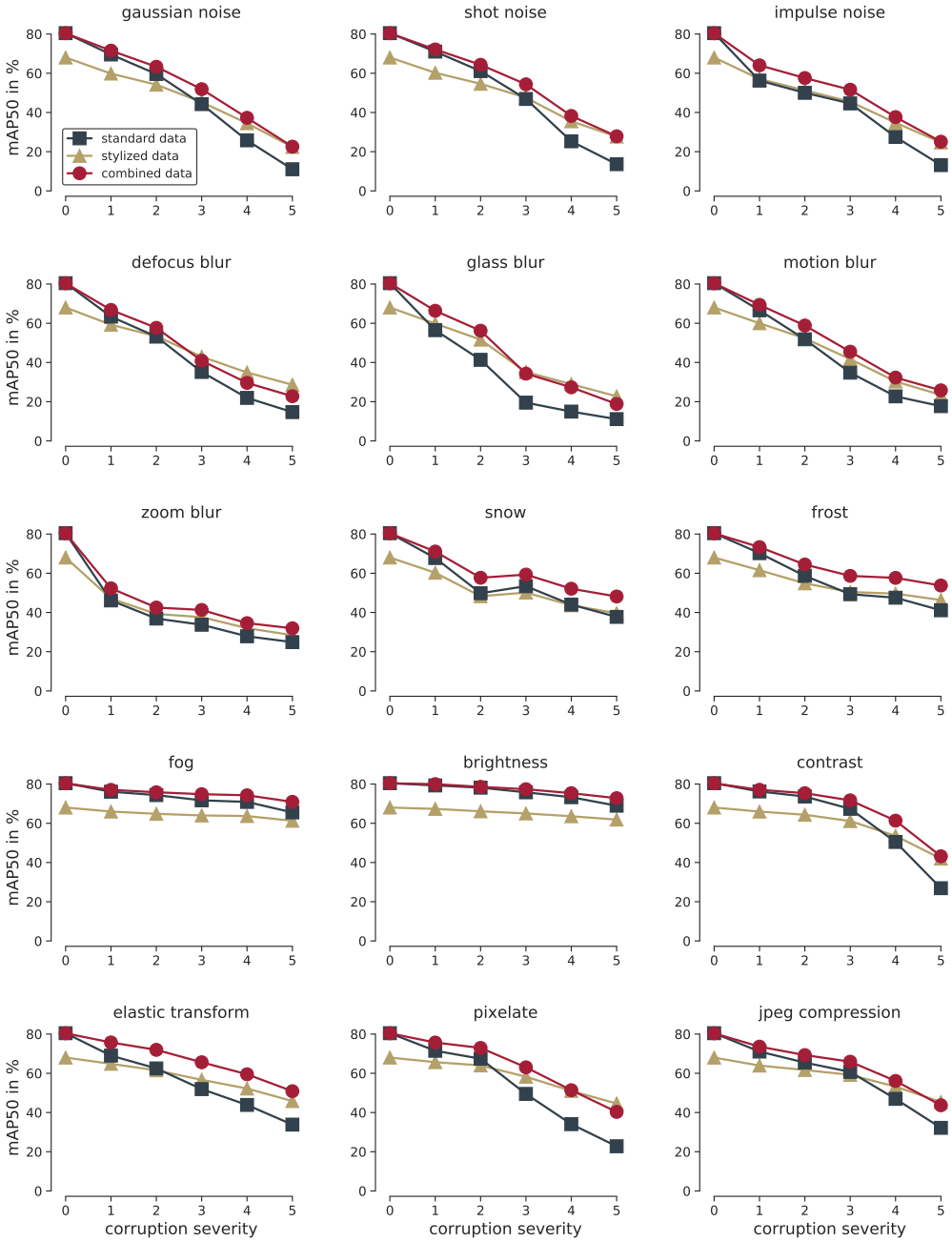


Figure 7: Results for each corruption type on PASCAL-C.

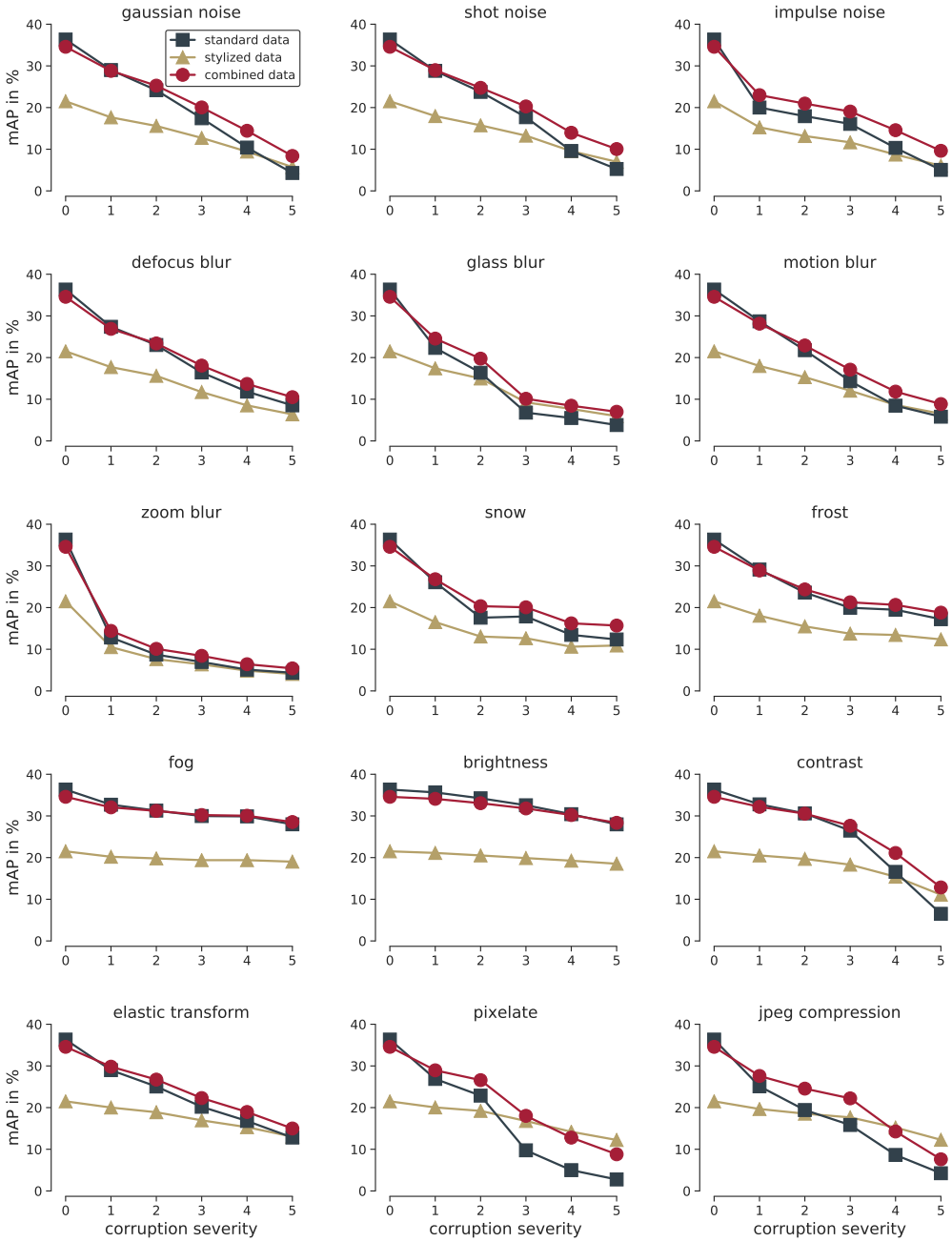


Figure 8: Results for each corruption type on COCO-C.

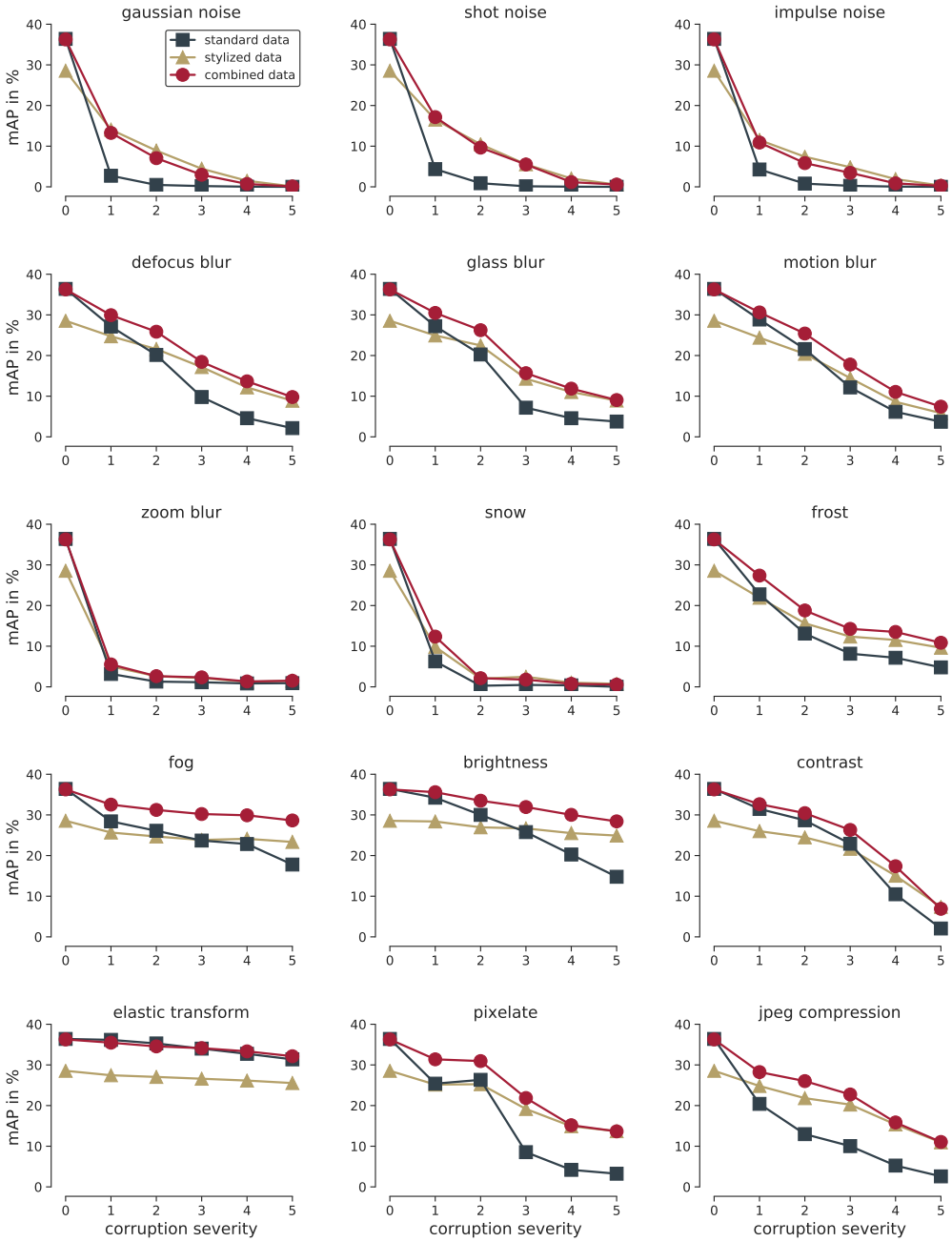


Figure 9: Results for each corruption type on Cityscapes-C.

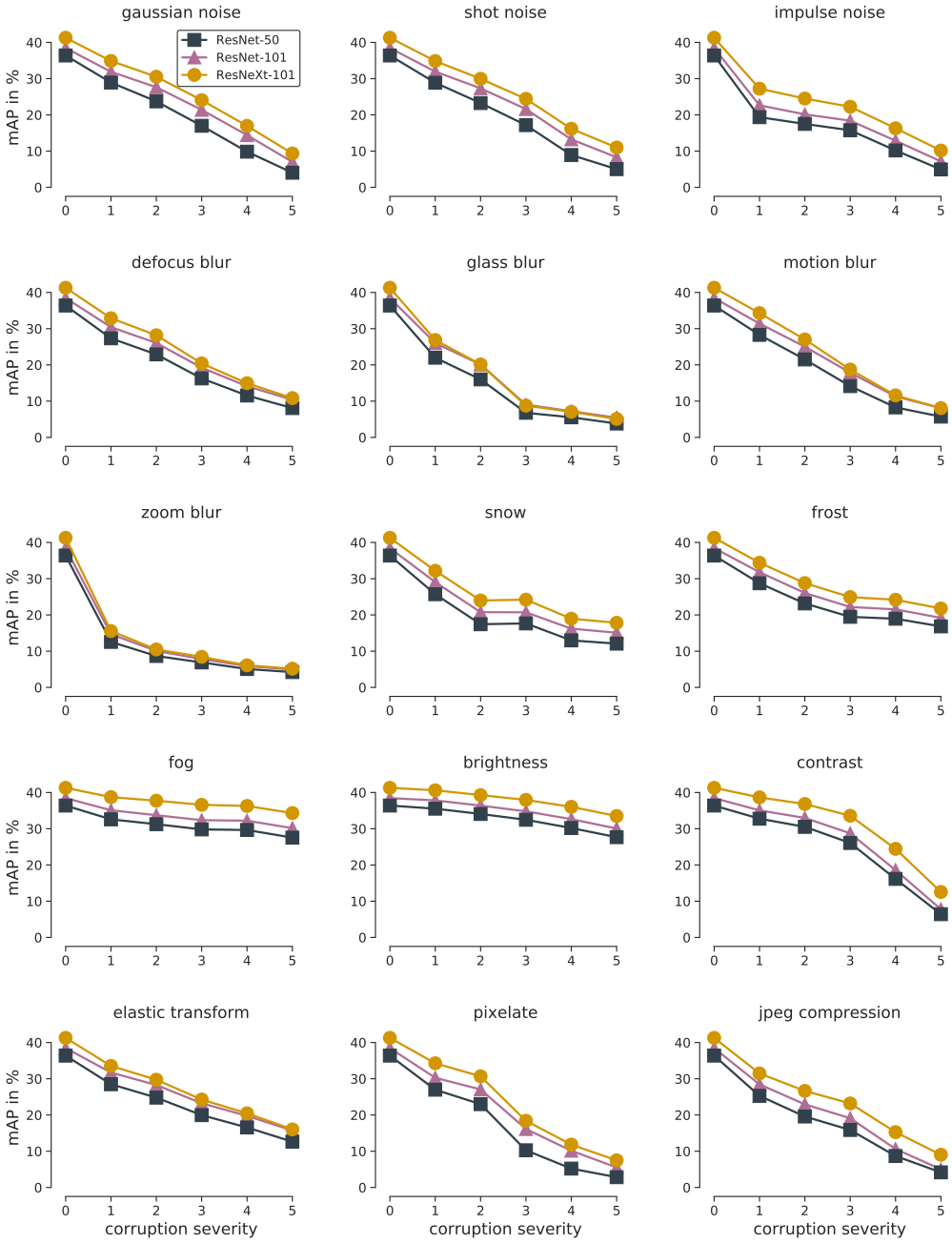


Figure 10: Results for each corruption type using different backbones. Faster R-CNN trained on MS COCO with ResNet-50, ResNet-101 and ResNext-101_64x4d backbones.

arXiv:2407.00143v2 [cs.LG] 16 Apr 2025

InfoNCE: Identifying the Gap Between Theory and Practice

Evgenia Rusak[*,1,2] **Patrik Reizinger**[*,2,4] **Attila Juhos**[*,2,4] **Oliver Bringmann**[1]
Roland Zimmermann[†,2,4] **Wieland Brendel**[†,2,3,4]

[1]University of Tübingen [2]Max Planck Institute for Intelligent Systems
[3]ELLIS Institute Tübingen [4]Tübingen AI Center

Abstract

Prior theory work on Contrastive Learning via the InfoNCE loss showed that, under certain assumptions, the learned representations recover the ground-truth latent factors. We argue that these theories overlook crucial aspects of how CL is deployed in practice. Specifically, they either assume equal variance across all latents or that certain latents are kept invariant. However, in practice, positive pairs are often generated using augmentations such as strong cropping to just a few pixels. Hence, a more realistic assumption is that all latent factors change with a continuum of variability across all factors. We introduce AnInfoNCE, a generalization of InfoNCE that can provably uncover the latent factors in this anisotropic setting, broadly generalizing previous identifiability results in CL. We validate our identifiability results in controlled experiments and show that AnInfoNCE increases the recovery of previously collapsed information in CIFAR10 and ImageNet, albeit at the cost of downstream accuracy. Finally, we discuss the remaining mismatches between theoretical assumptions and practical implementations.

1 INTRODUCTION

Self-Supervised Learning (SSL) employs surrogate objectives to learn useful representations from large, unlabeled datasets. A particularly successful and simple sub-family, called *Contrastive Learning (CL)*, minimizes the representational distance between similar samples (e.g., augmentations of the same sample) while maximizing the distance between dissimilar samples (drawn i.i.d.) (Oord et al., 2018; Chen et al., 2020a,c; Caron et al., 2020; Chen et al., 2020b). Two similar samples form a *positive pair*, while two dissimilar samples form a *negative pair*. Most of today's successful SSL techniques build on this principle, including non-contrastive ones that only use positive pairs (Chen and He, 2021; Grill et al., 2020; Zbontar et al., 2021).

InfoNCE (Oord et al., 2018) is a commonly used objective for CL. Zimmermann et al. (2021) have shown that under certain assumptions put on the *Data Generating Process (DGP)*, a model can recover the ground-truth latent factors if trained with InfoNCE. Their identifiability result requires an isotropic change across all latent factors in the positive pair. Von Kügelgen et al. (2021) relaxed the isotropic assumption on the positive conditional distributions to two partitions: content (δ-distribution) and style (non-degenerate distribution), yielding a weaker block-identifiability result.

In practice, positive pairs for CL are generated using augmentations. For instance, strong cropping, which often extracts just a tiny area of the image, is argued to be crucial for high downstream performance (Chen et al., 2020a). Assuming an underlying latent variable model, these augmentations correspond to changes in latent factors, modeled by the positive conditional distribution. Cropping discards a significant amount of information and, hence, harshly changes some latent factors. Another augmentation, Gaussian blurring, causes moderate perturbations to the sharpness of objects. Lastly, latents alluding to class information are invariably left intact. We posit that augmentations imply anisotropic changes to different latent factors, a setting previously not covered theoretically. In fact, we believe that this gap between theory and practice can in part explain why empirical work shows that CL loses information (Chen et al., 2020a; Jing et al., 2022),

[*]Equal contribution. [†]Joint senior authors.
Correspondence to `evgenia.rusak@uni-tuebingen.de`

Code at `github.com/brendel-group/AnInfoNCE`.
Website at `brendel-group.github.io/AnInfoNCE/`.

Proceedings of the 28[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2025, Mai Khao, Thailand. PMLR: Volume 258. Copyright 2025 by the author(s).

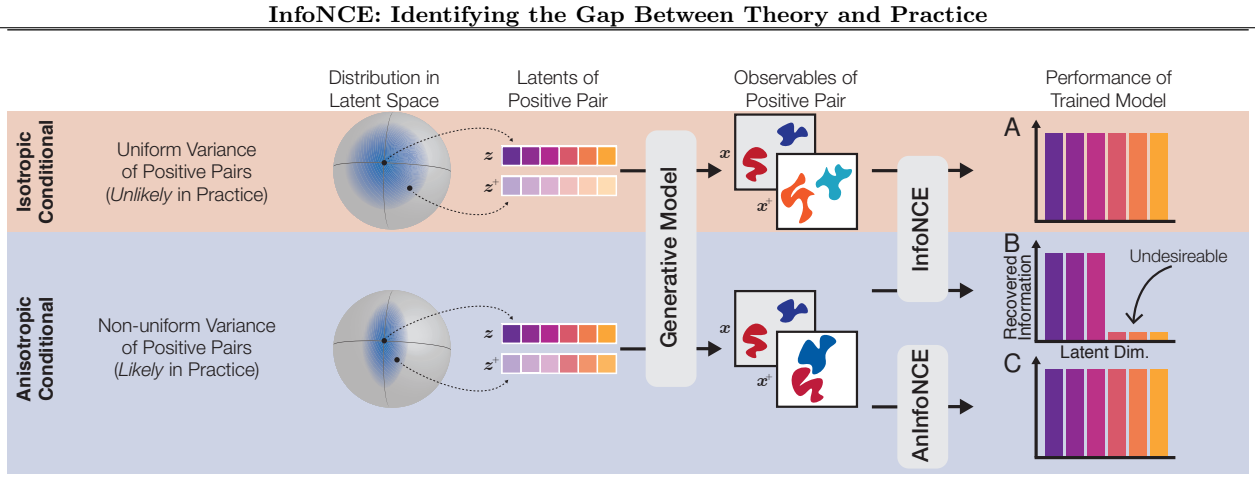


Figure 1: **Illustration of the Mismatch Between the Standard Contrastive Learning (CL) Model and Practice. A:** CL with the commonly used InfoNCE objective is identifiable when all latents change to the same extent across the positive pair (Zimmermann et al., 2021), which is unlikely to happen in practice. **B:** The more likely scenario, when augmentations affect different latents to a different extent, leads to dimensional collapse and information loss. **C:** Our proposed objective, Anistropic InfoNCE (AnInfoNCE), models features that can vary to a different degree in the positive pair, avoiding collapse.

despite the identifiability guarantees of Zimmermann et al. (2021); von Kügelgen et al. (2021).

The information loss in CL hinders the development of task-agnostic models, which require an accurate and holistic low- and high-level representation of their inputs. The failure of InfoNCE to capture style latents is already apparent in popular vision-language models (VLMs) trained by adaptations of InfoNCE, such as CLIP (Radford et al., 2021). By focusing on a few semantic concepts and dismissing low-level style information, CLIP-based VLMs struggle with low-level vision (Rahmanzadehgervi et al., 2024), scene compositionality (Yuksekgonul et al., 2023a), and attribute binding tasks (Trusca et al., 2024). Identifiability considers how to reconstruct *all* latent factors underlying the inputs, aiming to capture general-purpose representations, suitable for a plethora of downstream tasks.

Our work serves two purposes. Firstly, we push the frontier of identifiability research by generalizing existing results to anisotropic latent distributions, which allow for more expressive generative processes. Equally importantly, we design a non-trivial extension of InfoNCE, coined AnInfoNCE, that is provably able to capture these inhomogeneous latents. We validate our results in controlled experiments and show that AnInfoNCE successfully increases latent recovery on CIFAR10 and ImageNet, although at the expense of downstream classification accuracy. We perform ablations to understand the remaining gap caused by the accuracy–identifiability trade-off. Our **contributions**:

- We introduce AnInfoNCE, a generalized contrastive loss, that provably recovers latents with augmentation-induced anisotropic variances (§ 3.2).
- We extend the identifiability result to hard negative mining (§ 3.3) and loss ensembles (§ 3.4).
- We verify our loss in synthetic and well-controlled image experiments, having full knowledge of the ground-truth generative process (§ 4.1 and § 4.2).
- We further demonstrate the efficacy of AnInfoNCE on CIFAR10 and ImageNet in recovering the latent factors, but observe a trade-off with downstream classification accuracy, done with linear probing (§ 4.3).
- We link the accuracy–identifiability trade-off on real-world data to augmentations and analyze remaining mismatches between CL theory and practice (§ 5).

2 BACKGROUND

Contrastive Learning. Contrastive Learning (CL) is an SSL paradigm where an encoder $\boldsymbol{f} : \mathcal{X} \to \mathcal{Z}$ learns to map observations $\boldsymbol{x}$ to latent vectors $\boldsymbol{z}$. CL uses positive pairs $(\boldsymbol{x}, \boldsymbol{x}^+)$ (similar data points, e.g., augmentations of the same sample), and negative pairs $(\boldsymbol{x}, \boldsymbol{x}^-)$ (dissimilar data points, e.g., uniformly drawn from the dataset). CL mostly employs the InfoNCE loss (Sohn, 2016; Gutmann and Hyvärinen, 2010; Oord et al., 2018):

$$\mathcal{L}_{\text{INCE}}(\boldsymbol{f}) = \tag{1}$$
$$= \mathop{\mathbb{E}}_{\substack{\boldsymbol{x},\boldsymbol{x}^+ \\ \{\boldsymbol{x}_i^-\}}} \left[-\ln \frac{e^{\boldsymbol{f}^\top(\boldsymbol{x})\boldsymbol{f}(\boldsymbol{x}^+)/\boldsymbol{\tau}}}{e^{\boldsymbol{f}^\top(\boldsymbol{x})\boldsymbol{f}(\boldsymbol{x}^+)/\boldsymbol{\tau}} + \sum_{i=1}^{M} e^{\boldsymbol{f}^\top(\boldsymbol{x})\boldsymbol{f}(\boldsymbol{x}_i^-)/\boldsymbol{\tau}}} \right],$$

where $\boldsymbol{\tau}$ is a scalar temperature, and $\boldsymbol{x}^+ \sim p(\boldsymbol{x}^+|\boldsymbol{x})$ and $(\boldsymbol{x}, \boldsymbol{x}^-) \sim^{i.i.d.} p(\boldsymbol{x})$ are the positive and negative pairs observed.

In order to analyze the behavior of CL, we surmise that

the observations come from a Data Generating Process (DGP) $\boldsymbol{g} : \mathcal{Z} \to \mathcal{X}$. The samples follow a marginal distribution $p(\boldsymbol{z})$ over the latent space $\mathcal{Z}$, and the positive samples follow a conditional $p(\boldsymbol{z}^+|\boldsymbol{z})$. Under certain assumptions, CL inverts the DGP, i.e., the composition $\boldsymbol{h} = \boldsymbol{f} \circ \boldsymbol{g}$ is a trivial map (usually an affine linear map, depending on the assumptions) (Zimmermann et al., 2021). The identifiability of SSL is further analyzed by von Kügelgen et al. (2019); Lyu et al. (2021) and Cui et al. (2022), with insightful connections between contrastive and non-contrastive methods in Balestriero and LeCun (2022); Garrido et al. (2023); Assran et al. (2023).

Content-style Partitioning of $\mathcal{Z}$. Von Kügelgen et al. (2021) present an identifiability result for a partitioning of $\boldsymbol{z} \in \mathcal{Z}$ into invariant (*content*) $\boldsymbol{z}_c \in \mathcal{Z}_c$ and variant (*style*) $\boldsymbol{z}_s \in \mathcal{Z}_s$ dimensions with $\mathcal{Z} = \mathcal{Z}_c \times \mathcal{Z}_s$ (see Appx. A). The content latents $\boldsymbol{z}_c$ in the positive pairs are assumed to follow a δ-distribution, whereas the distribution of $\boldsymbol{z}_s$ is not degenerate, yielding $p(\boldsymbol{z}^+|\boldsymbol{z}) = \delta(\boldsymbol{z}_c^+ - \boldsymbol{z}_c)\, p(\boldsymbol{z}_s^+|\boldsymbol{z}_s)$. Note that Jing et al. (2022) implicitly use a similar distinction—defined as different augmentation amplitudes. Thus, their approach is akin to generalized content and style variables with variances $\boldsymbol{\sigma}_s^2 \gg \boldsymbol{\sigma}_c^2 > 0$; Dubois et al. (2022) also have a similar approach. Other means to partition the latent variables are tied to the "simplicity" of features (Chen et al., 2021) or sparsity arguments Wen and Li (2021). In concurrent work, Eastwood et al. (2023) propose a *relative* notion of content and style, assuming the presence of multiple environments, each with distinct content and style dimensions.

Practical Shortcomings of CL. Despite the identifiability guarantees of CL methods (Zimmermann et al., 2021; von Kügelgen et al., 2021; Cui et al., 2022), practitioners struggle to make models learn all underlying latent factors. Chen et al. (2020a) noted that the representations could become invariant to some input transformations. Several works connected data augmentations to failing to learn all latent factors (Jing et al., 2022; Wang et al., 2022; Bendidi et al., 2023; Wu et al., 2020; HaoChen et al., 2021; Cosentino et al., 2022; Wagner et al., 2022; Zhai et al., 2023).

A potential way to improve augmentation readout (a proxy for capturing the information in highly-varying, i.e., style, latents) is by introducing inductive biases. Lee et al. (2021a) add a regularizer to predict the augmentation parameters between two views, whereas Dangovski et al. (2021) use a regularizer to induce equivariance by predicting rotation parameters. Xiao et al. (2021) introduce Leave-one-out Contrastive Learning (LooC) with multiple latent spaces. They train multiple embeddings by leaving out one augmentation at a time, thus inducing different invariances, which results in better downstream and augmentation readout performance. Zhang and Ma (2022) propose a similar strategy with different augmentation partitions and calculate the contrastive loss at different hierarchical levels in the encoder, meant to induce invariances at different feature levels. Recently, Eastwood et al. (2023) proposed a similar method to LooC (Xiao et al., 2021), but with strong theoretical guarantees.

3 IDENTIFIABLE ANISOTROPIC CL

3.1 Setup and Intuition

We argue that the augmentations usually employed in applied Contrastive Learning (CL) do not change all latents evenly. For example, color distortions do not drastically affect latents encoding semantic information or edges. We use a Latent Variable Model (LVM) to model CL, where a positive conditional $p(\boldsymbol{z}^+|\boldsymbol{z})$ relates the positive pairs $\boldsymbol{z}^+$ to the anchor sample $\boldsymbol{z}$. The augmentations used in practice induce a specific form of the positive conditional $p(\boldsymbol{z}^+|\boldsymbol{z})$, e.g., by assigning larger variance to latents strongly affected by augmentations. As the current CL theory does not account for augmentations that affect latents to different levels, we propose an anisotropic model for CL.

Following the practice of CL, we assume that the latent space is a $(d-1)$-dimensional hypersphere: $\mathcal{Z} = \mathbb{S}^{d-1}$. As proposed by Zimmermann et al. (2021), the anchor latent $\boldsymbol{z}$, and all negative pairs $\{\boldsymbol{z}_i^-\}$ are assumed to be uniformly distributed on $\mathcal{Z}$. Akin to Kirchhof et al. (2022), we allow non-uniform variances for the individual latent factors in the positive pair $\boldsymbol{z}^+$. We model this in the positive conditional by scaling the latent factors with *unknown* positive *concentration parameters*, collected in the diagonal matrix $\boldsymbol{\Lambda}$:

$$p(\boldsymbol{z}) = \text{const}, \quad p(\boldsymbol{z}^+|\boldsymbol{z}) \propto e^{-(\boldsymbol{z}^+ - \boldsymbol{z})^\top \boldsymbol{\Lambda} (\boldsymbol{z}^+ - \boldsymbol{z})}, \tag{2}$$

where a higher value of $\boldsymbol{\Lambda}_{ii}$ means a higher concentration of $\boldsymbol{z}^+$ around the anchor $\boldsymbol{z}$ along dimension i. Introducing $\boldsymbol{\Lambda}$ in the conditional generalizes previous works: $\boldsymbol{\Lambda} = \mathbf{I}_d$ recovers the von Mises-Fisher (vMF) conditional from Zimmermann et al. (2021), whereas two latent partitions, namely content (high $\boldsymbol{\Lambda}_{ii}$) and style (low $\boldsymbol{\Lambda}_{jj}$), yields the model of von Kügelgen et al. (2021). Our new conditional incorporates an entire spectrum of distributions between the two edge cases.

We assume that observations $\boldsymbol{x}, \boldsymbol{x}^+, \{\boldsymbol{x}_i^-\}$ are the result of passing the respective latents through an invertible generator $\boldsymbol{g} : \mathcal{Z} \to \mathcal{X} \subseteq \mathbb{R}^D$. Our goal is to train an encoder $\boldsymbol{f} : \mathbb{R}^D \to \mathcal{Z}$ such that the reconstructed latents, $\hat{\boldsymbol{z}} = \boldsymbol{f}(\boldsymbol{x})$, recover $\boldsymbol{z}$ up to permissible transformations. To train $\boldsymbol{f}$, we introduce our new loss:

$$\mathcal{L}_{\text{AINCE}}(\boldsymbol{f}, \hat{\boldsymbol{\Lambda}}) = \tag{3}$$
$$= \mathop{\mathbb{E}}_{\substack{\boldsymbol{x},\boldsymbol{x}^+ \\ \{\boldsymbol{x}_i^-\}}} \left[-\ln \frac{e^{-\left\|\boldsymbol{f}(\boldsymbol{x}^+)-\boldsymbol{f}(\boldsymbol{x})\right\|^2_{\hat{\boldsymbol{\Lambda}}}}}{e^{-\left\|f(\boldsymbol{x}^+)-\boldsymbol{f}(\boldsymbol{x})\right\|^2_{\hat{\boldsymbol{\Lambda}}}} + \sum_{i=1}^{M} e^{-\left\|\boldsymbol{f}(\boldsymbol{x}_i^-)-\boldsymbol{f}(\boldsymbol{x})\right\|^2_{\hat{\boldsymbol{\Lambda}}}}} \right],$$

where $\hat{\boldsymbol{\Lambda}}$ is a *trainable* diagonal scaling matrix, aimed to capture $\boldsymbol{\Lambda}$, and the weighted and squared ℓ_2-distance, $-\|\boldsymbol{f}(\boldsymbol{x}^+) - \boldsymbol{f}(\boldsymbol{x})\|^2_{\hat{\boldsymbol{\Lambda}}} = -(\boldsymbol{f}(\boldsymbol{x}^+) - \boldsymbol{f}(\boldsymbol{x}))^\top \hat{\boldsymbol{\Lambda}} (\boldsymbol{f}(\boldsymbol{x}^+) - \boldsymbol{f}(\boldsymbol{x}))$, is the *similarity function*. The idea behind AnInfoNCE is that the similarity function should be flexible enough to accommodate latents anisotropically perturbed by the augmentations. We defer the details to Appx. C and summarize our assumptions here:

Assumption 1 (Anisotropic CL on $\mathbb{S}^{d-1}$)**.** *The **DGP** satisfies (cf. Defn. C.1 for details):*

i) $\mathcal{Z} = \mathbb{S}^{d-1}$, $\mathcal{X} \subseteq \mathbb{R}^D$ are latent and observation spaces.
ii) Anchors are uniform on $\mathcal{Z}$, i.e., $p(\boldsymbol{z}) = \text{const.}$
iii) For $\boldsymbol{\Lambda} \in \mathbb{R}^{d \times d}$ PD diagonal matrix, the positive conditional $p(\boldsymbol{z}^+|\boldsymbol{z})$ has the form of (2).
iv) Negative pairs $\{\boldsymbol{z}_i^-\}$ are uniform on $\mathcal{Z}$.
v) A continuous, invertible generator function $\boldsymbol{g}$ maps $\boldsymbol{z}, \boldsymbol{z}^\pm \in \mathcal{Z} \overset{\boldsymbol{g}}{\mapsto} \boldsymbol{x}, \boldsymbol{x}^\pm \in \mathcal{X}$.

*The **model** satisfies (cf. Defn. C.2 for details):*

vi) The encoder $\boldsymbol{f} : \mathbb{R}^D \to \mathbb{S}^{d-1}$ is continuous.
vii) $\hat{\boldsymbol{\Lambda}} \in \mathbb{R}^{d \times d}$ is a PD diagonal matrix.
viii) $\boldsymbol{f}$ and $\hat{\boldsymbol{\Lambda}}$ are trained with $\mathcal{L}_{\text{AINCE}}$, i.e. (3).

Collectively i)-viii) define an anisotropic CL problem.

3.2 Identifiability of Anisotropic CL

When studying identifiability, we aim to recover the ground-truth latent $\boldsymbol{z}$ from observations $\boldsymbol{x} = \boldsymbol{g}(\boldsymbol{z})$ with as little ambiguity as possible. Our main contribution is proving the identifiability of the DGP from Assum. 1, generalizing Zimmermann et al. (2021); von Kügelgen et al. (2021):

Theorem 3.1 (Identifiability of Anisotropic CL)**.** *Under Assum. 1, if a pair $(\boldsymbol{f}, \hat{\boldsymbol{\Lambda}})$ minimizes $\mathcal{L}_{\text{AINCE}}$ (3), then $\boldsymbol{f} \circ \boldsymbol{g}$ is a block-orthogonal transformation, where each block acts on latents with equal weight $\boldsymbol{\Lambda}_{ii}$. In other words, $\boldsymbol{z}$ is identified up to a block-orthogonal transformation.*

Thm. 3.1 means that if the encoder $\boldsymbol{f}$, usually parametrized by a neural network, is expressive enough to minimize AnInfoNCE, then the reconstructed latent $\hat{\boldsymbol{z}} = \boldsymbol{f}(\boldsymbol{x})$ is related to the ground-truth $\boldsymbol{z}$ via a simple orthogonal linear transformation. This simple ambiguity remains inconsequential as solving any downstream task involves at least one linear layer trained on top of the backbone $\boldsymbol{f}$.

Proof Sketch (full proof in Appx. C.1). We rewrite AnInfoNCE into a categorical cross-entropy of predicting the correct index of the positive pair from randomly shuffled data $\boldsymbol{x}^+, \{\boldsymbol{x}_i^-\}$. The minimum loss is attained iff the similarity function matches the log-density-ratio between the positive and negative conditionals (Thm. B.1). This provides us with a functional equation that we solve for $(\boldsymbol{f}, \hat{\boldsymbol{\Lambda}})$. □

3.3 Extending Identifiability to Hard Negative Mining

Our generalized result in § 3.2 accounts for anisotropically varying latent factors. Here, we build on Thm. 3.1 to theoretically model the practice of hard negative (HN) mining.

HN mining effectively uses a negative conditional distribution to select negative samples more similar to the anchor, instead of sampling from a uniform marginal. Put differently, HN mining makes the classification problem of positive/negative samples *more difficult*, which is reasoned to help avoid latent collapse (Wu et al., 2020; Chuang et al., 2020; Robinson et al., 2021b; Zheng et al., 2021; Wang and Liu, 2021; Robinson et al., 2021a; Khosla et al., 2020; Yuksekgonul et al., 2023b). Our proposed anisotropic loss still identifies the ground-truth latents, when the negative conditional has the same form as (2) (we denote its concentration parameter as $\boldsymbol{\Lambda}^-$, and the one of the positive conditional as $\boldsymbol{\Lambda}^+$). Thus, we further generalize the setting of § 3.2, which is subsumed as $\boldsymbol{\Lambda}^- = \boldsymbol{0}$ (i.e., a uniform marginal). Intuitively, with a negative conditional, the density ratios will be the same as with a uniform marginal and a positive conditional with $\boldsymbol{\Lambda} = \boldsymbol{\Lambda}^+ - \boldsymbol{\Lambda}^-$.

Corollary 3.1 (Identifiability with HN mining. Proof in Appx. C.2)**.** *Under Assum. 1, but with a negative conditional in iv) as (2) with a concentration parameter $\boldsymbol{\Lambda}^-$, such that $\boldsymbol{\Lambda} = \boldsymbol{\Lambda}^+ - \boldsymbol{\Lambda}^-$ has full rank, the pair $(\boldsymbol{f}, \hat{\boldsymbol{\Lambda}})$ minimizing $\mathcal{L}_{\text{AINCE}}$ identifies the latent factors up to a block-orthogonal transformation.*

We include experimental results for hard negative mining in Appx. D.3. We empirically show that our theory correctly predicts the loss optimum at $\boldsymbol{\Lambda} = \boldsymbol{\Lambda}^+ - \boldsymbol{\Lambda}^-$, and that fine-tuning an encoder on a concatenation of hard negative samples and regular negative samples improves the identifiability score.

3.4 Extending Identifiability to Ensemble AnInfoNCE

Recent works started exploring both theoretically (Eastwood et al., 2023; Kirchhof et al., 2023) and empirically (Xiao et al., 2021; Zhang and Ma, 2022) the advantages of ensemble CL losses, where each loss component is calculated with different (potentially partially overlapping) data augmentations. We show that an ensemble version of AnInfoNCE is also identifiable. For more details, see Appx. C.3.

Corollary 3.2 (Identifia. of ensemble AnInfoNCE)**.** *Assume k DGPs, such that each satisfy Assum. 1. Assume that they share $\boldsymbol{g}$, but have different concentration parameters $\{\boldsymbol{\Lambda}^i\}$, for $1 \leq i \leq k$. Assume a shared encoder $\boldsymbol{f}$ and k learnable $\hat{\boldsymbol{\Lambda}}^i$ parameters. Then, the tuple $(\boldsymbol{f}, \{\hat{\boldsymbol{\Lambda}}^i\})$ minimizing the ensemble loss $\sum_i^k \mathcal{L}_{\text{AINCE}}(\boldsymbol{f}, \hat{\boldsymbol{\Lambda}}^i)$ identifies the latents up to a block-orthogonal transformation.*

Cor. 3.2 shows that using an ensemble with proper augmentations could lead to stronger identifiability guarantees (e.g., up to permutation, cf. Rem. C.4). Ideally, we could design such sets of augmentations, but it is not evident that this is possible in practice.

We include experimental results for ensemble AnInfoNCE in Appx. D.3. To verify Cor. 3.2, we define two DGPs that share a uniform anchor distribution and differ in their respective conditional distributions. We sample data from both DGPs, calculate the individual losses, and train the encoder on the sum of both losses. We see improved linear identifiability scores with ensemble AnInfoNCE, compared to training the encoder with InfoNCE or AnInfoNCE.

4 EXPERIMENTS

We now validate our theoretical results in experiments with increasing complexity. This section focuses on our main result and demonstrates the efficacy of AnInfoNCE in various settings. We include additional experimental ablations, results for hard negative mining and ensembling AnInfoNCE and detailed compute requirements for all our experiments in the Appendix.

4.1 Synthetic Experiments

Preliminaries. We follow Zimmermann et al. (2021) and consider a fully-controlled Data Generating Process (DGP), which implements Assum. 1. That is, the ground-truth latent space is a hypersphere ($\mathcal{Z} = \mathbb{S}^{d-1}$), and positive/negative pairs are defined through conditional sampling in the latent space. We model the generative process $\boldsymbol{g}$ as an invertible Multi-Layer Perceptron (MLP) with the same input, intermediate, and output dimensionality d and with Leaky ReLU non-linearities. We set $d = 10$. The encoder $\boldsymbol{f}$ is also parameterized by an MLP with the same input, intermediate, and output dimensionality, and with Leaky ReLU non-linearities. The encoder is trained only with observations, which are obtained by passing $\boldsymbol{z}$ through $\boldsymbol{g}$. As we have access to the ground-truth $\boldsymbol{z}$, we can post-hoc compute the information captured in the inferred latents $\hat{\boldsymbol{z}}$ about the ground-truth $\boldsymbol{z}$. We do this by fitting a linear map $\mathbf{A}$ between $\hat{\boldsymbol{z}}$ and $\boldsymbol{z}$ and compute the average coefficient of determination R^2 Wright (1921) between the relevant features (e.g. all of them, content-only dimensions, style-only dimensions) of $\mathbf{A}\hat{\boldsymbol{z}}$ and $\boldsymbol{z}$. Unless noted otherwise, we use a uniform marginal and a projected Gaussian as the conditional distribution, modeling Eq. (2) and the assumptions in § 3.1.

Results. We study the effect of anisotropy in the positive conditional by varying the ground-truth $\boldsymbol{\Lambda}$: We keep the value of five latent dimensions fixed at $\Lambda_1 = 5$, while we vary the value of the other five dimensions (i.e., Λ_2), resulting in a ground-truth diagonal $\boldsymbol{\Lambda} = \text{diag}\,(\Lambda_1, \dots, \Lambda_1, \Lambda_2, \dots \Lambda_2)$. While von Kügelgen et al. (2021) coined the terms "content" and "style" to refer to invariant and variant latents, respectively, we here relax this notion and use these terms comparatively: In an anisotropic setting, where $\boldsymbol{\Lambda}$ is composed of different values, higher (lower) values reflect more content-like (style-like) dimensions. We train an encoder on the observed data generated by the process described above and summarize the results in Fig. 2**A**. We observe high R^2 scores over a wide range of $\boldsymbol{\Lambda}$-values for both content and style latents when training with our AnInfoNCE loss (markers in Fig. 2**A** indicate converged training runs). In contrast, regular InfoNCE loss cannot identify the style latents ($R^2 \approx 0$). We also analyze the setting where the conditionals of five latents are distributed according to $\boldsymbol{\Lambda}_2 = 25$ in more detail in Fig. 2**B-D**. We observe that content dimensions are learned before style (Fig. 2**B**). Our loss converges to the global minimum value, which can be computed based on the ground-truth latents (Fig. 2**C**). When learning $\boldsymbol{\Lambda}$, we observe that the learned $\hat{\boldsymbol{\Lambda}}$ values almost match the ground-truth values (dashed black lines in Fig. 2**D**).

4.2 VAE Experiments on MNIST

Preliminaries. We now move closer to actual images by replacing the MLP in our generative model with a ten-dimensional Variational Autoencoder (VAE; Kingma and Welling, 2013) trained on MNIST (Deng, 2012). This setup is beneficial as we have valid images while fully controlling the DGP. Thus, we still sample ground-truth latents, which we then pass through the VAE decoder to generate observations to train on. Both the encoder and the decoder are fully connected three-layer MLPs with Leaky ReLU activations. Generated samples for different concentration parameters $\boldsymbol{\Lambda}$ are shown in Appx. D.5, Fig. 13.

Results. We start with the same anisotropic ground-truth $\boldsymbol{\Lambda}$ as in § 4.1. When training an encoder on MNIST samples with AnInfoNCE, we observe perfect linear identifiability scores (R^2) for all latents, whereas training with the standard InfoNCE loss yields zero R^2 scores for the style latents (Fig. 3**A**). The identifiability scores correlate with KNN classification accuracy,

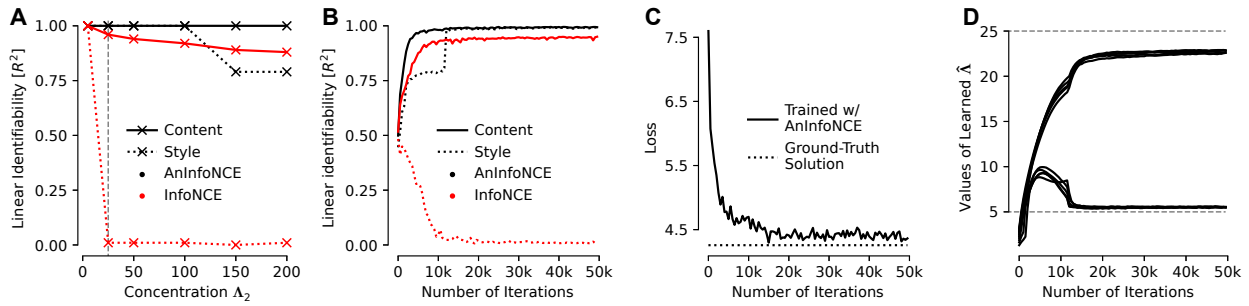


Figure 2: **Behavior of AnInfoNCE on Synthetic Data.** ($\Lambda_1 = 5$) **A:** We maintain high R^2-scores with AnInfoNCE for both content and style dimensions, while the style dimensions are lost when training with the regular InfoNCE loss. For $\Lambda_2 = 25$ (dotted black vertical line), we show: **B:** The evolution of the linear R^2 scores during training; **C:** AnInfoNCE reaches the global minimum, computed based on ground-truth latents; **D:** The evolution of the learned $\hat{\mathbf{\Lambda}}$ values.

which is stable across a wide range of $\mathbf{\Lambda}$-values when training with AnInfoNCE, but degrades when training with InfoNCE. As our theoretical claims are only valid for the representations after the final layer of the model, we evaluate those. This means we deviate from previous literature and do not use a projection head.

Next, we model each latent with a distinct concentration parameter $\mathbf{\Lambda}_{ii}$ by linearly sampling values between 5 and a maximum of either 50 or 200. When training the encoder with AnInfoNCE, we recover all latent dimensions: The linear identifiability scores are 99.7% and 99.6%, respectively. However, training with the regular InfoNCE loss leads to substantially lower scores of 77.4% and 76.3%, respectively. Fig. 3**C & D** show how the learned $\hat{\mathbf{\Lambda}}$ values change during training. We observe that the learned $\hat{\mathbf{\Lambda}}$ approaches the ground truth, except for very high $\mathbf{\Lambda}$ values: This is visible in Fig. 3**D** when $\mathbf{\Lambda}_{ii} > 100$). We analyze this behavior further in Appx. D.

4.3 Real-world Experiments

Preliminaries. On CIFAR10 (Krizhevsky, 2009), we train ResNet18 (He et al., 2016) models for 1000 epochs with the code from Hua (2021), while on ImageNet (Russakovsky et al., 2015), we train ResNet50 models for 100 epochs using the code from Facebook Research (2022).

Results. On real-world datasets, we do not have access to true generative factors and the latent conditional distribution. Therefore, we need to create views using regular SimCLR augmentations. Another consequence is that we cannot directly measure how well models recover the ground-truth information. Thus, we use proxy evaluations and calculate the linear readout accuracy on the augmentations used during training to judge how well we recover the latent factors. Chance-level performance is 50%. Our experiments on CIFAR10 and ImageNet show that our loss with a trainable $\hat{\mathbf{\Lambda}}$ leads to much higher readout accuracy on the used augmentations, i.e., we successfully recover more latent dimensions compared to the regular InfoNCE loss (Tab. 1). While we reduce the information loss, this surprisingly does not lead to better downstream accuracy (Tab. 1, *Classes* column). Removing ℓ_2-normalization results in an encoder with the highest augmentations readout accuracy and simultaneously the lowest classification accuracy. We analyze this discrepancy, the validity of our assumptions for real-world data, and the required steps for closing the remaining gap between CL theory and practice in § 5.

5 ANALYSIS

Our experiments (§ 4) show some scenarios in which recovering more (latent) information results in better downstream performance (e.g., readout accuracy on MNIST). While AnInfoNCE outperforms InfoNCE in these controlled scenarios, it fails in others: On CIFAR10 and ImageNet, we observe a trade-off between the augmentation readout and linear classification readout accuracies. While higher accuracy in augmentation readout indicates a better capture of style latents (i.e., more latent information is recovered), it does not coincide with higher classification accuracy. Scaling up synthetic experiments to ImageNet comes with changes to various training aspects: The encoder's architecture is switched to a ResNet from an invertible MLP, views are sampled using augmentations instead of the ground-truth DGP, and the encoder's output dimensionality is orders of magnitude higher. To control for these differences, we conduct a minimal-change experiment on MNIST, changing only how the views are generated while the encoder and the training procedure remain the same. We demonstrate that using augmentations

P3: InfoNCE: Identifying the Gap between Theory and Practice.

Rusak*, Reizinger*, Juhos*, Bringmann, Zimmermann†, Brendel†

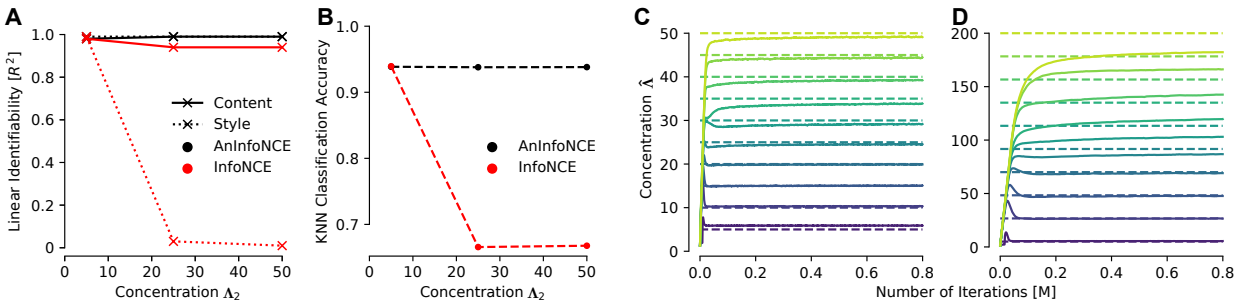


Figure 3: **Behavior of AnInfoNCE on MNIST. A:** Linear identifiability (R^2) scores for an encoder trained on VAE-generated MNIST samples, when varying Λ_2 and keeping $\Lambda_1 = 5$ for the positive conditional. Style dimensions collapse ($R^2 = 0$) for regular InfoNCE. **B:** KNN-accuracy evaluated on the regular MNIST dataset using the encoders trained as described in **A**. KNN accuracy degrades when style dimensions are lost. **C & D:** The evolution of learned $\hat{\mathbf{\Lambda}}$ during training. We set the diagonal entries of $\mathbf{\Lambda}$ to ten different values by linearly interpolating between 5 and 50 (**C**) or 5 and 200 (**D**). The ground-truth $\mathbf{\Lambda}$ values are indicated by dashed lines. The learned $\hat{\mathbf{\Lambda}}$ are shown in the corresponding colors as solid lines.

Table 1: **Better Augmentations Readout Does Not Imply Better Classification Performance.** Comparison of different training objectives for linear downstream classification (*Classes*) and augmentation readout (*Grayscale* to *Saturate*, *Avg.* denotes the average over those) performance. Learning $\mathbf{\Lambda}$ results in higher readout accuracy of the used augmentations, which can be interpreted as a better recovery of style latents. However, better augmentations readout does not result in better linear classification accuracy. We note that our theory and all synthetic and VAE experiments apply to the post-projector (backb.+proj.) features which are typically not used in practice because using backbone features results in better performance.

	Training Objective	Features	Classes	Grayscale	Brightness	Contrast	Hue	Saturate	Avg.
CIFAR10	InfoNCE	backbone	90.9	56.1	59.3	62.6	55.4	55.1	57.7
	AnInfoNCE	backbone	88.4	80.1	93.4	91.1	65.5	72.3	80.5
	AnInfoNCE, w/o ℓ_2-norm.	backbone	83.2	96.2	98.0	96.6	76.8	88.4	91.2
	InfoNCE	backb.+proj.	89.8	52.0	52.0	52.0	51.8	51.0	51.8
	AnInfoNCE	backb.+proj.	85.4	71.4	71.4	77.1	59.7	65.5	69.0
	AnInfoNCE, w/o ℓ_2-norm.	backb.+proj.	79.1	65.5	65.5	87.6	70.5	83.8	74.6
ImageNet	InfoNCE	backbone	68.2	98.7	71.6	72.5	68.0	61.2	74.4
	AnInfoNCE	backbone	59.0	98.8	81.5	82.3	71.6	62.5	79.3
	InfoNCE	backb.+proj.	57.7	58.6	51.9	51.9	51.5	51.1	53.0
	AnInfoNCE	backb.+proj.	26.7	71.0	55.2	54.1	52.6	52.7	57.1

instead of generating the views using the ground-truth DGP is the main reason for this trade-off.

Minimal-change experiment on MNIST exhibits the same identifiability-accuracy trade-off. To visualize how augmentations influence the positive conditional distribution in latent space, we train a VAE on augmented MNIST images and then project these augmented images onto the learned VAE's latent space. We observe that the latents of positive pairs do not follow a projected Gaussian distribution and can even follow a bimodal one (see Fig. 7 in Appx. D). Our theory does not cover this case; neither do other theories we are aware of, and so this mismatch presents a fruitful direction for future research on closing the gap between theory and practice.

Further, to better analyze the interplay between augmentations and disentanglement, we train a model either using InfoNCE or AnInfoNCE on MNIST when views are generated using SimCLR augmentations (except color augmentations as MNIST is grayscale):

Loss	Class	Lin. id. [R^2]	Crop	Gaussian Blur
InfoNCE	**0.97**	0.34	0.53	0.58
AnInfoNCE	0.94	**0.44**	**0.59**	**0.63**

We observe the same accuracy-identifiability trade-off on MNIST as before on ImageNet and CIFAR10: While the augmentations' prediction accuracy improves, the classification accuracy decreases. Using the VAE trained on MNIST, we can also calculate linear identifiability scores, which are higher for AnInfoNCE (but

Table 2: **We Observe the Accuracy-Identifiability Trade-Off on C-3DIdent.** When using augmentations to generate views (Aug.), InfoNCE achieves better accuracy, while AnInfoNCE has higher R^2 scores. When using the ground truth DGP to generate views (DGP), AnInfoNCE outperforms InfoNCE in terms of both accuracy and identifiability scores. We report averaged results over three random seeds and the standard deviation.

	Data Augmentation			DGP		
Loss	Class	Lin. id. [R^2]	Nonlin. id. [R^2]	Class	Lin. id. [R^2]	Nonlin. id. [R^2]
InfoNCE	**1.0**$_{\pm 0.0}$	0.31$_{\pm 0.0}$	0.51$_{\pm 0.0}$	1.0$_{\pm 0.0}$	0.2$_{\pm 0.0}$	0.24$_{\pm 0.005}$
AnInfoNCE	0.84$_{\pm 0.012}$	**0.34**$_{\pm 0.008}$	**0.65**$_{\pm 0.0}$	**1.0**$_{\pm 0.0}$	**0.38**$_{\pm 0.017}$	**0.94**$_{\pm 0.0}$

still moderately low). Note that training with AnInfoNCE on MNIST when views are generated by passing ground-truth latents through the VAE leads to perfect linear identifiability scores ($R^2 = 1$). In that case, we do not observe a trade-off between linear identifiability scores and downstream accuracy.

We observe the accuracy-identifiability trade-off on the Causal 3DIdent dataset. The Causal 3DIdent dataset (C-3DIdent, von Kügelgen et al., 2021; Zimmermann et al., 2021) is a computer-generated image dataset with seven classes and ten ground-truth latent factors, rendered with Blender (Blender Online Community, 2018). This dataset is more complex than MNIST and thus constitutes an interesting intermediate step between MNIST and visually complex datasets such as ImageNet.

We repeat our analysis from above and train a ResNet18 encoder with InfoNCE and AnInfoNCE on C-3DIdent when views are generated using both the ground truth DGP and data augmentation. In addition to the linear identifiability scores, we also report nonlinear identifiability scores calculated using a Kernel Ridge Regression, following von Kügelgen et al. (2021). In Tab. 2, we observe the same accuracy-identifiability trade-off as on the other datasets when using augmentations: Training with AnInfoNCE leads to better identifiability scores as predicted by our theory. However, training with InfoNCE results in better downstream accuracy. When the ground-truth generative model (DGP) is used for generating the views, AnInfoNCE outperforms InfoNCE in both identifiability and accuracy. Training details can be found in Appx. D.6.

The accuracy-identifiability trade-off can be mitigated when using only one augmentation. To better understand the accuracy-identifiability trade-off, we train models with InfoNCE and AnInfoNCE on Causal 3DIdent using a single augmentation to generate the views. We generate views using small crops (scale parameter between 0.08 and 1.0), large crops (scale parameter between 0.8 and 1.0), by applying color jitter or rotations. We further repeat these experiments using the DGP to generate views by modifying the corresponding latents. We report the results in Table 3.

We observe that modeling singular data augmentations, e.g., only applying crops or color jitter, works very well with AnInfoNCE, and we see no trade-off between classification accuracy and identifiability. For all augmentations we tested, AnInfoNCE is always better at recovering them than InfoNCE. However, the classification accuracy is generally not very high when using only one augmentation type, except for color jitter where the classification accuracy is 98% with AnInfoNCE. This observation does not hold on real-world data and demonstrates the limitations of the Causal 3D-Ident dataset: When using only color jitter for generating positive samples and training the encoder on CIFAR10, the downstream accuracy drops from 90.9% to 48.6%. While combining multiple augmentations results in high classification accuracy, it also leads to the accuracy-identifiability trade-off as shown in Table 2. We hypothesize that modeling two augmentations is more challenging and requires extending our simple diagonal matrix to a more complex function. This likely would make deriving theoretical guarantees more challenging. When using transformation in the latent codes, AnInfoNCE is always superior to InfoNCE, irrespective of which transformation has been used.

We conclude that AnInfoNCE performs better than InfoNCE on all singular data augmentations we tested and is not sensitive to any of them. However, it struggles when *multiple* augmentations are combined, which is necessary for good downstream accuracy on real-world datasets.

Further ablation studies. We analyze the learning dynamics of AnInfoNCE and conduct a thorough hyperparameter study in Appx. D. We observe that the performance of AnInfoNCE is strongly influenced by sample efficiency, i.e., the batch size and the latent dimensionality. Further, we observe that larger concentration parameters also require larger batch sizes. We link this observation to the role of the *augmentation overlap* (Wang et al., 2022): Higher $\mathbf{\Lambda}$ means a more concentrated conditional and requires larger batch sizes for sufficient augmentation overlap. Investigating the

Table 3: **There is no accuracy-identifiability trade-off, if a single augmentation is used for training.** On Causal 3DIdent, classification and identifiability results improve simultaneously when switching from InfoNCE to AnInfoNCE, if using a single data augmentation to generate views (**DA**). The same is observed when perturbing single components in the latent code (**DGP**). This relative advantage of AnInfoNCE compared to InfoNCE was unlocked by giving up sophisticated augmentation pipelines and, hence, absolute performance.

	DA: Small Crops			DA: Large Crops			DGP: Change Position		
Loss	Class	Lin. id. [R^2]	Nonlin. id. [R^2]	Class	Lin. id. [R^2]	Nonlin. id. [R^2]	Class	Lin. id. [R^2]	Nonlin. id. [R^2]
InfoNCE	0.14	0.02	0.10	0.26	0.07	0.23	**1.0**	0.25	0.28
AnInfoNCE	**0.37**	**0.17**	**0.44**	**0.47**	**0.18**	**0.41**	**1.0**	**0.31**	**0.91**

	DA: Rotations			DGP: Rotations		
Loss	Class	Lin. id. [R^2]	Nonlin. id. [R^2]	Class	Lin. id. [R^2]	Nonlin. id. [R^2]
InfoNCE	0.29	0.08	0.30	**1.0**	0.33	0.45
AnInfoNCE	**0.48**	**0.15**	**0.31**	**1.0**	**0.57**	**0.86**

	DA: Apply ColorJitter			DGP: Change Hues		
Loss	Class	Lin. id. [R^2]	Nonlin. id. [R^2]	Class	Lin. id. [R^2]	Nonlin. id. [R^2]
InfoNCE	0.90	0.22	0.41	**1.0**	0.31	0.36
AnInfoNCE	**0.98**	**0.56**	**0.85**	**1.0**	**0.44**	**0.85**

evolution of the identifiability scores over time, we observe a step-like behavior in how the latents are learned, corroborating the results by Simon et al. (2023). We discuss connections between AnInfoNCE and TriCL (Zhang et al., 2023) and show how the losses compare experimentally in Appx. D.4.

Discussion. Given our analysis above, we identify the use of augmentations in real-world datasets for generating views as the main cause for the accuracy-identifiability trade-off: Assuming an underlying LVM, these augmentations correspond to changes in latent factors, modeled by the positive conditional distribution. This points to a potential cause for unexpectedly low performance: The conditional distribution implied by augmentations may not correspond to the one assumed by the loss. First, the form of the conditional implicitly defined by the augmentations may not be Gaussian or even uni-modal. Second, the corresponding concentration parameters may be high, leading to a flat loss landscape and optimization issues in practice.

6 CONCLUSION

Our work advances CL theory by generalizing InfoNCE. The proposed framework, AnInfoNCE-based Contrastive Learning (CL), is identifiable and relies on more realistic assumptions on the data, reducing the gap between CL theory and practice: It can account for anisotropic latent conditionals that more likely reflect how augmentations affect the latent factors in practice. In addition, we showed extensions of our results towards other empirical techniques such as hard negative mining and loss ensembling. We demonstrated that our framework accounts for certain known limitations of CL, such as collapsing information, by better capturing style information. At the same time, we showed that there exists a trade-off between recovered style information and downstream classification. Further, even our relaxed assumptions can still be too strong in real-world problems: For example, the positive conditional distribution can be non-Gaussian.

Exploring the gap between theory and practice led to a new loss formulation. We hope that further investigations into overcoming the remaining gap between theory and practice will result in refinements and insights that can improve practical self-supervised learning techniques and our understanding of them. We discussed some directions in § 5 and Appx. D, suggesting that more theoretical work on the identifiability of large-scale DGPs is a fruitful endeavor for future work.

Author Contributions

WB conceived the project idea. ER led the project. ER, RSZ, and WB designed the experiments and ER executed them. ER conducted the analysis with support from RSZ and PR. ER and RSZ jointly created all figures. PR conceived how to incorporate anisotropy in SimCLR. This idea led to WB and RSZ developing the main theorem. AJ developed corollaries for hard negative mining and loss ensembling with help from PR and feedback from RSZ. RSZ and WB jointly supervised the project. AJ was responsible for presenting the theoretical results in a standardized way. ER, PR, AJ, RSZ, and WB contributed to writing the manuscript.

Acknowledgements

The authors thank Julian Bitterwolf, Jack Brady, Steffen Schneider, Thaddäus Wiedemer, and Zac Cranko for helpful discussions. We would also like to thank Mihály Weiner for all the insights that aided us in filling edge cases of our proof. The authors thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting Evgenia Rusak, Patrik Reizinger, Attila Juhos, and Roland S. Zimmermann. Patrik Reizinger acknowledges his membership in the European Laboratory for Learning and Intelligent Systems (ELLIS) PhD program. This work was supported by the German Federal Ministry of Education and Research (BMBF): Tübingen AI Center, FKZ: 01IS18039A. WB acknowledges financial support via an Emmy Noether Grant funded by the German Research Foundation (DFG) under grant no. BR 6382/1-1 and via the Open Philantropy Foundation funded by the Good Ventures Foundation. WB is a member of the Machine Learning Cluster of Excellence, EXC number 2064/1 – Project number 390727645. This research utilized compute resources at the Tübingen Machine Learning Cloud, DFG FKZ INST 37/1057-1 FUGG.

References

Assran, M., Balestriero, R., Duval, Q., Bordes, F., Misra, I., Bojanowski, P., Vincent, P., Rabbat, M. G., and Ballas, N. (2023). The hidden uniform cluster prior in self-supervised learning. In *ICLR*. OpenReview.net. Cited on page 3.

Ba, J. L., Kiros, J. R., and Hinton, G. E. (2016). Layer normalization. *arXiv preprint*. Cited on page 30.

Balestriero, R. and LeCun, Y. (2022). Contrastive and non-contrastive self-supervised learning recover global and local spectral embedding methods. In *NeurIPS*. Cited on page 3.

Bendidi, I., Bardes, A., Cohen, E., Lamiable, A., Bollot, G., and Genovesio, A. (2023). No Free Lunch in Self Supervised Representation Learning. *arXiv preprint*. Cited on pages 3 and 30.

Blender Online Community (2018). *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam. Cited on page 8.

Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., and Joulin, A. (2020). Unsupervised Learning of Visual Features by Contrasting Cluster Assignments. In *NeurIPS*. Cited on page 1.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. E. (2020a). A Simple Framework for Contrastive Learning of Visual Representations. In *ICML*. PMLR. Cited on pages 1, 3, and 24.

Chen, T., Kornblith, S., Swersky, K., Norouzi, M., and Hinton, G. E. (2020b). Big Self-Supervised Models are Strong Semi-Supervised Learners. *Advances in neural information processing systems*. Cited on page 1.

Chen, T., Luo, C., and Li, L. (2021). Intriguing properties of contrastive losses. In *NeurIPS*. Cited on pages 3 and 16.

Chen, X., Fan, H., Girshick, R., and He, K. (2020c). Improved Baselines with Momentum Contrastive Learning. *arXiv preprint*. Cited on page 1.

Chen, X. and He, K. (2021). Exploring simple siamese representation learning. In *CVPR*. Computer Vision Foundation / IEEE. Cited on pages 1 and 30.

Chuang, C., Hjelm, R. D., Wang, X., Vineet, V., Joshi, N., Torralba, A., Jegelka, S., and Song, Y. (2022). Robust contrastive learning against noisy views. In *CVPR*. IEEE. Cited on page 31.

Chuang, C., Robinson, J., Lin, Y., Torralba, A., and Jegelka, S. (2020). Debiased contrastive learning. In *NeurIPS*. Cited on pages 4 and 31.

Cosentino, R., Sengupta, A., Avestimehr, S., Soltanolkotabi, M., Ortega, A., Willke, T., and Tepper, M. (2022). Toward a Geometrical Understanding of Self-supervised Contrastive Learning. *arXiv preprint*. Cited on page 3.

Cui, J., Huang, W., Wang, Y., and Wang, Y. (2022). AggNCE: Asymptotically Identifiable Contrastive Learning. In *NeurIPS 2022 Workshop: Self-Supervised Learning - Theory and Practice*. Cited on pages 3 and 29.

Dangovski, R., Jing, L., Loh, C., Han, S., Srivastava, A., Cheung, B., Agrawal, P., and Soljačić, M. (2021). Equivariant Contrastive Learning. *arXiv preprint*. Cited on page 3.

Daunhawer, I., Bizeul, A., Palumbo, E., Marx, A., and Vogt, J. E. (2023). Identifiability results for multimodal contrastive learning. In *ICLR*. OpenReview.net. Cited on page 29.

Deng, L. (2012). The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, (6). Cited on page 5.

Dubois, Y., Ermon, S., Hashimoto, T. B., and Liang, P. (2022). Improving self-supervised learning by characterizing idealized representations. In *NeurIPS*. Cited on pages 3, 16, and 29.

Eastwood, C., von Kügelgen, J., Ericsson, L., Bouchacourt, D., Vincent, P., Schölkopf, B., and Ibrahim, M. (2023). Self-Supervised Disentanglement by Leveraging Structure in Data Augmentations. *arXiv preprint*. Cited on pages 3, 4, 23, and 29.

Ericsson, L., Gouk, H., and Hospedales, T. M. (2021). Why Do Self-Supervised Models Transfer? Investigating the Impact of Invariance on Downstream Tasks. *arXiv preprint*. Cited on page 30.

Ermolov, A., Siarohin, A., Sangineto, E., and Sebe, N. (2021). Whitening for self-supervised representation learning. In *ICML*. PMLR. Cited on page 31.

Facebook Research (2022). Vicreg: Variance-invariance-covariance regularization for self-supervised learning. `https://github.com/facebookresearch/vicreg`. Cited on page 6.

FOLDING.LAR.SYSTEMS (Accessed on May 14th, 2024). Gpu folding overview. `https://folding.lar.systems/gpu_ppd/brands/nvidia/folding_profile/ga100_a100_pcie_40gb`. Cited on page 29.

Garrido, Q., Chen, Y., Bardes, A., Najman, L., and LeCun, Y. (2023). On the duality between contrastive and non-contrastive self-supervised learning. In *ICLR*. OpenReview.net. Cited on page 3.

Gresele, L., Rubenstein, P. K., Mehrjou, A., Locatello, F., and Schölkopf, B. (2019). The incomplete rosetta stone problem: Identifiability results for multi-view nonlinear ICA. In *UAI*. AUAI Press. Cited on page 23.

Grill, J., Strub, F., Altché, F., Tallec, C., Richemond, P. H., Buchatskaya, E., Doersch, C., Pires, B. Á., Guo, Z., Azar, M. G., Piot, B., Kavukcuoglu, K., Munos, R., and Valko, M. (2020). Bootstrap your own latent - A new approach to self-supervised learning. In *NeurIPS*. Cited on page 1.

Gutmann, M. and Hyvärinen, A. (2010). Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *AISTATS*. JMLR.org. Cited on pages 2 and 17.

Halvagal, M. S., Laborieux, A., and Zenke, F. (2022). An eigenspace view reveals how predictor networks and stop-grads provide implicit variance regularization. Cited on page 31.

HaoChen, J. Z., Wei, C., Gaidon, A., and Ma, T. (2021). Provable guarantees for self-supervised deep learning with spectral contrastive loss. In *NeurIPS*. Cited on page 3.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *CVPR*. IEEE Computer Society. Cited on page 6.

Hua, P. (2021). Simsiam: A pytorch implementation. `https://github.com/PatrickHua/SimSiam`. Cited on page 6.

Hyvärinen, A. and Morioka, H. (2016). Unsupervised feature extraction by time-contrastive learning and nonlinear ICA. In *NeurIPS*. Cited on page 23.

Jing, L., Vincent, P., LeCun, Y., and Tian, Y. (2022). Understanding dimensional collapse in contrastive self-supervised learning. In *ICLR*. OpenReview.net. Cited on pages 1, 3, and 16.

Khosla, P., Teterwak, P., Wang, C., Sarna, A., Tian, Y., Isola, P., Maschinot, A., Liu, C., and Krishnan, D. (2020). Supervised contrastive learning. In *NeurIPS*. Cited on page 4.

Kingma, D. P. and Welling, M. (2013). Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*. Cited on page 5.

Kirchhof, M., Kasneci, E., and Oh, S. J. (2023). Probabilistic contrastive learning recovers the correct aleatoric uncertainty of ambiguous inputs. In *ICML*. PMLR. Cited on page 4.

Kirchhof, M., Roth, K., Akata, Z., and Kasneci, E. (2022). A non-isotropic probabilistic take on proxy-based deep metric learning. In *ECCV (26)*. Springer. Cited on page 3.

Krizhevsky, A. (2009). Learning multiple layers of features from tiny images. *Master's thesis, University of Toronto*. Cited on page 6.

Lachapelle, S., Deleu, T., Mahajan, D., Mitliagkas, I., Bengio, Y., Lacoste-Julien, S., and Bertrand, Q. (2022). Synergies Between Disentanglement and Sparsity: a Multi-Task Learning Perspective. *arXiv preprint*. Cited on page 29.

Lee, H., Lee, K., Lee, K., Lee, H., and Shin, J. (2021a). Improving transferability of representations via augmentation-aware self-supervision. In *NeurIPS*. Cited on page 3.

Lee, J. D., Lei, Q., Saunshi, N., and Zhuo, J. (2021b). Predicting what you already know helps: Provable self-supervised learning. In *NeurIPS*. Cited on page 29.

Liu, B., Rosenfeld, E., Ravikumar, P. K., and Risteski, A. (2022). Analyzing and improving the optimization landscape of noise-contrastive estimation. In *ICLR*. OpenReview.net. Cited on page 26.

Liu, B. and Wang, B. (2023). Bayesian Self-Supervised Contrastive Learning. *arXiv preprint*. Cited on page 31.

Lyu, Q., Fu, X., Wang, W., and Lu, S. (2021). Latent Correlation-Based Multiview Learning and Self-Supervision: A Unifying Perspective. *arXiv preprint*. Cited on pages 3 and 29.

Ma, Z. and Collins, M. (2018). Noise Contrastive Estimation and Negative Sampling for Conditional Models: Consistency and Statistical Efficiency. *arXiv preprint arXiv:1809.01812*. Cited on page 17.

Marcel, S. and Rodriguez, Y. (2010). Torchvision the machine-vision package of torch. In *ACM International Conference on Multimedia*. Cited on page 29.

Matthes, S., Han, Z., and Shen, H. (2023). Towards a Unified Framework of Contrastive Learning for Disentangled Representations. In *Thirty-seventh Conference on Neural Information Processing Systems*. Cited on page 17.

Merkel, D. (2014). Docker: Lightweight linux containers for consistent development and deployment. *Linux J.*, 2014(239). Cited on page 29.

Oord, A. v. d., Li, Y., and Vinyals, O. (2018). Representation Learning with Contrastive Predictive Coding. *arXiv preprint*. Cited on pages 1 and 2.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. (2017). Automatic differentiation in PyTorch. In *NIPS Autodiff Workshop*. Cited on page 29.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. (2021). Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR. Cited on page 2.

Rahmanzadehgervi, P., Bolton, L., Taesiri, M. R., and Nguyen, A. T. (2024). Vision language models are blind. *arXiv preprint arXiv:2407.06581*. Cited on page 2.

Rajendran, G., Reizinger, P., Brendel, W., and Ravikumar, P. (2023). An interventional perspective on identifiability in gaussian lti systems with independent component analysis. *arXiv preprint*. Cited on page 23.

Rensmart (Accessed on May 14th, 2024). Carbon emissions calculator. `https://www.rensmart.com/Calculators/KWH-to-CO2`. Cited on page 29.

Robinson, J., Sun, L., Yu, K., Batmanghelich, K., Jegelka, S., and Sra, S. (2021a). Can contrastive learning avoid shortcut solutions? In *NeurIPS*. Cited on pages 4 and 31.

Robinson, J. D., Chuang, C., Sra, S., and Jegelka, S. (2021b). Contrastive learning with hard negative samples. In *ICLR*. OpenReview.net. Cited on pages 4 and 31.

Roeder, G., Metz, L., and Kingma, D. (2021). On linear identifiability of learned representations. In *ICML*. PMLR. Cited on page 27.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, (3). Cited on page 6.

Saunshi, N., Ash, J., Goel, S., Misra, D., Zhang, C., Arora, S., Kakade, S., and Krishnamurthy, A. (2022). Understanding contrastive learning requires incorporating inductive biases. In *ICML*. PMLR. Cited on page 26.

Saunshi, N., Plevrakis, O., Arora, S., Khodak, M., and Khandeparkar, H. (2019). A theoretical analysis of contrastive unsupervised representation learning. In *ICML*. PMLR. Cited on pages 29 and 31.

Saxe, A. M., McClelland, J. L., and Ganguli, S. (2014). Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. In *ICLR*. Cited on page 26.

Simon, J. B., Knutins, M., Ziyin, L., Geisz, D., Fetterman, A. J., and Albrecht, J. (2023). On the stepwise nature of self-supervised learning. In *ICML*. Cited on pages 9 and 26.

Sohn, K. (2016). Improved Deep Metric Learning with Multi-class N-pair Loss Objective. *Advances in neural information processing systems*, 29. Cited on pages 2 and 17.

Sylabs Inc (2018). Singularity 2.5.2 - linux application and environment containers for science. `https://doi.org/10.5281/zenodo.1308868`. Cited on page 29.

Tamkin, A. and He, X. (2022). Feature Dropout: Revisiting the Role of Augmentations in Contrastive Learning. In *NeurIPS 2022 Workshop: Self-Supervised Learning - Theory and Practice*. Cited on page 16.

Techpowerup (Accessed on May 14th, 2024). Gpu specs database. `https://www.techpowerup.com/gpu-specs`. Cited on page 29.

Tian, Y. (2023). Understanding the role of nonlinearity in training dynamics of contrastive learning. In *ICLR*. OpenReview.net. Cited on pages 16 and 30.

Travel Navigator (Accessed on May 14th, 2024). https://travelnav.com/emissions-from- new-york-ny-to-london-united-kingdom. Cited on page 29.

Trusca, M. M., Nuyts, W., Thomm, J., Honig, R., Hofmann, T., Tuytelaars, T., and Moens, M.-F. (2024). Object-attribute binding in text-to-image generation: Evaluation and control. *arXiv preprint arXiv:2404.13766*. Cited on page 2.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Jarrod Millman, K., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C., Polat, İ., Feng, Y., Moore, E. W., Vand erPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., and Contributors, S. . . (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272. Cited on page 29.

von Kügelgen, J., Rubenstein, P. K., Schölkopf, B., and Weller, A. (2019). Optimal experimental design via bayesian optimization: active causal structure learning for gaussian process networks. *arXiv preprint*. Cited on page 3.

von Kügelgen, J., Sharma, Y., Gresele, L., Brendel, W., Schölkopf, B., Besserve, M., and Locatello, F. (2021). Self-supervised learning with data augmentations provably isolates content from style. In *NeurIPS*. Cited on pages 1, 2, 3, 4, 5, 8, 16, 27, 28, and 29.

Wagner, D., Ferreira, F., Stoll, D., Schirrmeister, R. T., Müller, S., and Hutter, F. (2022). On the Importance of Hyperparameters and Data Augmentation for Self-Supervised Learning. *arXiv preprint*. Cited on pages 3 and 30.

Wang, F. and Liu, H. (2021). Understanding the behaviour of contrastive loss. In *CVPR*. Computer Vision Foundation / IEEE. Cited on page 4.

Wang, Y., Zhang, Q., Wang, Y., Yang, J., and Lin, Z. (2022). Chaos is a ladder: A new theoretical understanding of contrastive learning via augmentation overlap. In *ICLR*. OpenReview.net. Cited on pages 3, 8, and 24.

Wen, Z. and Li, Y. (2021). Toward understanding the feature learning process of self-supervised contrastive learning. In *ICML*. PMLR. Cited on pages 3 and 16.

Wen, Z. and Li, Y. (2022). The mechanism of prediction head in non-contrastive self-supervised learning. In *NeurIPS*. Cited on page 30.

Wright, S. (1921). Correlation and causation. *Journal of Agricultural Research*, (7). Cited on page 5.

Wu, M., Zhuang, C., Mosse, M., Yamins, D., and Goodman, N. (2020). On Mutual Information in Contrastive Learning for Visual Representations. *arXiv preprint*. Cited on pages 3, 4, 30, and 31.

Xiao, T., Wang, X., Efros, A. A., and Darrell, T. (2021). What should not be contrastive in contrastive learning. In *ICLR*. OpenReview.net. Cited on pages 3, 4, 23, 29, and 30.

Yuksekgonul, M., Bianchi, F., Kalluri, P., Jurafsky, D., and Zou, J. (2023a). When and why vision-language models behave like bags-of-words, and what to do about it? In *The Eleventh International Conference on Learning Representations*. Cited on page 2.

Yuksekgonul, M., Bianchi, F., Kalluri, P., Jurafsky, D., and Zou, J. (2023b). When and why vision-language models behave like bags-of-words, and what to do about it? In *International Conference on Learning Representations*. Cited on page 4.

Zbontar, J., Jing, L., Misra, I., LeCun, Y., and Deny, S. (2021). Barlow Twins: Self-Supervised Learning via Redundancy Reduction. In Meila, M. and Zhang, T., editors, *ICML*. PMLR. Cited on page 1.

Zhai, R., Liu, B., Risteski, A., Kolter, Z., and Ravikumar, P. (2023). Understanding Augmentation-based Self-Supervised Representation Learning via RKHS Approximation. *arXiv preprint*. Cited on pages 3 and 16.

Zhang, J. and Ma, K. (2022). Rethinking the augmentation module in contrastive learning: Learning hierarchical augmentation invariance with expanded views. In *CVPR*. Cited on pages 3, 4, 23, and 30.

Zhang, Q., Wang, Y., and Wang, Y. (2023). Identifiable contrastive learning with automatic feature importance discovery. In *NeurIPS*. Cited on pages 9 and 27.

Zheng, H., Chen, X., Yao, J., Yang, H., Li, C., Zhang, Y., Zhang, H., Tsang, I., Zhou, J., and Zhou, M. (2021). Contrastive Attraction and Contrastive Repulsion for Representation Learning. *arXiv preprint*. Cited on page 4.

Zimmermann, R. S., Sharma, Y., Schneider, S., Bethge, M., and Brendel, W. (2021). Contrastive Learning Inverts the Data Generating Process. In *ICML*. PMLR. Cited on pages 1, 2, 3, 4, 5, 8, 16, 27, and 29.

P3: InfoNCE: Identifying the Gap between Theory and Practice.

InfoNCE: Identifying the Gap Between Theory and Practice

Ziyin, L., Lubana, E. S., Ueda, M., and Tanaka, H. (2022). Loss Landscape of Self-Supervised Learning. In *NeurIPS 2022 Workshop: Self-Supervised Learning - Theory and Practice*. Cited on page 16.

P3: InfoNCE: Identifying the Gap between Theory and Practice.

Rusak*, Reizinger*, Juhos*, Bringmann, Zimmermann†, Brendel†

Checklist

i) For all models and algorithms presented, check if you include:
  i) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes] Our theoretical assumptions are summarized in Assum. 1, the experimental settings in § 4.
  ii) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes] We provide a detailed analysis of the effect of batch size and computational requirements in Appxs. D and E.
  iii) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes] We include anonymized code in the supplement. In addition, we described our software stack in Appx. F.

ii) For any theoretical claim, check if you include:
  i) Statements of the full set of assumptions of all theoretical results. [Yes] Confer Assum. 1 and Appx. C.
  ii) Complete proofs of all theoretical results. [Yes] Confer Appx. C
  iii) Clear explanations of any assumptions. [Yes] Confer § 3.

iii) For all figures and tables that present empirical results, check if you include:
  i) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes] We describe our experimental details in § 4 and the software stack in Appx. F.
  ii) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes] We describe our experimental details in § 4
  iii) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes] Confer § 4
  iv) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes] We detail compute requirements in Appx. E.

iv) If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
  i) Citations of the creator If your work uses existing assets. [Yes] Confer § 4 and Appx. F.
  ii) The license information of the assets, if applicable. [Not Applicabble]
  iii) New assets either in the supplemental material or as a URL, if applicable. [No] We will release the code upon acceptance.
  iv) Information about consent from data providers/curators. [Not Applicable]
  v) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

v) If you used crowdsourcing or conducted research with human subjects, check if you include:
  i) The full text of instructions given to participants and screenshots. [Not Applicable]
  ii) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
  iii) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

A Partitioning the Latent Space

Zimmermann et al. (2021) assumes that the latent variables are interchangeable for both their marginals and conditionals are the same. This gives rise to specific symmetries in the latent space $\mathcal{Z}$, which might become restrictive in real-world scenarios. von Kügelgen et al. (2021) introduces a two-fold partitioning of $\mathcal{Z}$, where the marginal is assumed to be the same for all components z_i of the latent vector $\boldsymbol{z}$. Namely, they assume that some latent factors (termed content variables $\boldsymbol{z}_c$) have a delta distribution as conditional, i.e., those variables do not change across the positive pairs. This model has been deemed useful in understanding why dimensionality collapse happens. However, such an assumption might be too simplistic in specific scenarios (at least for some of the latents), which depends on the augmentation strategy Dubois et al. (2022); Ziyin et al. (2022); Tian (2023); Chen et al. (2021); Tamkin and He (2022); Wen and Li (2021); Zhai et al. (2023). Thus, we can think of Jing et al. (2022) as a further generalization, though the authors do not make the connection explicit. They consider augmentations, but their reasoning concerns the observation space $\mathcal{X}$. They conclude—similar to Dubois et al. (2022); Tamkin and He (2022)—that the stronger the augmentation, the more prevalent the collapse. Both to generalize and to formalize these insights, we introduce the notion of partitioning of $\mathcal{Z}$:

Definition A.1 (Partitions of $\mathcal{Z}$)**.** Given a latent space $\mathcal{Z}$ and a latent vector $\boldsymbol{z} \in \mathcal{Z}$, we say that subsets of latent components $z_i^k, \dots, z_{i+n-1}^k$ belong to partition $\mathcal{Z}^k$ of dimension n, when in the ground truth DGP their conditional variances $\sigma^2_{z_i|\boldsymbol{z}}$ are the same, i.e.:

$$\forall j \neq l \in \{i, \dots, i+n-1\} : \sigma^2_{z_j^+|\boldsymbol{z}} = \sigma^2_{z_l^+|\boldsymbol{z}}. \tag{4}$$

When all conditional variances are equal, then we get back the setting of Zimmermann et al. (2021); when the number of partitions is two, with one of the conditional variances being zero, then we are in the regime of von Kügelgen et al. (2021). We formalize this scenario as follows:

Definition A.2 (Content–style partitioning of $\mathcal{Z}$)**.** Following von Kügelgen et al. (2021), the latent space $\mathcal{Z}$ is said to be partitioned into content ($\boldsymbol{z}_c$, invariant) and style ($\boldsymbol{z}_s$, changing) subspaces if and only if

i) $\mathcal{Z} = \mathcal{Z}_c \times \mathcal{Z}_s : \mathcal{Z}_c \cap \mathcal{Z}_s = \emptyset; \quad \dim \mathcal{Z}_c + \dim \mathcal{Z}_s = \dim \mathcal{Z}$;
ii) the conditional distribution of $\boldsymbol{z}_c$ latents is a δ-distribution (von Kügelgen et al., 2021, Assum. 3.1);
iii) the conditional distribution of $\boldsymbol{z}_s$ latents is non-degenerate (von Kügelgen et al., 2021, Assum. 3.2);

yielding the following factorization of the latent conditional $p\left(\boldsymbol{z}^+|\boldsymbol{z}\right) = \delta\left(\boldsymbol{z}^{+,c} - \boldsymbol{z}_c\right) p(\boldsymbol{z}^{+,s}|\boldsymbol{z}_s)$.

That is, our DGP does not cover the exact setting of von Kügelgen et al. (2021), as we cannot have zero variance. If having zero variance (or equivalently, infinite concentration in $\boldsymbol{\Lambda}$), then we might say that our results incorporate the content-style setting.

B Contrastive Learning: Bayes Optimum

For the sake of completeness, this section presents contrastive learning in a more general form, alongside a theorem regarding its unconditional (Bayes-) optimum. This theorem is referenced several times across the identifiability proofs in Appx. C. Although the content of this section is considered to be known by the research community, we still decided to include it to make the presentation of the theory more self-contained. Appx. C presents the novelties introduced by this paper.

Remark B.1. In the following, we will refer to vectors as $\boldsymbol{y}$ to provide a unified treatment both for $\boldsymbol{y} = \boldsymbol{z}$ and $\boldsymbol{y} = \boldsymbol{x}$.

Definition B.1 (General Contrastive Learning (CL) Problem)**.** Let $\mathcal{Y}$ be a data domain. Let $\boldsymbol{Y} := (\boldsymbol{y}, \boldsymbol{y}^+, \boldsymbol{y}_1^-, \dots, \boldsymbol{y}_M^-) \sim p_{\boldsymbol{Y}}$ be a random vector with all elements coming from $\mathcal{Y}$. We refer to $\boldsymbol{y}$ as the *anchor point*, to $\boldsymbol{y}^+$ as its *positive pair* and to $\boldsymbol{y}_i^-$ as one of its *negative pairs*. Let us assume that

i) the pairs $\boldsymbol{y}^+, \boldsymbol{y}_1^-, \dots, \boldsymbol{y}_M^-$ are conditionally completely independent when conditioned on $\boldsymbol{y}$ and
ii) the conditional distribution of negative pairs w.r.t. the anchor coincide, i.e., for any i, j we have $p_{\boldsymbol{y}_i^-|\boldsymbol{y}} = p_{\boldsymbol{y}_j^-|\boldsymbol{y}}$.

Let $\mathcal{U}$ be a class of bivariate functions $u : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$, hereinafter called *similarity functions* and let us optimize the following objective amongst all functions $u \in \mathcal{U}$:

$$\min_{u \in \mathcal{U}} \quad \mathcal{L}_g(u) := \mathbb{E}\left[-\ln \frac{e^{u(\boldsymbol{y}, \boldsymbol{y}^+)}}{e^{u(\boldsymbol{y}, \boldsymbol{y}^+)} + \sum_{j=1}^M e^{u(\boldsymbol{y}, \boldsymbol{y}_j^-)}} \right]. \tag{5}$$

The tuple $(\mathcal{Y}, p_{\boldsymbol{Y}}, \mathcal{U}, \mathcal{L}_g)$ is called a *general contrastive learning (CL) problem.*

Remark B.2. Denoting $p_{\text{pos}} := p_{\boldsymbol{y}^+|\boldsymbol{y}}$ and $p_{\text{neg}} := p_{\boldsymbol{y}_j^-|\boldsymbol{y}}$, the latter being independent of the exact choice of j per assumption *ii*), the joint data distribution is the following:

$$p_{\boldsymbol{Y}}(\boldsymbol{y}, \boldsymbol{y}^+, \boldsymbol{y}_1^-, \dots, \boldsymbol{y}_M^-) = p_{\boldsymbol{y}}(\boldsymbol{y})\, p_{\text{pos}}(\boldsymbol{y}^+|\boldsymbol{y}) \prod_{j=1}^{M} p_{\text{neg}}(\boldsymbol{y}_j^-|\boldsymbol{y}). \tag{6}$$

Theorem B.1 (Bayes-Optima of CL)**.** *Let $(\mathcal{Y}, p_{\boldsymbol{Y}}, \mathcal{U}, \mathcal{L}_g)$ be a general CL problem (Defn. B.1) with $\boldsymbol{Y} := (\boldsymbol{y}, \boldsymbol{y}^+, \boldsymbol{y}_1^-, \dots, \boldsymbol{y}_M^-) \sim p_{\boldsymbol{Y}}$ a* continuous random vector*, such that:*

i) $\boldsymbol{y}$ is supported on $\mathcal{Y}$,

ii) for any $\boldsymbol{y} \in \mathcal{Y}$, both conditional distributions $p_{\text{neg}}(\cdot|\boldsymbol{y}), p_{\text{pos}}(\cdot|\boldsymbol{y}) := p_{\boldsymbol{y}^+|\boldsymbol{y}}(\cdot|\boldsymbol{y})$ are supported on $\mathcal{Y}$.

Then an arbitrary measurable function $u : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ is a global minimum of $\mathcal{L}_g$ (amongst all measurable u's) if and only if there exists a measurable function $c : \mathcal{Y} \to \mathbb{R}$ such that the following holds almost everywhere (w.r.t. any continuous measure of $\mathcal{Y}$):

$$u(\boldsymbol{y}, \tilde{\boldsymbol{y}}) = c(\boldsymbol{y}) + \ln \frac{p_{\text{pos}}(\tilde{\boldsymbol{y}}|\boldsymbol{y})}{p_{\text{neg}}(\tilde{\boldsymbol{y}}|\boldsymbol{y})}. \tag{7}$$

This statement has already been proven by Matthes et al. (2023), however we provide an alternative argument. The key ideas have also emerged in Sohn (2016); Gutmann and Hyvärinen (2010); Ma and Collins (2018).

To prove the theorem, we rewrite the optimization objective as the classification objective of discriminating the positive pair from all the negative pairs, given an anchor point. More specifically, all the pairs $\boldsymbol{y}^+, \boldsymbol{y}_1^-, \dots, \boldsymbol{y}_M^-$ are randomly shuffled, and, then, the task is to estimate the index K of the positive pair from the new sequence, $\tilde{\boldsymbol{y}}_0, \dots, \tilde{\boldsymbol{y}}_M$, given the unchanged anchor $\boldsymbol{y}$. The optimization objective is a categorical cross-entropy (CE) with a predictor of the form (being the output of a softmax function, the predictor's output is normalized to a finite probability distribution):

$$\left[\frac{e^{u(\boldsymbol{y}, \tilde{\boldsymbol{y}}_i)}}{\sum_j e^{u(\boldsymbol{y}, \tilde{\boldsymbol{y}}_j)}} \,\middle|\, i \in \{0, 1, \dots, M\} \right]. \tag{8}$$

We formalize the instance discrimination task as follows:

Definition B.2 (Instance Discrimination Data)**.** The *instance discrimination data* $(\boldsymbol{y}, \tilde{\boldsymbol{y}}_0, \dots, \tilde{\boldsymbol{y}}_M; K)$ corresponding to a general CL problem $(\mathcal{Y}, p_{\boldsymbol{Y}}, \mathcal{U}, \mathcal{L}_g)$ is defined by the following process:

Step 1) $\boldsymbol{y} \sim p_{\boldsymbol{y}}$

Step 2) $K \sim Uni(\{0, 1, \dots, M\})$

Step 3) $\tilde{\boldsymbol{y}}_K \sim p_{\text{pos}}(\cdot|\boldsymbol{y})$

Step 4) $\forall j \in \{0, 1, \dots, K-1, K+1, \dots, M\} : \tilde{\boldsymbol{y}}_j \sim p_{\text{neg}}(\cdot|\boldsymbol{y})$.

Remark B.3. The joint data distribution of the instance discrimination data is:

$$p(\boldsymbol{y}, \tilde{\boldsymbol{y}}_0, \dots, \tilde{\boldsymbol{y}}_M, K = i) = p_{\boldsymbol{y}}(\boldsymbol{y})\, \mathbb{P}(K = i)\, p_{\text{pos}}(\tilde{\boldsymbol{y}}_i|\boldsymbol{y}) \prod_{0 \le j \le M,\, j \ne i} p_{\text{neg}}(\tilde{\boldsymbol{y}}_j|\boldsymbol{y}). \tag{9}$$

Definition B.3 (Instance Discrimination Problem)**.** Let the instance discrimination data $(\boldsymbol{y}, \tilde{\boldsymbol{y}}_0, \dots, \tilde{\boldsymbol{y}}_M; K)$ correspond to a general CL problem $(\mathcal{Y}, p_{\boldsymbol{Y}}, \mathcal{U}, \mathcal{L}_g)$ (Defn. B.2) that also satisfies the assumptions of Thm. B.1. We define the *instance discrimination problem* as:

$$\min_{u \in \mathcal{U}} \quad \mathcal{L}^{id}(u) := \mathbb{E}\left[-\sum_{i=0}^{M} \delta_{K=i} \ln \frac{e^{u(\boldsymbol{y}, \tilde{\boldsymbol{y}}_i)}}{\sum_{j=0}^{M} e^{u(\boldsymbol{y}, \tilde{\boldsymbol{y}}_j)}} \right], \tag{10}$$

where δ_E is the indicator variable of event E.

Remark B.4. When conditioned on $K = k$, the data distribution Eq. (9) can be slightly simplified to:

$$p(\boldsymbol{y}, \tilde{\boldsymbol{y}}_0, \dots, \tilde{\boldsymbol{y}}_M | K = k) = p_{\boldsymbol{y}}(\boldsymbol{y})\, p_{\text{pos}}(\tilde{\boldsymbol{y}}_k|\boldsymbol{y}) \prod_{0 \le j \le M,\, j \ne k} p_{\text{neg}}(\tilde{\boldsymbol{y}}_j|\boldsymbol{y}). \tag{11}$$

Comparing Eq. (11) to Eq. (6) in Rem. B.2, this distribution acts as if $\tilde{\boldsymbol{y}}_k$ (or $\tilde{\boldsymbol{y}}_{j \ne k}$) was a positive pair (or negative pairs) in a contrastive learning problem w.r.t. anchor $\boldsymbol{y}$.

Lemma B.1. *Under the assumptions of Defn. B.3 we have $\mathcal{L}_g(u) = \mathcal{L}^{id}(u)$, for any $u \in \mathcal{U}$.*

Proof. First, using the law of total expectation $\mathbb{E}[A] = \mathbb{E}[\mathbb{E}[A|B]]$, we expand $\mathcal{L}^{id}$ w.r.t. K:

$$\mathcal{L}^{id}(u) = \mathbb{E}_K\left[\mathbb{E}\Big[-\sum_{i=0}^{M}\delta_{K=i}\ln\frac{e^{u(\boldsymbol{y},\tilde{\boldsymbol{y}}_i)}}{\sum_{j=0}^{M}e^{u(\boldsymbol{y},\tilde{\boldsymbol{y}}_j)}}\Big|K\Big]\right] \tag{12}$$

$$= \frac{1}{M+1}\sum_{k=0}^{M}\mathbb{E}\Big[-\sum_{i=0}^{M}\delta_{K=i}\ln\frac{e^{u(\boldsymbol{y},\tilde{\boldsymbol{y}}_i)}}{\sum_{j=0}^{M}e^{u(\boldsymbol{y},\tilde{\boldsymbol{y}}_j)}}\Big|K=k\Big]. \tag{13}$$

However, for the individual summands:

$$\mathbb{E}\Big[-\sum_{i=0}^{M}\delta_{K=i}\ln\frac{e^{u(\boldsymbol{y},\tilde{\boldsymbol{y}}_i)}}{\sum_{j=0}^{M}e^{u(\boldsymbol{y},\tilde{\boldsymbol{y}}_j)}}\Big|K=k\Big] = \mathbb{E}\Big[-\ln\frac{e^{u(\boldsymbol{y},\tilde{\boldsymbol{y}}_k)}}{\sum_{j=0}^{M}e^{u(\boldsymbol{y},\tilde{\boldsymbol{y}}_j)}}\Big|K=k\Big] \tag{14}$$

Considering Rem. B.4, the last term equals the contrastive objective $\mathcal{L}_g(u)$.
Therefore, substituting into Eq. (12) we get

$$\mathcal{L}^{id}(u) = \frac{1}{M+1}\sum_{k=0}^{M}\mathcal{L}_g(u) = \mathcal{L}_g(u). \tag{15}$$

□

After having rewritten the general contrastive objective, we are ready to prove Thm. B.1.

Proof of Thm. B.1. By taking the corresponding instance discrimination problem $\mathcal{L}^{id}$, it remains to compute the optima of $\mathcal{L}^{id}$. For this, we again expand $\mathcal{L}^{id}$ with the law of total expectation, but now w.r.t. $(\boldsymbol{y}, \tilde{\boldsymbol{y}}_0, \dots, \tilde{\boldsymbol{y}}_M)$:

$$\mathcal{L}^{id}(u) = \mathbb{E}\left[\mathbb{E}\Big[-\sum_{i=0}^{M}\delta_{K=i}\ln\frac{e^{u(\boldsymbol{y},\tilde{\boldsymbol{y}}_i)}}{\sum_{j=0}^{M}e^{u(\boldsymbol{y},\tilde{\boldsymbol{y}}_j)}}\Big|\boldsymbol{y},\tilde{\boldsymbol{y}}_0,\dots,\tilde{\boldsymbol{y}}_M\Big]\right] \tag{16}$$

$$= \mathbb{E}\Big[-\sum_{i=0}^{M}\mathbb{E}\Big[\delta_{K=i}\Big|\boldsymbol{y},\tilde{\boldsymbol{y}}_0,\dots,\tilde{\boldsymbol{y}}_M\Big]\ln\frac{e^{u(\boldsymbol{y},\tilde{\boldsymbol{y}}_i)}}{\sum_{j=0}^{M}e^{u(\boldsymbol{y},\tilde{\boldsymbol{y}}_j)}}\Big] \tag{17}$$

$$= \mathbb{E}\Big[-\sum_{i=0}^{M}\mathbb{P}\Big(K=i\Big|\boldsymbol{y},\tilde{\boldsymbol{y}}_0,\dots,\tilde{\boldsymbol{y}}_M\Big)\ln\frac{e^{u(\boldsymbol{y},\tilde{\boldsymbol{y}}_i)}}{\sum_{j=0}^{M}e^{u(\boldsymbol{y},\tilde{\boldsymbol{y}}_j)}}\Big], \tag{18}$$

where we used the properties that $\mathbb{E}\big[A\,f(B)\big|B\big] = f(B)\,\mathbb{E}\big[A\big|B\big]$ and that for any event E, $\mathbb{E}[\delta_E] = \mathbb{P}(E)$.
Let us denote $\tilde{\boldsymbol{Y}} = (\boldsymbol{y}, \tilde{\boldsymbol{y}}_0, \dots, \tilde{\boldsymbol{y}}_M)$ and $\eta_i\big(\tilde{\boldsymbol{Y}}\big) \coloneqq \mathbb{P}\big(K=i\big|\boldsymbol{y},\tilde{\boldsymbol{y}}_0,\dots,\tilde{\boldsymbol{y}}_M\big)$, the quantity $\boldsymbol{\eta}\big(\tilde{\boldsymbol{Y}}\big) = (\eta_0,\dots,\eta_M)$ is the ground-truth conditional distribution of index K. Denoting $F_i\big(\tilde{\boldsymbol{Y}}\big) = e^{u(\boldsymbol{y},\tilde{\boldsymbol{y}}_i)}/(\sum_{j=0}^{M}e^{u(\boldsymbol{y},\tilde{\boldsymbol{y}}_j)})$, the quantity $\boldsymbol{F}\big(\tilde{\boldsymbol{Y}}\big) = (F_0,\dots,F_M)$ is the inferred conditional distribution of index K. Then,

$$\mathcal{L}^{id}(u) = \mathbb{E}_{\tilde{\boldsymbol{Y}}}\Big[-\sum_{i=0}^{M}\eta_i\big(\tilde{\boldsymbol{Y}}\big)\ln F_i\big(\tilde{\boldsymbol{Y}}\big)\Big] \tag{19}$$

$$= \mathbb{E}_{\tilde{\boldsymbol{Y}}}\Big[\sum_{i=0}^{M}\eta_i\big(\tilde{\boldsymbol{Y}}\big)\ln\frac{\eta_i\big(\tilde{\boldsymbol{Y}}\big)}{F_i\big(\tilde{\boldsymbol{Y}}\big)}\Big] + \mathbb{E}_{\tilde{\boldsymbol{Y}}}\Big[-\sum_{i=0}^{M}\eta_i\big(\tilde{\boldsymbol{Y}}\big)\ln\eta_i\big(\tilde{\boldsymbol{Y}}\big)\Big] \tag{20}$$

$$= \mathbb{E}_{\tilde{\boldsymbol{Y}}}\Big[\,KL\Big(\boldsymbol{\eta}\big(\tilde{\boldsymbol{Y}}\big)\Big\|\boldsymbol{F}\big(\tilde{\boldsymbol{Y}}\big)\Big)\,\Big] + \mathbb{E}_{\tilde{\boldsymbol{Y}}}\Big[\,H\Big(\boldsymbol{\eta}\big(\tilde{\boldsymbol{Y}}\big)\Big)\,\Big], \tag{21}$$

where KL and H denote the Kullback-Leibler Divergence (KL) and entropy of the finite distributions over indices $\{0,1,\dots,M\}$, provided as arguments. Due to the non-negativity of the KL, we can lower bound $\mathcal{L}_g(u)$:

$$\mathcal{L}_g(u) = \mathcal{L}^{id}(u) \geq \mathbb{E}_{\tilde{\boldsymbol{Y}}}\Big[\,H\Big(\boldsymbol{\eta}\big(\tilde{\boldsymbol{Y}}\big)\Big)\,\Big], \tag{22}$$

which is independent of the choice of the function u. Equality is attained if and only if the KL vanishes, which occurs if and only if the true and inferred conditional distribution of index K coincide almost everywhere w.r.t. $\tilde{\boldsymbol{Y}} \sim p(\boldsymbol{y}, \tilde{\boldsymbol{y}}_0, \dots, \tilde{\boldsymbol{y}}_M)$. The statement is in terms of the indices, since the instance discrimination task is about classifying which index belongs to the positive pair. Hence, for any i and almost everywhere w.r.t. $\tilde{\boldsymbol{Y}}$:

$$\frac{e^{u(\boldsymbol{y},\tilde{\boldsymbol{y}}_i)}}{\sum_{j=0}^{M} e^{u(\boldsymbol{y},\tilde{\boldsymbol{y}}_j)}} = F_i(\tilde{\boldsymbol{Y}}) = \eta_i(\tilde{\boldsymbol{Y}}) = \mathbb{P}\big(K = i \big| \boldsymbol{y}, \tilde{\boldsymbol{y}}_0, \dots, \tilde{\boldsymbol{y}}_M\big). \tag{23}$$

To rewrite the last expression, we use assumptions *i)* and *ii)* and the uniformity of K (step *Step 2*)). Thus, the data distribution Eq. (9) of the instance discrimination problem can be rewritten as

$$p(\boldsymbol{y}, K = i, \tilde{\boldsymbol{y}}_0, \dots, \tilde{\boldsymbol{y}}_M) = \frac{1}{M+1}\, p_{\boldsymbol{y}}(\boldsymbol{y})\, \frac{p_{\text{pos}}(\tilde{\boldsymbol{y}}_i|\boldsymbol{y})}{p_{\text{neg}}(\tilde{\boldsymbol{y}}_i|\boldsymbol{y})} \prod_{j=0}^{M} p_{\text{neg}}(\tilde{\boldsymbol{y}}_j|\boldsymbol{y}). \tag{24}$$

With this in mind, Eq. (23) becomes

$$\frac{e^{u(\boldsymbol{y},\tilde{\boldsymbol{y}}_i)}}{\sum_{j=0}^{M} e^{u(\boldsymbol{y},\tilde{\boldsymbol{y}}_j)}} = \mathbb{P}\big(K = i \big| \boldsymbol{y}, \tilde{\boldsymbol{y}}_0, \dots, \tilde{\boldsymbol{y}}_M\big) = \frac{p(\boldsymbol{y}, K = i, \tilde{\boldsymbol{y}}_0, \dots, \tilde{\boldsymbol{y}}_M)}{\sum_{j=0}^{M} p(\boldsymbol{y}, K = j, \tilde{\boldsymbol{y}}_0, \dots, \tilde{\boldsymbol{y}}_M)} = \frac{\frac{p_{\text{pos}}(\tilde{\boldsymbol{y}}_i|\boldsymbol{y})}{p_{\text{neg}}(\tilde{\boldsymbol{y}}_i|\boldsymbol{y})}}{\sum_{j=0}^{M} \frac{p_{\text{pos}}(\tilde{\boldsymbol{y}}_j|\boldsymbol{y})}{p_{\text{neg}}(\tilde{\boldsymbol{y}}_j|\boldsymbol{y})}}. \tag{25}$$

From this it can be seen that there exists a measurable, positive function $\tilde{c} : \mathcal{Y} \to \mathbb{R}$ such that

$$e^{u(\boldsymbol{y},\tilde{\boldsymbol{y}})} = \tilde{c}(\boldsymbol{y}) \cdot \frac{p_{\text{pos}}(\tilde{\boldsymbol{y}}|\boldsymbol{y})}{p_{\text{neg}}(\tilde{\boldsymbol{y}}|\boldsymbol{y})} \quad \text{holds almost everywhere (w.r.t. any continuous measure of } \mathcal{Y}). \tag{26}$$

By taking the logarithm we get the desired result. □

C Identifiability Theory

C.1 Identifiability of Anisotropic CL

Definition C.1 (Anisotropic data generating process, ADGP)**.** Let $\mathcal{Z} := \mathbb{S}^{d-1}$ be the $(d-1)$-dimensional unit hypersphere and let $\boldsymbol{g} : \mathcal{Z} \to \mathcal{X} \subseteq \mathbb{R}^D$ be an invertible and continuous function. Let $\boldsymbol{\Lambda} \in \mathbb{R}^{d \times d}$ be a positive definite diagonal matrix.

The following process is called an *anisotropic data generating process (ADGP)*:

Step 1° $\boldsymbol{z} \sim Uni(\mathcal{Z})$
Step 2° $\boldsymbol{z}^+ \sim p_{\boldsymbol{z}^+|\boldsymbol{z}}(\cdot|\boldsymbol{z})$, where $p_{\boldsymbol{z}^+|\boldsymbol{z}}(\boldsymbol{z}^+|\boldsymbol{z}) \propto e^{-\|\boldsymbol{z}^+ - \boldsymbol{z}\|^2_{\boldsymbol{\Lambda}}}$
Step 3° for all $j \in \{1, \dots, M\}, \boldsymbol{z}_j^- \sim Uni(\mathcal{Z})$
Step 4° $\boldsymbol{x}, \boldsymbol{x}^+, \{\boldsymbol{x}_j^-\} \xleftarrow{\boldsymbol{g}} \boldsymbol{z}, \boldsymbol{z}^+, \{\boldsymbol{z}_j^-\}$

The pair $(\boldsymbol{g}, \boldsymbol{\Lambda})$ is called the *characterization* of the process. Sets $\mathcal{Z}, \mathcal{X}$ are called the *latent* and *observation spaces*, respectively.

Remark C.1. The random vector $(\boldsymbol{x}, \boldsymbol{x}^+, \{\boldsymbol{x}_j^-\})$ meets the assumptions of Defn. B.1, so a contrastive learning problem can be well-defined.

Definition C.2 (Anistropic InfoNCE loss, AnInfoNCE)**.** Let an ADGP be characterized by $(\boldsymbol{g}, \boldsymbol{\Lambda})$ and let $(\boldsymbol{x}, \boldsymbol{x}^+, \{\boldsymbol{x}_j^-\})$ be the observed data (all coming from observation space $\mathcal{X} \subseteq \mathbb{R}^D$). Let us optimize the following objective amongst all positive definite (PD) diagonal matrices $\hat{\boldsymbol{\Lambda}}$ and all continuous encoders $\boldsymbol{f} : \mathbb{R}^D \to \mathcal{Z}$:

$$\min_{\substack{\hat{\boldsymbol{\Lambda}} \text{ PD diag} \\ \boldsymbol{f} \text{ continuous}}} \mathcal{L}_{\text{AINCE}}(\boldsymbol{f}, \hat{\boldsymbol{\Lambda}}) := \mathbb{E}\left[-\ln \frac{e^{-\|\boldsymbol{f}(\boldsymbol{x}^+) - \boldsymbol{f}(\boldsymbol{x})\|^2_{\hat{\boldsymbol{\Lambda}}}}}{e^{-\|\boldsymbol{f}(\boldsymbol{x}^+) - \boldsymbol{f}(\boldsymbol{x})\|^2_{\hat{\boldsymbol{\Lambda}}}} + \sum_{j=1}^{M} e^{-\|\boldsymbol{f}(\boldsymbol{x}_j^-) - \boldsymbol{f}(\boldsymbol{x})\|^2_{\hat{\boldsymbol{\Lambda}}}}} \right]. \tag{27}$$

We call the objective $\mathcal{L}_{\text{AINCE}}$ the *Anistropic InfoNCE (AnInfoNCE) loss.*

Remark C.2. The tuple $(\mathcal{X}, p(\boldsymbol{x}, \boldsymbol{x}^+, \{\boldsymbol{x}_j^-\}), \mathcal{U}, \mathcal{L}_{\text{AINCE}})$ is a special case of the general contrastive learning problem (Defn. B.1), where $\mathcal{U} = \{u \,|\, u(\boldsymbol{x}, \boldsymbol{x}^+) = -\|\boldsymbol{f}(\boldsymbol{x}^+) - \boldsymbol{f}(\boldsymbol{x})\|^2_{\hat{\boldsymbol{\Lambda}}} \text{ for any pair } (\boldsymbol{f}, \hat{\boldsymbol{\Lambda}})\}$ is the hypothesis class of similarity functions.

Theorem 3.1b (Identifiability of Anisotropic CL)**.** *Let an ADGP (Defn. C.1) be characterized by* $(\boldsymbol{g}, \boldsymbol{\Lambda})$ *with latent and observed data being* $(\boldsymbol{z}, \boldsymbol{z}^+, \{\boldsymbol{z}_j^-\})$ *and* $(\boldsymbol{x}, \boldsymbol{x}^+, \{\boldsymbol{x}_j^-\})$ *. Let* $\mathcal{L}_{\text{AINCE}}$ *denote the AnInfoNCE loss (Defn. C.2).*

If a pair $(\boldsymbol{f}, \hat{\boldsymbol{\Lambda}})$ *(globally) minimizes the* $\mathcal{L}_{\text{AINCE}}$ *loss, then:*

i) $\hat{\boldsymbol{\Lambda}}$ *is equal to* $\boldsymbol{\Lambda}$ *up to a permutation of elements and*

ii) $\boldsymbol{f} \circ \boldsymbol{g}$ *is a block-orthogonal transformation, where each block acts on latents of equal weight* $\boldsymbol{\Lambda}_{ii}$*.*

In other words, latent $\boldsymbol{z}$ *is identified up to a block-orthogonal transformation.*

Proof of Thm. 3.1b. Our proof consists of two steps:

Step 1: Based on Appx. B, optimums of AnInfoNCE connect the density ratio to the similarity function u.
Step 2: Then, we solve the resulting functional equation.

Step 1: Connecting the density ratio to the similarity function. First, we are going rewrite the $\mathcal{L}_{\text{AINCE}}$ loss function into a form that favors the analysis of $\boldsymbol{f} \circ \boldsymbol{g}$. Plugging in $\boldsymbol{x} = \boldsymbol{g}(\boldsymbol{z})$ (*Step 4°* in Defn. C.1) into the $\mathcal{L}_{\text{AINCE}}$-loss:

$$\mathcal{L}_{\text{AINCE}}(\boldsymbol{f}, \hat{\boldsymbol{\Lambda}}) = \mathbb{E}\left[- \ln \frac{e^{-\|\boldsymbol{f}\circ\boldsymbol{g}(\boldsymbol{z}^+) - \boldsymbol{f}\circ\boldsymbol{g}(\boldsymbol{z})\|^2_{\hat{\boldsymbol{\Lambda}}}}}{e^{-\|\boldsymbol{f}\circ\boldsymbol{g}(\boldsymbol{z}^+) - \boldsymbol{f}\circ\boldsymbol{g}(\boldsymbol{z})\|^2_{\hat{\boldsymbol{\Lambda}}}} + \sum_{j=1}^M e^{-\|\boldsymbol{f}\circ\boldsymbol{g}(\boldsymbol{z}_j^-) - \boldsymbol{f}\circ\boldsymbol{g}(\boldsymbol{z})\|^2_{\hat{\boldsymbol{\Lambda}}}}} \right] =: \tilde{\mathcal{L}}(\boldsymbol{f} \circ \boldsymbol{g}, \hat{\boldsymbol{\Lambda}}), \tag{28}$$

with the optimization still over $\boldsymbol{f}$ (and $\hat{\boldsymbol{\Lambda}}$). However, as the generator $\boldsymbol{g}$ is continuously invertible on the *compact* set $\mathcal{Z}$, its inverse $\boldsymbol{g}^{-1}$ is automatically continuous as well. Therefore, any continuous function $\boldsymbol{h} : \mathcal{Z} \to \mathcal{Z}$ can take the role of $\boldsymbol{f} \circ \boldsymbol{g}$, by substituting $\boldsymbol{f} = \boldsymbol{h} \circ \boldsymbol{g}^{-1}$ continuous. Hence, minimizing $\mathcal{L}_{\text{AINCE}}(\boldsymbol{f}, \hat{\boldsymbol{\Lambda}})$ for $\boldsymbol{f}$ continuous is equivalent to minimizing the new, latent space loss $\tilde{\mathcal{L}}(\boldsymbol{h}, \hat{\boldsymbol{\Lambda}})$ for $\boldsymbol{h}$ continuous:

$$\min_{\substack{\hat{\boldsymbol{\Lambda}} \text{ PD diag} \\ \boldsymbol{f} \text{ cont}}} \mathcal{L}_{\text{AINCE}}(\boldsymbol{f}, \hat{\boldsymbol{\Lambda}}) = \min_{\substack{\hat{\boldsymbol{\Lambda}} \text{ PD diag} \\ \boldsymbol{h} \text{ cont}}} \tilde{\mathcal{L}}(\boldsymbol{h}, \hat{\boldsymbol{\Lambda}}). \tag{29}$$

The new hypothesis class is $\tilde{\mathcal{U}} = \{u \,|\, u(\boldsymbol{z}, \boldsymbol{z}^+) = -\|\boldsymbol{h}(\boldsymbol{z}^+) - \boldsymbol{h}(\boldsymbol{z})\|^2_{\hat{\boldsymbol{\Lambda}}} \text{ for any pair } (\boldsymbol{h}, \hat{\boldsymbol{\Lambda}})\}$.

We claim that $\tilde{\mathcal{L}}$ can attain its Bayes- or unconditional optimum (Thm. B.1), i.e., as if there were no constraints on the similarity function u. According to Thm. B.1, an arbitrary similarity function u minimizes $\tilde{\mathcal{L}}$ amongst all possible measurable u's (attaining the Bayes-optimum) if and only if for a suitable function c:

$$u(\boldsymbol{z}, \tilde{\boldsymbol{z}}) = c(\boldsymbol{z}) + \ln \frac{p_{\boldsymbol{z}^+|\boldsymbol{z}}(\tilde{\boldsymbol{z}}|\boldsymbol{z})}{p_{\boldsymbol{z}^-|\boldsymbol{z}}(\tilde{\boldsymbol{z}}|\boldsymbol{z})} \quad \text{holds almost everywhere on } \mathcal{Z}. \tag{30}$$

In our special case, u is restricted to take the form $u(\boldsymbol{z}, \tilde{\boldsymbol{z}}) = -\|\boldsymbol{h}(\tilde{\boldsymbol{z}}) - \boldsymbol{h}(\boldsymbol{z})\|^2_{\hat{\boldsymbol{\Lambda}}}$. Besides that, $\boldsymbol{z}_j^- \sim Uni(\mathcal{Z})$ with a constant density function and $p_{\boldsymbol{z}^-|\boldsymbol{z}} = p_{\boldsymbol{z}^-} \equiv \text{const}$. We also plug in the anisotropic positive conditional distribution $p_{\boldsymbol{z}^+|\boldsymbol{z}}(\tilde{\boldsymbol{z}}|\boldsymbol{z}) \propto e^{-\|\tilde{\boldsymbol{z}} - \boldsymbol{z}\|^2_{\boldsymbol{\Lambda}}}$. After merging every constant additive term (including normalization constants) into function c, we get the functional equation with unknowns $\boldsymbol{h}, \hat{\boldsymbol{\Lambda}}, c$:

$$\|\boldsymbol{h}(\tilde{\boldsymbol{z}}) - \boldsymbol{h}(\boldsymbol{z})\|^2_{\hat{\boldsymbol{\Lambda}}} = c(\boldsymbol{z}) + \|\tilde{\boldsymbol{z}} - \boldsymbol{z}\|^2_{\boldsymbol{\Lambda}} \quad \text{holds almost everywhere on } \mathcal{Z}. \tag{31}$$

Observe that the equation is trivially satisfied for $\boldsymbol{h} = id_{\mathcal{Z}}, \hat{\boldsymbol{\Lambda}} = \boldsymbol{\Lambda}, c \equiv 0$. This special case corresponds to having $\boldsymbol{f} = \boldsymbol{g}^{-1}$, i.e., we inverted the exact generator and also reconstructed $\boldsymbol{\Lambda}$. Consequently, our loss function attains its Bayes-minimum even in our restricted hypothesis class. Now leveraging again Thm. B.1 in the other direction, the above shows us that all minimizers are, in fact, solutions of Eq. (31).

Let's take an arbitrary solution $\boldsymbol{h}, \hat{\boldsymbol{\Lambda}}, c$. Noticeably, the single-variable, measurable function c is equal (almost everywhere) to a two-variable continuous function. This is only possible if the latter function, depending on $(\boldsymbol{z}, \tilde{\boldsymbol{z}})$ is a single-variable, continuous function of $\boldsymbol{z}$. In this case, swapping c to this exact function will also reveal a good solution and, therefore, it is only required to solve Eq. (31) for c continuous and on the full space $\mathcal{Z}$. By taking the special case of $\tilde{\boldsymbol{z}} \leftarrow \boldsymbol{z}$, we see that $c(\boldsymbol{z}) = 0$ follows.

Step 2: Solving the functional equation. What is left to solve is the following functional equation in terms of the variables $\boldsymbol{h}, \hat{\boldsymbol{\Lambda}}$:

$$\|\boldsymbol{h}(\tilde{\boldsymbol{z}}) - \boldsymbol{h}(\boldsymbol{z})\|^2_{\hat{\boldsymbol{\Lambda}}} = \|\tilde{\boldsymbol{z}} - \boldsymbol{z}\|^2_{\boldsymbol{\Lambda}} \quad \text{holds for any } \tilde{\boldsymbol{z}}, \boldsymbol{z} \text{ on } \mathcal{Z}. \tag{32}$$

For this, let $\boldsymbol{z}_0 \in \mathcal{Z}$ be fixed and let us define the following functions:

$$\boldsymbol{a}(\boldsymbol{z}) = \hat{\boldsymbol{\Lambda}}^{1/2}\big(\boldsymbol{h}(\boldsymbol{z}) - \boldsymbol{h}(\boldsymbol{z}_0)\big) \quad \text{and} \quad \boldsymbol{b}(\boldsymbol{z}) = \boldsymbol{\Lambda}^{1/2}(\boldsymbol{z} - \boldsymbol{z}_0). \tag{33}$$

In this case, the following properties are easily verified:

$$\|\boldsymbol{a}(\tilde{\boldsymbol{z}}) - \boldsymbol{a}(\boldsymbol{z})\|^2 = \|\boldsymbol{b}(\tilde{\boldsymbol{z}}) - \boldsymbol{b}(\boldsymbol{z})\|^2, \tag{34}$$

$$\boldsymbol{a}(\boldsymbol{z}_0) = \boldsymbol{b}(\boldsymbol{z}_0) = 0. \tag{35}$$

By substituting $\tilde{\boldsymbol{z}} \leftarrow \boldsymbol{z}_0$ into Eq. (34) and using Eq. (35), we get that

$$\|\boldsymbol{a}(\boldsymbol{z})\|^2 = \|\boldsymbol{b}(\boldsymbol{z})\|^2 \text{ holds for any } \boldsymbol{z} \in \mathcal{Z}. \tag{36}$$

After expanding Eq. (34) and using Eq. (36) twice, we get that, similarly to the distance and norm, the scalar product is equivalently transformed by $\boldsymbol{a}$ and $\boldsymbol{b}$:

$$\langle \boldsymbol{a}(\tilde{\boldsymbol{z}}), \boldsymbol{a}(\boldsymbol{z}) \rangle = \langle \boldsymbol{b}(\tilde{\boldsymbol{z}}), \boldsymbol{b}(\boldsymbol{z}) \rangle \text{ for any } \tilde{\boldsymbol{z}}, \boldsymbol{z} \in \mathcal{Z}. \tag{37}$$

From this, it can be proven that there exists an orthogonal linear transformation $\boldsymbol{O}$ of $\mathbb{R}^d$ such that:

$$\boldsymbol{a}(\boldsymbol{z}) = \boldsymbol{O}\boldsymbol{b}(\boldsymbol{z}) \text{ holds for any } \boldsymbol{z} \in \mathcal{Z}. \tag{38}$$

To see this, we first prove that the following mapping is well-defined on the linear span of $\boldsymbol{b}(\mathcal{Z})$:

$$\boldsymbol{O} : \sum_{i=1}^{n} \alpha_i \boldsymbol{b}(\boldsymbol{z}_i) \mapsto \sum_{i=1}^{n} \alpha_i \boldsymbol{a}(\boldsymbol{z}_i) \tag{39}$$

Assume there exist two equal expansions (with potentially zero coefficients):

$$\sum_{i=1}^{n} \alpha_i \boldsymbol{b}(\boldsymbol{z}_i) = \sum_{i=1}^{n} \beta_i \boldsymbol{b}(\boldsymbol{z}_i). \tag{40}$$

In this case the difference and the distance square vanish:

$$\left\| \sum_{i=1}^{n} (\alpha_i - \beta_i) \boldsymbol{b}(\boldsymbol{z}_i) \right\|^2 = 0. \tag{41}$$

However, due to the equivalent transformation of the scalar product (Eq. (37)), after the expansion of Eq. (41) we can essentially change every occurrence of $\boldsymbol{b}$ to $\boldsymbol{a}$ and receive that:

$$\left\| \sum_{i=1}^{n} (\alpha_i - \beta_i) \boldsymbol{a}(\boldsymbol{z}_i) \right\|^2 = 0, \tag{42}$$

which may hold if and only if

$$\sum_{i=1}^{n} \alpha_i \boldsymbol{a}(\boldsymbol{z}_i) = \sum_{i=1}^{n} \beta_i \boldsymbol{a}(\boldsymbol{z}_i), \tag{43}$$

proving that $\boldsymbol{O}$ in Eq. (39) is well-defined on the linear span of $\boldsymbol{b}(\mathcal{Z})$.

Secondly, the facts that $\boldsymbol{O}$ is linear and that $\boldsymbol{a}(\boldsymbol{z}) = \boldsymbol{O}\boldsymbol{b}(\boldsymbol{z})$ hold trivially. We also see that the image of $\boldsymbol{b}$, i.e. $\boldsymbol{b}(\mathcal{Z}) = \boldsymbol{b}(\mathbb{S}^{d-1})$ is full dimensional and hence $\boldsymbol{O}$ is defined on the entire $\mathbb{R}^d$. The orthogonality is a direct consequence of the scalar product preservation property (Eq. (37)) and the definition of $\boldsymbol{O}$ (Eq. (39)).

Consequently, we have proven that there exists an orthogonal linear transformation $\boldsymbol{O}$ of $\mathbb{R}^d$ such that:

$$\hat{\boldsymbol{\Lambda}}^{1/2}(\boldsymbol{h}(\boldsymbol{z}) - \boldsymbol{h}(\boldsymbol{z}_0)) = \boldsymbol{O}\boldsymbol{\Lambda}^{1/2}(\boldsymbol{z} - \boldsymbol{z}_0) \text{ holds for any } \boldsymbol{z} \in \mathcal{Z}. \tag{44}$$

As $\hat{\boldsymbol{\Lambda}}$ is positive definite and invertible, it follows (after merging constant terms containing $\boldsymbol{z}_0$), that:

$$\boldsymbol{h}(\boldsymbol{z}) = \hat{\boldsymbol{\Lambda}}^{-1/2}\boldsymbol{O}\boldsymbol{\Lambda}^{1/2}\boldsymbol{z} + \boldsymbol{c} \text{ holds for any } \boldsymbol{z} \in \mathcal{Z}, \text{ for some constant } c. \tag{45}$$

Let us denote $\boldsymbol{H} = \hat{\boldsymbol{\Lambda}}^{-1/2}\boldsymbol{O}\boldsymbol{\Lambda}^{1/2}$. To complete the proof, we have to show that $\boldsymbol{c} = 0$ and $\boldsymbol{H}$ is orthogonal. For this, we use the fact that $\boldsymbol{h}$ is a transformation of the unit hypersphere $\mathcal{Z} = \mathbb{S}^{d-1}$:

$$\langle \boldsymbol{H}\boldsymbol{z}, \boldsymbol{c}\rangle = \frac{1}{4}\Big(\|\boldsymbol{H}\boldsymbol{z} + \boldsymbol{c}\|^2 - \| -\boldsymbol{H}\boldsymbol{z} + \boldsymbol{c}\|^2\Big) \tag{46}$$

$$= \frac{1}{4}\Big(\|\boldsymbol{h}(\boldsymbol{z})\|^2 - \|\boldsymbol{h}(-\boldsymbol{z})\|^2\Big) = \frac{1}{4}(1-1) = 0, \text{ for any } \boldsymbol{z} \in \mathcal{Z}. \tag{47}$$

As $\boldsymbol{H}$ is invertible and, thus, with a full range of $\mathbb{R}^d$, $\boldsymbol{c} = 0$ is concluded. Therefore, $\boldsymbol{h}$ is linear. As unit vectors are mapped to unit vectors, $\boldsymbol{h}$ is norm-preserving and, thus, $\boldsymbol{h} = \boldsymbol{f} \circ \boldsymbol{g}$ is orthogonal.

Now, rewriting $\boldsymbol{H} = \hat{\boldsymbol{\Lambda}}^{-1/2}\boldsymbol{O}\boldsymbol{\Lambda}^{1/2}$ gives us:

$$\boldsymbol{O}\boldsymbol{\Lambda}^{1/2} = \hat{\boldsymbol{\Lambda}}^{1/2}\boldsymbol{H} = \boldsymbol{H}\big(\boldsymbol{H}^\top\hat{\boldsymbol{\Lambda}}^{1/2}\boldsymbol{H}\big), \tag{48}$$

where the first and last formulas provide polar decompositions of the same operator. As the polar decomposition of invertible operators is unique, we conclude that $\boldsymbol{O} = \boldsymbol{H}$, and

$$\boldsymbol{\Lambda}^{1/2} = \boldsymbol{H}^\top\hat{\boldsymbol{\Lambda}}^{1/2}\boldsymbol{H}. \tag{49}$$

After squaring both sides, we get:

$$\boldsymbol{\Lambda} = \boldsymbol{H}^\top\hat{\boldsymbol{\Lambda}}\boldsymbol{H}. \tag{50}$$

Then the right-hand side of Eq. (50) is the eigendecomposition of $\boldsymbol{\Lambda}$. Due to the uniqueness of eigenvalues, there must exist a permutation $\mathbf{P}_{\boldsymbol{h}}$ such that $\hat{\boldsymbol{\Lambda}} = \mathbf{P}_{\boldsymbol{h}}\boldsymbol{\Lambda}\mathbf{P}_{\boldsymbol{h}}^\top$. Then

$$\boldsymbol{\Lambda} = \boldsymbol{H}^\top\mathbf{P}_{\boldsymbol{h}}\boldsymbol{\Lambda}\mathbf{P}_{\boldsymbol{h}}^\top\boldsymbol{H} \tag{51}$$

$$\Leftrightarrow \mathbf{P}_{\boldsymbol{h}}^\top\boldsymbol{H}\boldsymbol{\Lambda} = \boldsymbol{\Lambda}\mathbf{P}_{\boldsymbol{h}}^\top\boldsymbol{H}. \tag{52}$$

In other words, $\mathbf{P}_{\boldsymbol{h}}^\top\boldsymbol{H}$ and $\boldsymbol{\Lambda}$ commute. Then $\mathbf{P}_{\boldsymbol{h}}^\top\boldsymbol{H}$ must be a diagonal matrix if all values on the diagonal of $\boldsymbol{\Lambda}$ are distinct. Should some values be the same, then the corresponding submatrix of $\mathbf{P}_{\boldsymbol{h}}^\top\boldsymbol{H}$ is orthogonal.

To see this more formally, we define $\mathbf{O} := \mathbf{P}_{\boldsymbol{h}}^\top\boldsymbol{H}$. Then, for all i, j,

$$\sum_k \mathrm{O}_{ik}\lambda_k\delta_{kj} = \sum_k \lambda_i\delta_{ik}\mathrm{O}_{kj}, \tag{53}$$

$$\Leftrightarrow \mathrm{O}_{ij}\lambda_j = \lambda_i\mathrm{O}_{ij}, \tag{54}$$

$$\Leftrightarrow \mathrm{O}_{ij}(\lambda_j - \lambda_i) = 0. \tag{55}$$

Hence, for distinct eigenvalues, all off-diagonal entries of $\mathbf{O}$ are zero. Within the space of equivalent eigenvalues, $\mathbf{O}$ is only constrained by the requirement to be orthogonal.

From $\mathbf{O} = \mathbf{P}_{\boldsymbol{h}}^\top\boldsymbol{H}$ we then know that $\boldsymbol{H}$ is — up to permutation — an orthogonal block-diagonal matrix where the size of each block corresponds to the multiplicity of the values on the diagonal of $\boldsymbol{\Lambda}$.

□

C.2 Including hard negative sampling

Definition C.3 (Anisotropic DGP with hard negatives (HN))**.** Let $\boldsymbol{g} : \mathcal{Z} := \mathbb{S}^{d-1} \to \mathcal{X} \subseteq \mathbb{R}^D$ be an invertible and continuous function. Let $\boldsymbol{\Lambda}^+, \boldsymbol{\Lambda}^- \in \mathbb{R}^{d\times d}$ be positive definite diagonal matrices such that $\boldsymbol{\Lambda}^+_{ii} > \boldsymbol{\Lambda}^-_{ii}$ (or equivalently, $\boldsymbol{\Lambda}^+ - \boldsymbol{\Lambda}^-$ is still PD).

The following process is called an *anisotropic DGP with hard negatives (HN)*, characterized by the triple $(\boldsymbol{g}, \boldsymbol{\Lambda}^+, \boldsymbol{\Lambda}^-)$:

Step 1) $\boldsymbol{z} \sim Uni(\mathcal{Z})$

Step 2) $\boldsymbol{z}^+ \sim p_{\boldsymbol{z}^+|\boldsymbol{z}}(\cdot|\boldsymbol{z})$, where $p_{\boldsymbol{z}^+|\boldsymbol{z}}(\boldsymbol{z}^+|\boldsymbol{z}) \propto e^{-\|\boldsymbol{z}^+ - \boldsymbol{z}\|^2_{\boldsymbol{\Lambda}^+}}$

Step 3) for all $j \in \{1, \ldots, M\}, \boldsymbol{z}_j^- \sim p_{\boldsymbol{z}^-|\boldsymbol{z}}(\cdot|\boldsymbol{z})$, where $p_{\boldsymbol{z}^-|\boldsymbol{z}}(\boldsymbol{z}^-|\boldsymbol{z}) \propto e^{-\|\boldsymbol{z}^- - \boldsymbol{z}\|^2_{\boldsymbol{\Lambda}^-}}$

Step 4) $\boldsymbol{x}, \boldsymbol{x}^+, \{\boldsymbol{x}_j^-\} \xleftarrow{\boldsymbol{g}} \boldsymbol{z}, \boldsymbol{z}^+, \{\boldsymbol{z}_j^-\}$.

Corollary 3.1b (Identifiability of anisotropic DGPs with HN)**.** *Let an ADGP with HN (Defn. C.3) be characterized by* $(\boldsymbol{g}, \boldsymbol{\Lambda}^+, \boldsymbol{\Lambda}^-)$ *with latent and observed data being* $(\boldsymbol{z}, \boldsymbol{z}^+, \{\boldsymbol{z}_j^-\})$ *and* $(\boldsymbol{x}, \boldsymbol{x}^+, \{\boldsymbol{x}_j^-\})$*. Let* $\mathcal{L}_{\text{AINCE}}$ *denote the AnInfoNCE loss (Defn. C.2), evaluated w.r.t.* $(\boldsymbol{x}, \boldsymbol{x}^+, \{\boldsymbol{x}_j^-\})$*.*

If a pair $(\boldsymbol{f}, \hat{\boldsymbol{\Lambda}})$ *(globally) minimizes the* $\mathcal{L}_{\text{AINCE}}$ *loss, then:*

i) $\hat{\boldsymbol{\Lambda}}$ *is equal to* $\boldsymbol{\Lambda}^+ - \boldsymbol{\Lambda}^-$ *up to a permutation of elements and*

ii) $\boldsymbol{f} \circ \boldsymbol{g}$ *is an orthogonal transformation, or latent* $\boldsymbol{z}$ *is identified up to an orthogonal transformation.*

Proof. We are following the steps of the Proof of Thm. 3.1b.

First, we again rewrite the $\mathcal{L}_{\text{AINCE}}$ by plugging in $\boldsymbol{x} = \boldsymbol{g}(\boldsymbol{z})$ and receiving $\mathcal{L}_{\text{AINCE}}(\boldsymbol{f}, \hat{\boldsymbol{\Lambda}}) = \tilde{\mathcal{L}}(\boldsymbol{f} \circ \boldsymbol{g}, \hat{\boldsymbol{\Lambda}})$, with optimization over $\boldsymbol{f}$ (and $\hat{\boldsymbol{\Lambda}}$). As $\boldsymbol{h} = \boldsymbol{f} \circ \boldsymbol{g}$ attains all possible continuous functions, we may optimize $\tilde{\mathcal{L}}(\boldsymbol{h}, \hat{\boldsymbol{\Lambda}})$. The hypothesis class becomes, again, $\mathcal{U} = \{u \,|\, u(\boldsymbol{z}, \boldsymbol{z}^+) = -\|\boldsymbol{h}(\boldsymbol{z}^+) - \boldsymbol{h}(\boldsymbol{z})\|^2_{\hat{\boldsymbol{\Lambda}}} \text{ for any pair } (\boldsymbol{h}, \hat{\boldsymbol{\Lambda}})\}$.

Secondly, according to Thm. B.1, an arbitrary similarity function u minimizes $\tilde{\mathcal{L}}$ amongst all possible measurable u's (attaining the Bayes-optimum) if and only if for a suitable function c:

$$u(\boldsymbol{z}, \tilde{\boldsymbol{z}}) = c(\boldsymbol{z}) + \ln \frac{p_{\boldsymbol{z}^+|\boldsymbol{z}}(\tilde{\boldsymbol{z}}|\boldsymbol{z})}{p_{\boldsymbol{z}^-|\boldsymbol{z}}(\tilde{\boldsymbol{z}}|\boldsymbol{z})} \quad \text{holds almost everywhere on } \mathcal{Z}. \tag{56}$$

We are about to plug in all our constraints. The major difference compared to the Proof of Thm. 3.1b is that besides the positive conditional distribution being $p_{\boldsymbol{z}^+|\boldsymbol{z}}(\tilde{\boldsymbol{z}}|\boldsymbol{z}) \propto e^{-\|\tilde{\boldsymbol{z}}-\boldsymbol{z}\|^2_{\boldsymbol{\Lambda}^+}}$, the negative conditional also changes to $p_{\boldsymbol{z}^-|\boldsymbol{z}}(\tilde{\boldsymbol{z}}|\boldsymbol{z}) \propto e^{-\|\tilde{\boldsymbol{z}}-\boldsymbol{z}\|^2_{\boldsymbol{\Lambda}^-}}$. After plugging in the restricted form of u and merging every constant normalizing term into function c, Eq. (56) becomes:

$$\begin{aligned}
-\|\boldsymbol{h}(\tilde{\boldsymbol{z}}) - \boldsymbol{h}(\boldsymbol{z})\|^2_{\hat{\boldsymbol{\Lambda}}} &= c(\boldsymbol{z}) + \ln \frac{e^{-\|\tilde{\boldsymbol{z}}-\boldsymbol{z}\|^2_{\boldsymbol{\Lambda}^+}}}{e^{-\|\tilde{\boldsymbol{z}}-\boldsymbol{z}\|^2_{\boldsymbol{\Lambda}^-}}} && (57)\\
&= c(\boldsymbol{z}) - \|\tilde{\boldsymbol{z}} - \boldsymbol{z}\|^2_{\boldsymbol{\Lambda}^+} + \|\tilde{\boldsymbol{z}} - \boldsymbol{z}\|^2_{\boldsymbol{\Lambda}^-} && (58)\\
&= c(\boldsymbol{z}) - \langle \tilde{\boldsymbol{z}} - \boldsymbol{z}, \boldsymbol{\Lambda}^+(\tilde{\boldsymbol{z}} - \boldsymbol{z})\rangle + \langle \tilde{\boldsymbol{z}} - \boldsymbol{z}, \boldsymbol{\Lambda}^-(\tilde{\boldsymbol{z}} - \boldsymbol{z})\rangle && (59)\\
&= c(\boldsymbol{z}) - \langle \tilde{\boldsymbol{z}} - \boldsymbol{z}, (\boldsymbol{\Lambda}^+ - \boldsymbol{\Lambda}^-)(\tilde{\boldsymbol{z}} - \boldsymbol{z})\rangle = c(\boldsymbol{z}) - \|\tilde{\boldsymbol{z}} - \boldsymbol{z}\|^2_{\boldsymbol{\Lambda}^+ - \boldsymbol{\Lambda}^-}. && (60)
\end{aligned}$$

Hence, we get the functional equation with unknowns $\boldsymbol{h}, \hat{\boldsymbol{\Lambda}}, c$:

$$\|\boldsymbol{h}(\tilde{\boldsymbol{z}}) - \boldsymbol{h}(\boldsymbol{z})\|^2_{\hat{\boldsymbol{\Lambda}}} = c(\boldsymbol{z}) + \|\tilde{\boldsymbol{z}} - \boldsymbol{z}\|^2_{\boldsymbol{\Lambda}^+ - \boldsymbol{\Lambda}^-} \quad \text{holds almost everywhere on } \mathcal{Z}. \tag{61}$$

We conclude our proof by continuing the Proof of Thm. 3.1b by substituting $\boldsymbol{\Lambda} = \boldsymbol{\Lambda}^+ - \boldsymbol{\Lambda}^-$ PD.

□

Remark C.3 (Subsuming the uniform marginal in the negative conditional with $\boldsymbol{\Lambda}^- = 0$). When using a negative conditional in the form of Eq. (2), $\boldsymbol{\Lambda}^- = 0$ accounts for a uniform marginal.

C.3 Identifiability of a Loss Ensemble

In this setting the observations $\boldsymbol{x}$ come from k different DGPs, with a *shared* generator $\boldsymbol{g}$ and separate concentration parameters $\{\boldsymbol{\Lambda}^i\}_{i=1}^k$. That is, this setting is akin to works investigating identifiability in multiple environments (Hyvärinen and Morioka, 2016; Gresele et al., 2019; Rajendran et al., 2023) and reflects the practice in CL of optimizing multiple loss components with different data augmentations (Eastwood et al., 2023; Xiao et al., 2021; Zhang and Ma, 2022).

Corollary 3.2b (Identifiability of ensemble AnInfoNCE)**.** *Assume k DGPs, each satisfying Assum. 1. Assume that they share* $\boldsymbol{g}$*, but have different concentration parameters* $\{\boldsymbol{\Lambda}^i\}$*, for* $1 \leq i \leq k$*. Assume a shared encoder* $\boldsymbol{f}$ *and k learnable* $\hat{\boldsymbol{\Lambda}}^i$ *parameters. Then, the tuple* $(\boldsymbol{f}, \{\hat{\boldsymbol{\Lambda}}^i\})$ *minimizing the ensemble loss* $\sum_i^k \mathcal{L}_{\text{AINCE}}(\boldsymbol{f}, \hat{\boldsymbol{\Lambda}}^i)$ *identifies the latents up to a block-orthogonal transformation.*

Proof. $\boldsymbol{f} = \boldsymbol{g}^{-1}$ together with $\hat{\boldsymbol{\Lambda}}^i = \boldsymbol{\Lambda}^i$ minimizes the i^{th} loss term. As $\boldsymbol{g}$ is shared, the individual losses can attain their minima simultaneously. Hence, this also holds for any minimum of the ensemble loss. Applying Thm. 3.1 to either of them concludes the proof. □

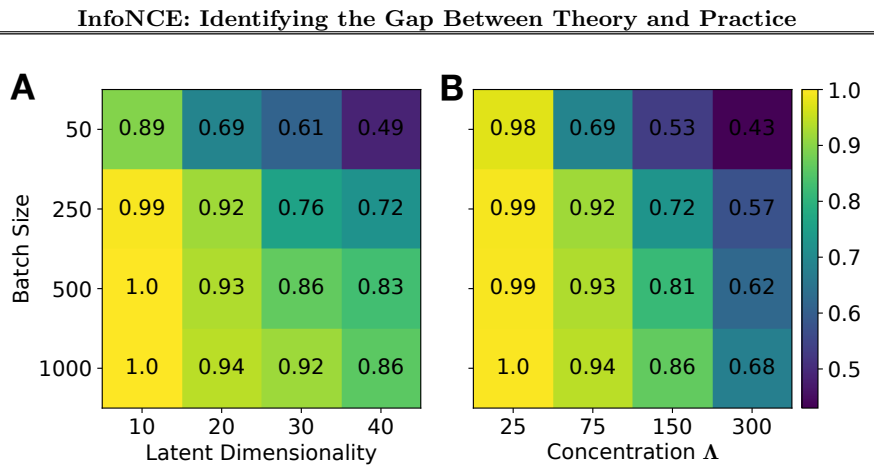


Figure 4: **Latent Dimensionality, Concentration, and Batch Size Influence Identifiability.** Linear identifiability, quantified by the R^2 score between reconstructed and ground-truth latents, degrades with **A:** higher latent dimensionality d; and **B:** larger concentration parameter $\mathbf{\Lambda}$ in the ground-truth positive conditional. Both detrimental effects can be countered by increasing the batch size.

Remark C.4 (Sufficiency condition for identifiability up to permutations for the ensemble AnInfoNCE loss). Ensemble AnInfoNCE is identifiable up to permutations if, under the conditions of Cor. 3.2, there are sufficiently many DGPs such that for all distinct latent factors z_k, z_l, there exists a DGP with concentration parameter $\mathbf{\Lambda}^i$ such that $\mathbf{\Lambda}^i_{kk} \neq \mathbf{\Lambda}^i_{ll}$.

D Additional Analysis: Ablations and Parameter Sensitivity

In Section 5 of the main manuscript, we identified the use of augmentations as the primary issue in observing a trade-off between accuracy and identifiability on real-world data. In this Section, we analyze other reasons why scaling AnInfoNCE to real-world data is challenging. In particular, we observe that the interplay between the batch size, the latent dimensionality and the value of the concentration parameter plays a crucial role. To demonstrate this, we run controlled experiments on synthetic data to investigate the assumptions and properties of our theory. Since our analysis equally applies to the isotropic and anisotropic cases, we use the isotropic one for our experiments for simplicity but consider the general anisotropic case in our additional theoretical results.

D.1 Issues when Scaling up AnInfoNCE

Batch Size Needs to Scale with the Dimensionality. One potential difference between the fully-controlled and the real-world experiments is the latent dimensionality, which will (most likely) be substantially higher for more complex image data. Therefore, we investigate the influence of the latent dimension and batch size on the linear identifiability score (Fig. 4A). We observe a significant degradation of performance with a smaller batch size and higher latent dimensionality, corroborating the theoretical predictions of Wang et al. (2022) and in line with the empirical observation that CL performs better with a higher batch size (Chen et al., 2020a). Even for a modest dimensionality, a rather high batch size is required. Extrapolating our results, feature encoders on the order of what is typically used on CIFAR10 and ImageNet (512 to 2048 dimensions) would require extremely large batch sizes to reach high identifiability scores even if all assumptions match our theory.

Batch Size Needs to Scale with the Concentration $\mathbf{\Lambda}$. Another important difference between our controlled and real-world experiments concerns the choice of concentration parameters and its influence on the batch size for identifiability. More concretely, the more concentrated the conditionals are, and the higher the dimensionality, the less training signal CL has to structure the representation (Wang et al., 2022). This was denoted by Wang et al. (2022) as the role of *augmentation overlap*. Our anisotropic positive conditional distribution could help explain the role of augmentation overlap from an identifiability perspective via the concentration parameter $\mathbf{\Lambda}$, whose effect is illustrated in Fig. 5. Larger $\mathbf{\Lambda}$ means a more concentrated conditional and requires larger batch sizes for sufficient augmentation overlap; otherwise, the R^2 score will degrade (see Fig. 4B for $d = 20$). In real-world experiments, the positive pairs distribution is determined by augmentations. Likely, the shape or the class of an

Rusak*, Reizinger*, Juhos*, Bringmann, Zimmermann†, Brendel†

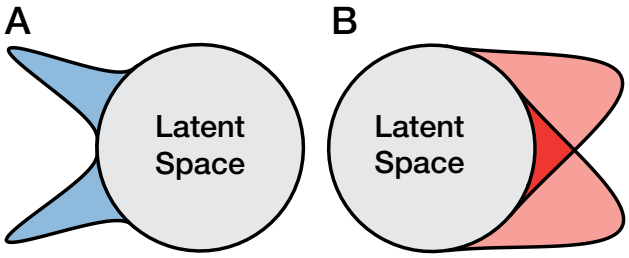


Figure 5: **Concentration of Positive Pairs Influences Augmentation Overlap**. We figuratively visualize conditional distributions with large (**A**) and small (**B**) concentration parameters $\mathbf{\Lambda}$. For large concentration values, samples from the conditional distribution of two anchor points do not overlap, signifying missing augmentation overlap. This is not the case for small $\mathbf{\Lambda}$ values.

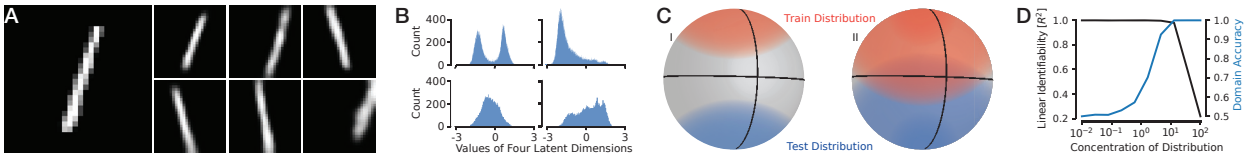


Figure 6: **Data Augmentation Can Violate Our Data Distribution Assumptions.** We train a VAE on augmented MNIST images. **A.** We show an MNIST sample and six augmented versions generated by the VAE. **B.** We project augmented versions of an MNIST sample into the VAE's latent space and visualize the distribution of four dimensions. This demonstrates a violated conditional assumption of a normalized Gaussian. **C.** We visualize violations of the uniformity assumption for the marginal distribution during training and testing. **D.** We observe degrading identifiability scores when the train and test domains can be distinguished. Since an almost perfect domain classifier can be trained on real-world datasets, we conclude that the uniformity assumption is likely violated in practice.

object are minimally affected by the augmentations, which corresponds to those latent dimensions having a large concentration parameter, thereby again increasing the necessary batch size to reach high identifiability scores.

Augmentations can Violate our Conditional Assumption. Next, we consider the role of different positive pair distributions. In § 3, we assume a normalized Gaussian distribution over a hypersphere. However, it is unclear whether practical augmentation strategies follow such a conditional. We test our assumption's validity by training a Variational Autoencoder (VAE) on augmented images from MNIST with only crops and horizontal flips as augmentations since MNIST images are grayscale. Training an encoder on MNIST using these augmentations leads to a KNN accuracy of 96.7%, validating our augmentation choice. We visualize an unaugmented image and VAE reconstructions of its augmented versions in Fig. 6A. To show the conditional distribution, we project augmented images onto the learned VAE latent space and demonstrate in Fig. 6B that the conditional distribution does not follow a normalized Gaussian distribution for most dimensions and can even be bimodal; we show more examples in Fig. 7. This holds even though the latent space of the VAE across all augmented data samples is Gaussian by construction. Our theory does not cover this case and so this mismatch presents a fruitful direction for future research on closing the gap between theory and practice.

Augmentations can Violate our Uniformity Assumption. Our theory assumes the same uniform marginal $p(\boldsymbol{z})$ for both training and testing. However, this assumption's validity is unclear in practice: During training, even anchor points are produced by data augmentation, creating a potential mismatch to the test domain which lacks these augmentations. We test the influence of violations of this assumption by interpolating the marginal between a uniform and a vMF distribution with varying concentration. We center the vMF at the north (south) pole during training (testing) (Fig. 6C). We train a binary classifier to distinguish samples from the training and test distributions. For different concentrations, we compute the domain classification accuracy and the identifiability score (Fig. 6D), showing a stark anticorrelation. Our domain classification results with regular SimCLR augmentations (MNIST: 99.1%, CIFAR10: 99.2%, and ImageNet 98.7%) demonstrate that binary

classifiers can almost perfectly distinguish augmented and non-augmented samples on real-world data. Connecting this to observed anticorrelation with identifiability, we conclude that the mismatch between distributions hurts the quality of representations.

The Conditional's Effect on Learning Dynamics. Finally, the generative process of real-world image data is most likely substantially richer compared to the previously studied synthetic DGPs, resulting in a generally harder learning problem requiring longer training time. In particular, we hypothesize that the conditional distribution's concentration partly influences the learning speed; this aligns with previous observations of flat loss landscapes for Noise Contrastive Estimation (NCE) (Liu et al., 2022). Our anisotropic conditional of positive pairs provides a possible explanation: We hypothesize that similar to Saxe et al. (2014), the factors controlling the variances (in our case, the concentration $\mathbf{\Lambda}$) affect the loss landscape, making some latents *slow* to learn. Thus, it is conceivable that more style-like (ground-truth) latents are learned slower (thus, harder) compared to more content-like ones (see, e.g., in Fig. 2B).

We demonstrate the influence of the concentration parameter on the learning dynamics as follows. We define two DGPs using the synthetic data from Section 4.1. We assume a uniform marginal and an isotropic normalized Gaussian conditional with two different $\mathbf{\Lambda}$ parameters: $\mathbf{\Lambda} = 25$ and $\mathbf{\Lambda} = 75$. A higher concentration parameter corresponds to a more narrow conditional distribution and thus, lower variation between the anchor and its conditional sample in the latent space. We analyze how $\mathbf{\Lambda}$ affects learning dynamics (Fig. 8) and observe that learning the latents follows a step-like behavior, corroborating Simon et al. (2023), where latents with larger $\mathbf{\Lambda}$ are learned slower (Fig. 8 A & D). Alternatively, reaching a target R^2 score requires more gradient steps unless the batch size is increased (Fig. 8 B & E). The cosine similarities' distribution for negative and positive pairs clearly shows that large $\mathbf{\Lambda}$ leads to insufficient augmentation overlap: Since the histograms are disjoint for large $\mathbf{\Lambda}$, the anchor's nearest neighbor is always a positive sample (Fig. 8 C & F). This makes distinguishing positive from negative samples trivial and allows for a short-cut solution. However, augmentation overlap cannot fully explain CL in practice: Saunshi et al. (2022) show that practical augmentations often provide no significant overlap; nonetheless, CL still succeeds, potentially due to (architectural) inductive biases.

D.2 Dependence of Identifiability on Batch Size, Training Duration, and Dimensionality

We now further analyze the dependence of the batch size, training duration, and dimensionality of the latent space on the identifiability scores. To this end, we train models with different batch sizes on data with various latent dimensionality and investigate their identifiability scores. We use 3 random seeds per configuration and train all models for 500k iterations using the same hyperparameters as described above. We consider the usual conditional distribution of positive pairs with a concentration value $\mathbf{\Lambda} = 75$.

Fig. 9 clearly shows that for higher dimensional latent spaces, one needs to train (exponentially) longer and use an increased batch size. Note that even for the fairly low dimensional settings investigated here, large batch sizes and long training runs were necessary to achieve high identifiability scores. Comparing this to the dimensionalities usually assumed/used in practice, e.g., 512 dimensions when training on images, we argue that the usual batch sizes and training durations are too small to achieve high identifiability scores. This observation would explain the mixed performance observed earlier when testing our loss on real-world image data despite its identifiability guarantees.

D.3 Hard Negative Mining and Ensemble AnInfoNCE

Hard Negative Mining. In the previous section, we observed that high concentration parameters require higher batch sizes for successful learning. HN mining can help counter highly concentrated conditionals by increasing the sample density close to the anchors. First, we demonstrate in our synthetic setting for $d = 20$ that with HN mining, the optimum for the loss in terms of $\hat{\mathbf{\Lambda}}$ is indeed at the predicted value of $\mathbf{\Lambda} = \mathbf{\Lambda}^+ - \mathbf{\Lambda}^-$ (Fig. 10A). We also finetune an encoder using AnInfoNCE by adding three hard negatives for each sample. We observe better linear identifiability scores compared to continued regular training, supporting our Cor. 3.1, (Fig. 10B).

Training With Ensemble AnInfoNCE. To test the benefit of ensembling AnInfoNCE for training, we define two DGPs with $d = 20$. The DGPs share a uniform marginal distribution and differ in their respective conditional distributions. To model latents that cannot be changed by either DGP, such as object class or shape, we set a high concentration parameter of 400 for three latents shared between both DGPs. For the remaining seventeen latents, we choose two $\mathbf{\Lambda}$ values of 15 and 250, modeling strongly and weakly varying latents and alternate them

between the DGPs:

$$\mathbf{\Lambda}_1 = \text{diag}\,(400, 400, 400, 15, \ldots, 15, 250, \ldots, 250)$$
$$\mathbf{\Lambda}_2 = \text{diag}\,(400, 400, 400, 250, \ldots, 250, 15, \ldots, 15)\,.$$

We sample data with both DGPs and use two losses with two learnable concentration parameters $\hat{\mathbf{\Lambda}}_1, \hat{\mathbf{\Lambda}}_2$ to model both DGPs. We train the encoder on the sum of both losses. We see improved linear identifiability scores when training the encoder on the loss ensemble compared to InfoNCE or AnInfoNCE (see also Fig. 11A):

Loss	InfoNCE	AnInfoNCE	Ens. AnInfoNCE
Lin. ident. [R^2]	0.48	0.93	0.98

Further, we observe a specialization in the learned $\hat{\Lambda}$ values (Fig. 11B+C), where same colors indicate the same $\hat{\Lambda}$ dimension): We observe an anti-correlation between $\hat{\Lambda}_1$ and $\hat{\Lambda}_2$. Optimizing an ensemble loss makes it more likely that for each latent dimension, there exists an augmentation making the dimensions more content-like with a higher $\mathbf{\Lambda}$.

D.4 Comparison to TriCL (Zhang et al., 2023)

We first discuss the conceptual differences, theoretical results and assumptions betwen TriCL and AnInfoNCE. TriCL extends and theoretically analyzes the spectral contrastive loss formulation (their extension to InfoNCE does not have a proof). Our paper extends InfoNCE and provides an identifiability result in that case. Note that the extensions are also different: We weigh the differences of latent coordinates instead of the latents themselves. The TriCL paper provides a result on encoder output equivalence (similar to Roeder et al. (2021), meaning that all encoders optimizing the TriCL loss are linearly related, or, more precisely $f_1(x) = Af_2(x)$. However, the encoder's output is not related to ground-truth latents in any way. Our paper provides a result on the equivalence of the output with the ground-truth latents, meaning that $f \circ g(z) = Oz$. On one hand, our type of result subsumes the TriCL type of result. Moreover, our result is well-suited for the qualitative assessment of inferred latent features, which is not possible with the latter. The theoretical result from TriCL requires that an additional decorrelation regularizer completely vanishes, whereas our method does not have any such term.

We also show an experimental comparison between TriCL and AnInfoNCE below. First, we noticed a small discrepancy between the paper's definition of TriCL and its experimental implementation in how the loss is computed. Based on the used implementation, the scale parameter is not applied to all terms in the sum but only to half of them. We are not sure how how this discrepancy affects the experiments.

We compare the evolution of linear identifiability scores when training with TriCL or AnInfoNCE in our synthetic experiment in Fig. 12. We set the latent dimensionality to 20 and test two cases for the positive conditional distribution. (1) Uniform Λ set to 100 (top row); (2) Dual partitioned Λ, such that half of the dimensions have a low Λ of 15 ("style"-like dimensions) and the other half high Λ of 250 ("content"-like dimensions) (bottom row). In both cases, we observe that AnInfoNCE behaves much more stably compared to TriCL and reaches higher linear identifiability scores, especially when there is an anisotropic conditional.

D.5 Training Details for VAE Training in MNIST

The VAE consists of three fully connected layers with Leaky ReLU activations and is trained on MNIST for 50 epochs. The latent dimensionality d of the VAE is set to 10. We use the trained VAE to generate training images in the following way: We first sample a latent vector z from the marginal distribution and then sample a vector conditioned on z. Since both vectors are normalized to the sphere surface, we multiply them with $\sqrt{d}$ to project them to a Gaussian vector. Finally, we use the VAE decoder to generate images which we train the encoder with the AnInfoNCE loss on.

To provide an intuition for the quality of samples generated by the VAE, we show images generated from random samples from the latent space in Fig. 13 (left). For the highlighted sample in the top-right corner, we also show samples from the conditional distribution for two different Λ values: $\Lambda = 5$ (middle) and $\Lambda = 50$ (right).

D.6 Training details and additional experiments for C-3DIdent

The Causal 3D-Ident dataset (von Kügelgen et al., 2021) is an extension of the 3D-Ident dataset (Zimmermann et al., 2021) to six more classes. The training split has 252000 images and the test split has 25200 images. We use

the original code base from (von Kügelgen et al., 2021) with the original training hyperparameters. The latent dimensionality is ten and the latent space of C-3DIdent is a hypercube. Following von Kügelgen et al. (2021), we use cosine similarity as our similarity measure which restricts the inferred latents to a hypersphere. To enable our encoder learning all hypercube latents, we set the latent dimensionality to eleven. We train all models for 100K gradient steps.

E Compute requirements

Synthetic experiments. For our synthetic experiments, as described in Sections 4.1, 4.2 as well as in Appendix D, we used a single NVIDIA GeForce RTX 2080 Ti with 11GB of RAM for each experiment. We used 8 CPUs and 8 workers per experiment. The convergence time for each experiment varied, e.g. higher batch size or a lower concentration parameter $\hat{\mathbf{\Lambda}}$ generally imply faster convergence: We show the relationship between the batch size and the concentration parameter in Fig. 8. We set the run time for synthetic experiments to ten hours to make sure that all runs converge; the main experiments shown in Figs. 2 and 3 train within one hour to convergence.

Real-world experiments. On CIFAR10, we trained the ResNet18 model on a single NVIDIA A100 GPU with 40GB RAM and 8 CPUs (with 8 workers). The training took about seven hours per run. The linear readout evaluation as well as the binary augmentations readout took about 30 minutes on the same compute setup.

On ImageNet, we trained a ResNet50 on 8 NVIDIA A100 GPUs with 40GB RAM and 64 CPUs. We set the number of workers to 16. The training took about 15 hours on this setup. The linear readout evaluation as well as the binary augmentations readout took about 3 hours using one GPU.

On C-3DIdent, we trained a ResNet18 model on a single NVIDIA A100 GPU with 40GB RAM and 32 CPUs (with 32 workers). The training took about fourteen hours per run.

Compute costs per experiment, main paper. While we used different GPUs between our synthetic and real-world experiments, we will use a metric of generic "GPU-hours" for both cases. As described below, both GPU types require about the same amount of power.

i) Synthetic data, Fig. 2: Experiments showing the influence of linear identifiability scores when varying $\mathbf{\Lambda}$. We show results for 6 values of $\mathbf{\Lambda}$ for InfoNCE and AnInfoNCE. The GPU run time for this experiment is $6 \cdot 2 \cdot 10h = 120h$.

ii) Synthetic data, Fig. 3: Experiments showing how AnInfoNCE behaves in a slightly more difficult experimental setting on MNIST. We show results for 3 values of $\mathbf{\Lambda}$ for InfoNCE and AnInfoNCE in Fig 3 A+B, and for 2 values of $\mathbf{\Lambda}$ when training with AnInfoNCE in Fig 3 C+D. The GPU run time for this experiment is $3 \cdot 2 \cdot 10h + 2 \cdot 10h = 80h$.

iii) Real-world data, Table 1: We show a trade-off between augmentations readout and downstream accuracy on CIFAR10 and ImageNet. We show results for training with InfoNCE and AnInfoNCE both on CIFAR10 and ImageNet. On CIFAR10, we additionally show results for training with AnInfoNCE without ℓ_2 normalization of the output before calculating the loss. We also conduct the linear readout evaluation on both the backbone and the post-projection features. The compute break-down is as follows:
   i) CIFAR10: Training: $3 \times 7h = 21h$, Linear readout: $3 \cdot 6 \cdot 0.5h \cdot 2 = 18h$. Total: $39h$
   ii) ImageNet: Training: $2 \times 15h \cdot 8$ GPUs$= 240h$, Linear readout: $2 \cdot 3h \cdot 6 \cdot 2 = 72h$. Total: $312h$
   The total compute for the real-world experiments is then 351 GPU hours.

iv) MNIST, inset Table in Sec.5; showing the accuracy-identifiability trade-off. The run time for this experiment has been: $2 \cdot 10h$=$20h$.

v) C-3DIdent, Tab. 2; we compare InfoNCE with AnInfoNCE on C-3DIdent and investigate the accuracy and identifiability trade-off. We tested 2 view generation strategies (via data augmentation and via ground-truth sampling) and ran 3 random seeds for two different loss functions. The total run time for this experiment has been: $2 \cdot 3 \cdot 2 \cdot 14h = 168h$.

Compute costs per experiment, Appendix.

i) Synthetic data, Fig. 4. We show ablation experiments for varying the batch size, the latent dimensionality and the concentration. We show results for 32 different parameter combinations. The GPU run time for this experiment is $32 \cdot 10h = 320h$.

ii) Synthetic data, Fig. 8. We analyze the interplay between higher dimensional latent spaces and batch size in

more detail. We ran 3 random seeds in this experiment, analyzed 9 different batch sizes across 5 different latent dimensionalities. The total run time for this experiment is $3 \cdot 5 \cdot 9 \cdot 10h = 1350h$.

iii) Synthetic data, Fig. 9. We utilize the previous runs from computing results for Fig. 4; this Figure does not incur additional compute costs.

iv) Hard-negative mining, Fig. 10. We show that hard-negative mining can improve identifiability. We finetune one model trained with regular InfoNCE on AnInfoNCE with hard negative examples. The run time for this experiment is $3h$.

v) Synthetic data, Fig. 11. We show that ensembling can improve upon InfoNCE and AnInfoNCE. The run time for this experiment is $3 \cdot 10h = 30h$.

vi) Synthetic data, Fig. 12. We show that AnInfoNCE is more stable and reaches higher linear identifiability scores compared to the TriCL loss. The run times for this experiment is $2 \cdot 10h = 20h$

Total compute estimation. Given the previous paragraphs, the training time estimation is 2774 GPU hours. The maximum power consumption for the 2080Ti GPUs is 250W (Techpowerup, 2024) and 300W for the A100 GPUs (FOLDING.LAR.SYSTEMS, 2024). We assume an upper bound on power consumption of 300W per GPU. Then, the energy spent by the GPUs for the experiments shown in this paper is 832.2kWh. We estimate an additional factor of 20 in terms of energy spent during the development stage of this project. Therefore, the total estimated project cost in terms of energy is about 16.6MWh. This corresponds to 3.4t CO_2 emissions (Rensmart, 2024) which is roughly equivalent to six single-person flights from London to New York (Travel Navigator, 2024).

F Software stack

We use different open source software packages for our experiments, most notably Docker (Merkel, 2014), Singularity (Sylabs Inc, 2018), scipy and numpy (Virtanen et al., 2020), PyTorch (Paszke et al., 2017), and torchvision (Marcel and Rodriguez, 2010).

G Additional related works

G.1 An LVM view of SSL

Saunshi et al. (2019) developed generalization bounds for downstream classification in contrastive learning, with one or more negative samples, assuming the presence of latent class variables and (for some results considering intra-class concentration) sub-Gaussian conditionals. Results also show that more negative samples can hurt performance when the negative samples have the same class (c.f. their Sec. 4.2). Zimmermann et al. (2021) connected SimCLR to nonlinear ICA and proves that the SimCLR loss is equivalent to the cross entropy between ground-truth and approximate conditionals. Based on this result, identifiability results are provided for latent spaces structured as hyperspheres or convex bodies - our theory extends the results of Zimmermann et al. (2021). von Kügelgen et al. (2021) introduced the content-style terminology and proved that content variables can be recovered with non-contrastive SSL under certain assumptions. Daunhawer et al. (2023) extended these results to the multi-modal case where some latent variables are modality-specific. Lyu et al. (2021) extended von Kügelgen et al. (2021) to get guarantees on the identifiability of style variables as well, though they require invertible encoders, whereas von Kügelgen et al. (2021) have results for the non-invertible case. However, Lyu et al. (2021) admit different encoders across views. Lachapelle et al. (2022) investigated the role of sparsity for identifiability in the multi-task scenario, i.e., when the representation is used to solve multiple downstream tasks. This setting is similar to that in Dubois et al. (2022), since the authors used downstream tasks that rely on all latent factors, i.e., for good performance all of them need to be identified. Inducing sparsity in the linear classification head yields an optimal and disentangled representation with identifiability guarantees up to permutation and scaling. Cui et al. (2022), building on Zimmermann et al. (2021); von Kügelgen et al. (2021), introduced the AggNCE loss function and proves that with multiple views, the InfoNCE loss is identifiable up to an invertible matrix. Lee et al. (2021b) quantified how specific pretext tasks can guarantee learning a good representation, i.e., one which can solve the downstream task by only using a linear layer on top. The authors relied on approximate conditional independence between the input and the pretext target (conditioned on the downstream task label and the latents). Eastwood et al. (2023) proposed a new SSL objective inspired from invariance-based methods that allow for the disentanglement of style variables alongside content. This paper can also be seen as a more rigorous treatment of Xiao et al. (2021). However, their identifiability results differ from ours as they mostly rely on multiple sets of augmentations, and a different model and loss pair. **?** unified and extended the results from von Kügelgen et al. (2021); Daunhawer et al. (2023) to the multi-view case with partial observability, showing

that any shared information across arbitrary subsets of views and modalities can be learned up to a smooth bijection using contrastive learning.

G.2 The role of data augmentations

The (empirical) study of different augmentation strategies relates to our work as the underlying data generating process (DGP) is affected by them and our work studies SSL from the DGP's perspective. Within this category, we distinguish two subcategories:

G.2.1 Augmentation strategies with a single latent space

Wu et al. (2020) empirically evaluated collision-free augmentations (i.e., when different data points cannot have the same view—e.g., changing brightness is collision-free, but cropping can introduce collisions). They conclude that collision-free augmentations are not useful for the downstream task of classification on ImageNet (c.f. their Figs. 1 and 5) and CIFAR10 (their Fig. 5)—whereas the ones with collision can improve performance. **?** proposed a more flexible class of data augmentations based on Gaussian random fields and showed that it improves performance on ImageNet and iNaturalist, but their method is sensitive to hyperparameters (strong transformations can degrade the image). Bendidi et al. (2023) extensively studied the effect of data augmentations in natural (biological) images w.r.t. downstream (classification) performance and clustering structure. They emphasize that the choice of data augmentation strategies is a form of weak supervision, and its design requires domain expertise to achieve better performance on a given task, since different augmentations can affect the classes differently. Wagner et al. (2022) introduced the GroupAugment augmentation strategy that searches over a very general space of both augmentations and sampling strategies. The authors also empirically study the role of data augmentation hyperparameters, highlighting that it dramatically impacts performance in SSL. Ericsson et al. (2021) studied learning invariances in contrastive learning and their transfer from synthetic to real-world scenarios. The authors consider two augmentation categories: spatial augmentations change the objects' position, whereas appearance mostly affects color and texture. The authors empirically demonstrate that using augmentations from one of the above groups leads to increased invariance w.r.t. the same augmentation types—these only transfer to some extent from synthetic settings to the real world. However, there is a trade-off in all studied models, i.e., there is no reliable means to learn invariances to both augmentation types. Since different downstream tasks are shown to rely on different invariances, the authors advocate for a fusion of multiple representations, i.e., leveraging multiple representations that were trained by using different types of augmentations.

G.2.2 Augmentation strategies with combined latent spaces

Xiao et al. (2021) introduced LooC (Leave-one-out Contrastive Learning) using multiple embeddings, where each embedding is trained with a different (sub)set of augmentations. The authors argued and empirically demonstrated that by inducing different invariances in the different embeddings, the combined latent space outperforms all considered baselines on multiple downstream tasks. This setting corresponds to our ensemble approach, with the only difference that we share the encoder, thus, we only have a single latent space. Zhang and Ma (2022) proposed a similar strategy to LooC with an add-one augmentation strategy (i.e., their augmentation subsets can be ordered, where the first includes two augmentations, the second the first two plus another one, etc.). Furthermore, they calculate the contrastive loss at different hierarchical levels in the siamese encoder—the authors motivate this step by conjecturing that this way invariances are imposed at different levels. Another key difference is that the embeddings of augmentation parameters are also included in the representation before the projection head (i.e., the loss has information about the data augmentation).

G.3 The effect of normalization

Tian (2023, Lem. 2) shed light into why normalized representations are favorable: the author shows that the Frobenius norm of all weight matrices below an ℓ_2- or layer normalization Ba et al. (2016) terms remains constant. The theory holds for nonlinear MLP with reversible activations, including ReLU; and, interestingly, provides a connection of why we could feasibly replace ℓ_2-normalization with layer normalization. Wen and Li (2022) explained dimensionality collapse for non-contrastive SimSiam (Chen and He, 2021) with a two-layer nonlinear model and output normalization, and analyze the role of the projector. Empirically, when the projector is initialized to the identitiy matrix and only the off-diagonal entries are trainable, the authors obtain competitive representations. They also point out that batch normalization (by enforcing a nonzero variance) is less prone to collapse and conjecture that a complete collapse is more probable with ℓ_2-normalization but do not prove the

conjecture. Their results are in unison with ours since both works state that the content variables are learned first.

Ermolov et al. (2021) proposed a non-contrastive SSL loss where an MSE objective is applied to whitened features and show empirical improvements regarding dimensionality collapse; furthermore, their empirical results suggest that using the InfoNCE loss with ℓ_2-normalization and whitened features leads to divergence. Intuitively, there are some similarities between their normalization strategy and AnInfoNCE , though they are distinct as we normalize the latent factors before calculating the weighted MSE, whereas the authors whiten before normalization. Partially, this can explain why we do not have divergence in the case of ℓ_2-normalization and weighting via the diagonal scaling matrix.

Halvagal et al. (2022) proposed an isotropic loss to "flatten out" the eigenvalue spectrum of the inferred latent covariance, which is remarkably similar to our strategy of using different temperature values.

G.4 The role of negatives and hard negative sampling

Saunshi et al. (2019) showed that more negative samples can hurt performance when the negative samples have the same class. Chuang et al. (2020) made the same observation, i.e., that sampling negative examples truly different (downstream) labels improves performance in a synthetic setting where labels are available. They uncover a NPC in InfoNCE and propose to resolve it with debiasing, resulting in DCL that corrects for the sampling of same-label datapoints, even without knowledge of the true labels. They also bound the difference between the SimCLR and the debiased contrastive losses. Robinson et al. (2021b) extended Chuang et al. (2020) and propose a strategy to draw hard negative samples in an unsupervised way—using the data density and positive conditional distribution—for contrastive learning and analyze the resulting loss theoretically. The authors connect hard negative samples to adversarial samples and analyze the loss in the infinite sample limit in (c.f. their Thm. 4). They also bound the misclassification risk in their Thm. 5. Liu and Wang (2023) extended (Chuang et al., 2020) into the Bayesian framework; the authors propose a hard negative sampling strategy with importance sampling.

Several works tackle the problem of designing (hard) negative sampling strategies. Wu et al. (2020) generalized the negative sampling in InfoNCE, their method include more "exotic" negative sampling schemes, e.g., from a "ring" on a sphere. Robinson et al. (2021a) proposed IFM as a solution that disturbs both positive and negative pairs, though in an implicit manner (it can be thought of as implicit hard negative sampling): it makes positive samples less, negative samples more similar (measured by cosine similarity) to the anchor point. This perturbation is done in the latent space (c.f. Fig. 4 of their paper) and improves linear readout accuracy on multiple image data sets (Fig. 6; without the trade-offs induced by changing τ) and also improves performance on medical data (Tab. 2). The authors also demonstrate that SimCLR combined with IFM improves robustness in the adversarial setting (Fig. 8).

Chuang et al. (2022) proposed RINCE, a symmetric drop-in replacement for InfoNCE, with a theoretical connection to robust symmetric losses in noisy binary classification. The loss (approximately) takes the q^{th} exponent of the positive and negative loss terms, implementing a trade-off between exploration and exploitation. In the limit of $q = 0$, it is the same as InfoNCE. Thus, the authors conclude, InfoNCE puts more weight on hard positives, whereas RINCE on easy positives. RINCE can be thought as an implicit hard/easy positive mining scheme with an exponential weighting factor on uniformity.

H Acronyms

CE cross-entropy

AnInfoNCE Anistropic InfoNCE

CL Contrastive Learning

DGP Data Generating Process

HN hard negative

i.i.d. independent and identically distributed

KL Kullback-Leibler Divergence

LooC Leave-one-out Contrastive Learning
LVM Latent Variable Model

MLP Multi-Layer Perceptron

NCE Noise Contrastive Estimation

PD positive definite

SSL Self-Supervised Learning

VAE Variational Autoencoder
vMF von Mises-Fisher

I Nomenclature

R^2 coefficient of determination
$\mathbb{S}$ hypersphere
$\mathcal{L}_{\text{AINCE}}$ AnInfoNCE loss function
$\mathcal{L}_{\text{INCE}}$ InfoNCE loss function
$\boldsymbol{f}$ encoder map $\mathcal{X} \to \mathcal{Z}$
$\boldsymbol{g}$ decoder map $\mathcal{Z} \to \mathcal{X}$
$\boldsymbol{h}$ composition of encoder and decoder, i.e., $\boldsymbol{f} \circ \boldsymbol{g}$
$\boldsymbol{\tau}$ temperature in $\mathcal{L}_{\text{INCE}}$

Algebra
$\hat{\mathbf{\Lambda}}$ inferred diagonal matrix of weighting factors
$\mathbf{O}$ orthogonal matrix
$\mathbf{\Lambda}$ diagonal matrix of weighting factors
$\mathbf{P}$ permutation matrix

Latents
$\boldsymbol{z}_c$ content latent vector
$\boldsymbol{z}_s$ style latent vector
$\boldsymbol{z}$ latent vector
$\hat{\boldsymbol{z}}$ reconstructed latent vector
$\mathcal{Z}_c$ content
$\mathcal{Z}_s$ style
$\mathcal{Z}$ latents
d dimensionality of the latent space $\mathcal{Z}$
z latent single component
$\boldsymbol{z}^+$ positive latent vector

Observations
D dimensionality of the observation space $\mathcal{X}$
M number of negative samples
$\boldsymbol{x}^-$ negative observation vector
$\boldsymbol{x}$ observation vector
$\mathcal{X}$ observation space
$\boldsymbol{x}^+$ positive observation vector

P3: InfoNCE: Identifying the Gap between Theory and Practice.

Rusak*, Reizinger*, Juhos*, Bringmann, Zimmermann†, Brendel†

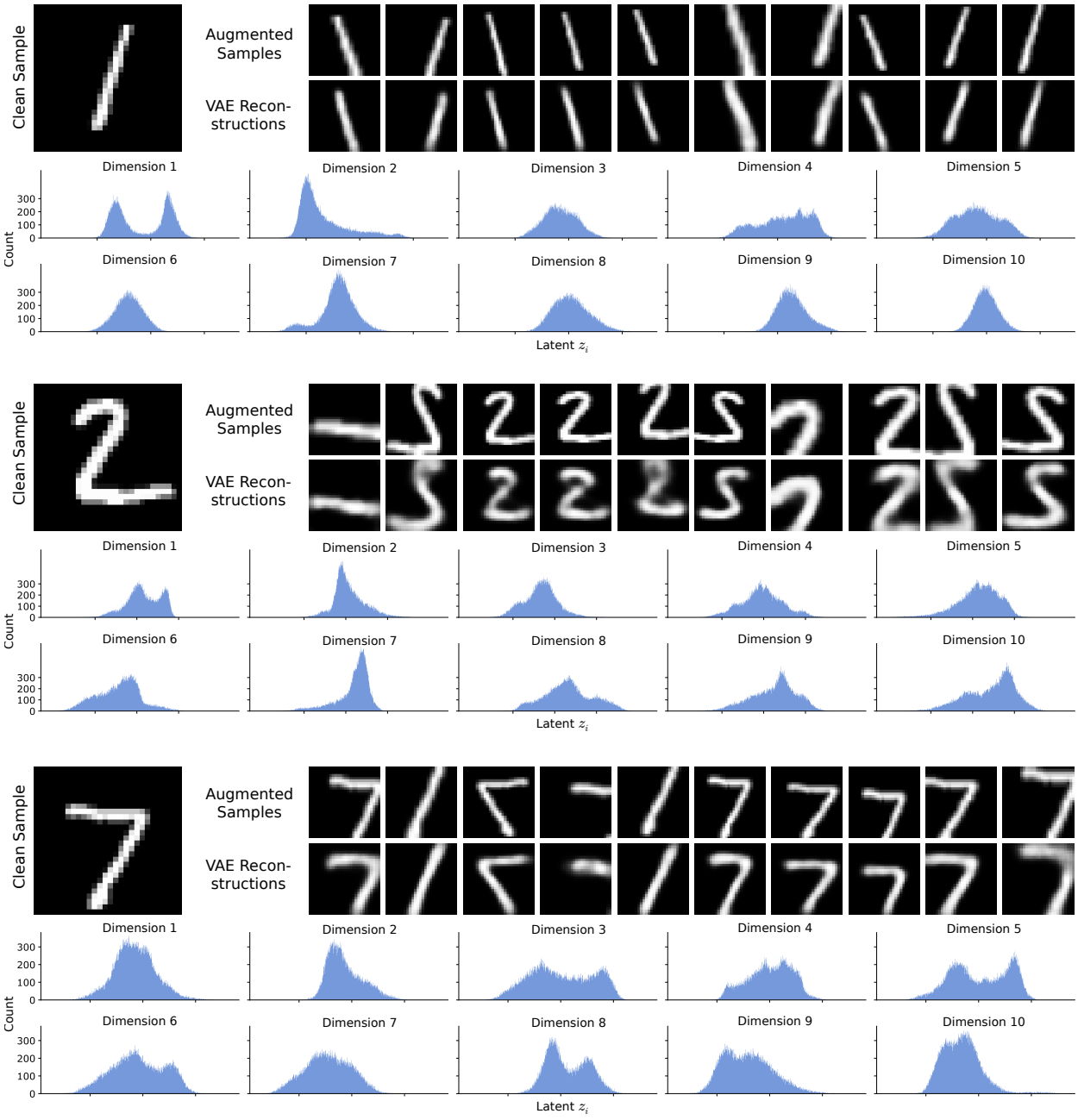


Figure 7: **When Using Augmentations to Generate Views for CL, the Conditional Generally no Longer Follows a Gaussian Distribution.** We show augmented images as well as their reconstructions using a VAE trained on augmented images. We project the augmented images into the latent space of the trained VAE and observe that their distribution does not resemble a Gaussian distribution and can even be bimodal.

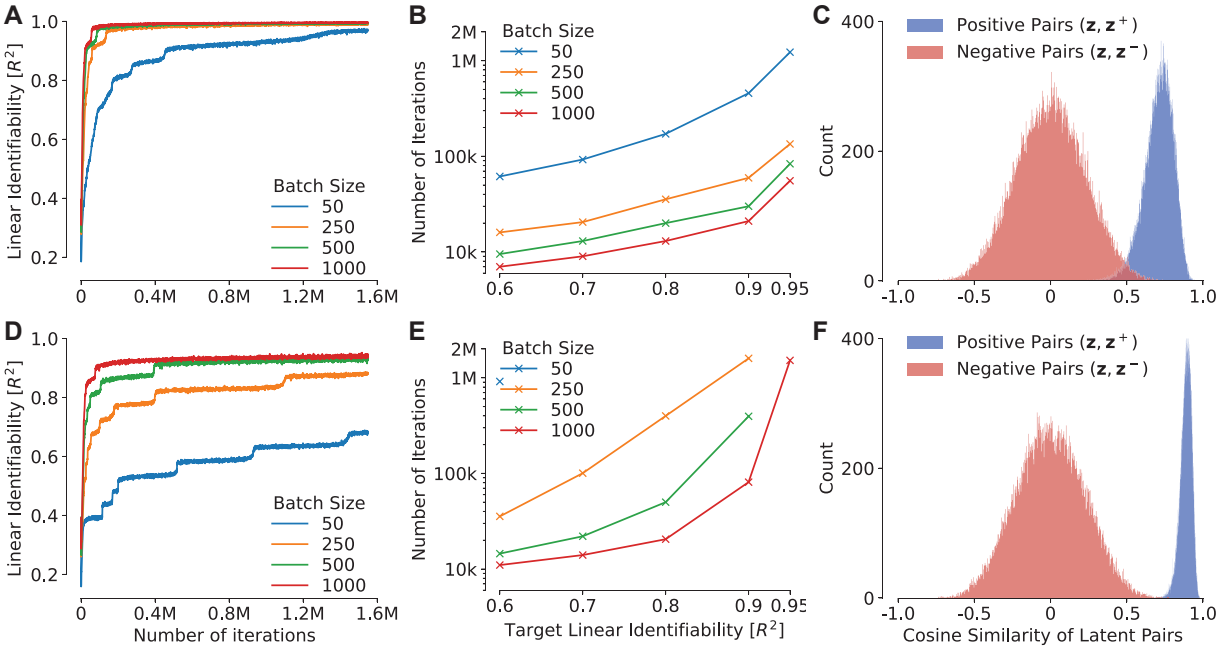


Figure 8: **Analysis of Learning Dynamics for a Small ($\mathbf{\Lambda} = 25$, A-C) and a Large ($\mathbf{\Lambda} = 75$, D-F) Concentration Parameter. A & D**: Linear identifiability measured by the R^2 scores show step-like behavior and generally need more training iterations to reach a certain R^2 score for higher $\mathbf{\Lambda}$. **B & E**: The number of training iterations needed for a certain target R^2 score grows exponentially. **C & F**: We explain degrading R^2 scores for higher concentration parameters $\mathbf{\Lambda}$ with missing overlap in the latent positive and negative distributions.

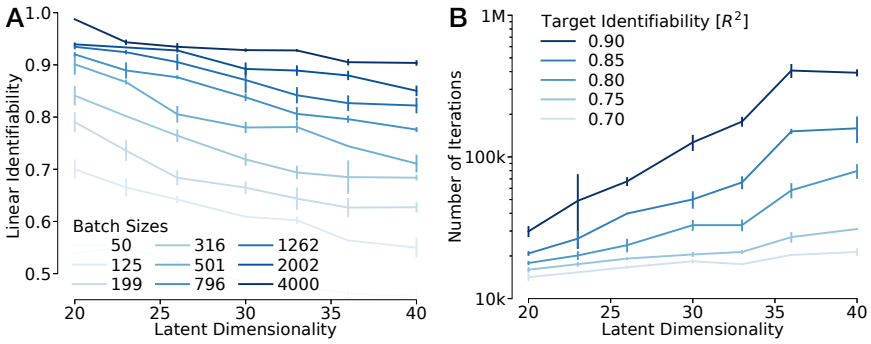


Figure 9: **Higher Dimensional Latent Spaces Require Longer Training with Larger Batch Sizes. A**: The linear identifiability achieved after training for 500k steps decreases with an increasing latent dimensionality but increases with larger batch sizes. **B**: We display the required number of iterations to achieve various target linear identifiability scores (0.70 to 0.90) for a fixed batch size of 4000. Note that the number of iterations required grows exponentially with the latent dimensionality.

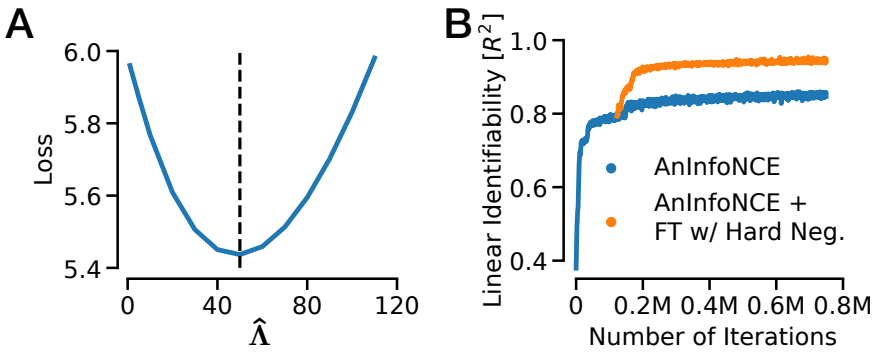


Figure 10: **Hard Negative Mining Improves Identifiability. A:** For $\mathbf{\Lambda}^+ = 150$ and $\mathbf{\Lambda}^- = 100$, we find a loss optimum at $\mathbf{\Lambda} = \mathbf{\Lambda}^+ - \mathbf{\Lambda}^- = 50$ as predicted by Cor. 3.1; **B:** Finetuning (FT) an encoder on a concatenation of hard negative samples and regular negative samples improves the identifiability score.

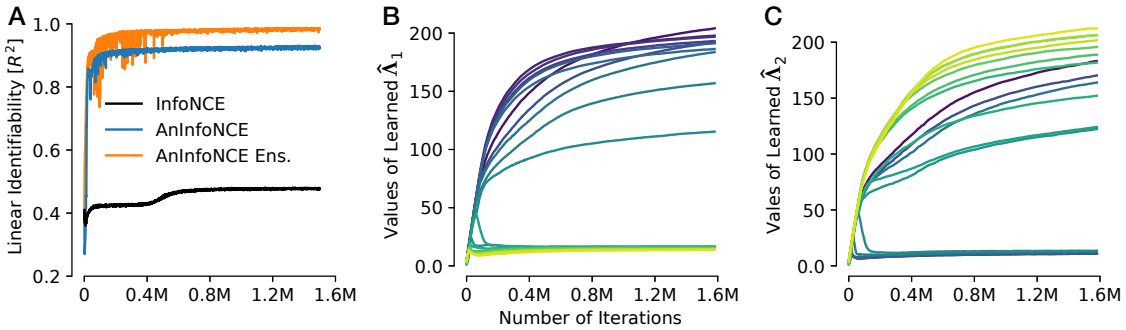


Figure 11: **Ensemble AnInfoNCE Improves Upon InfoNCE and AnInfoNCE:** We observe higher linear identifiability scores when training with Ensemble AnInfoNCE. When examining the learned $\hat{\Lambda}$ values, we see a specialization in the learned $\hat{\Lambda}$ values: The color scheme is chosen such that the same color indicates the same dimension, and we find that the learned $\hat{\Lambda}$-values for the two DGPs are anti-correlated with respect to each other.

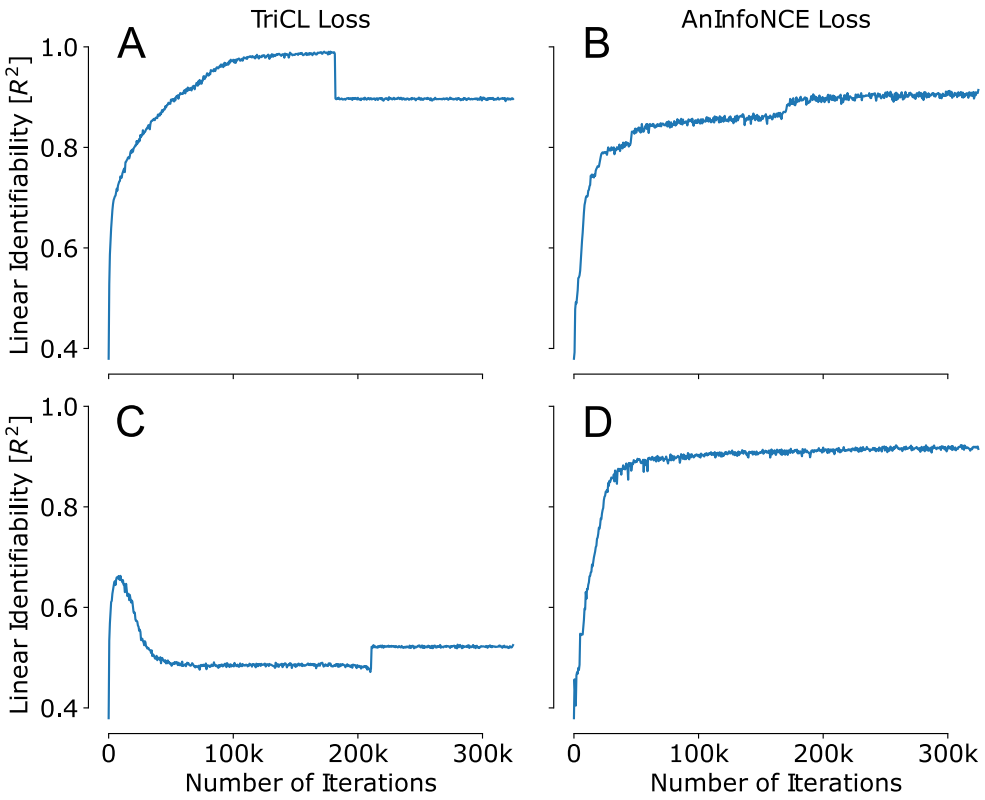


Figure 12: **Synthetic Experiments for a Latent Dimensionality of d=20 When Training With TriCL or AnInfoNCE.** Top row: Uniform Λ of 100. Bottom row: Dual partitioned Λ, such that half of the dimensions are varied according to a low Λ of 15 and the other half is varied with a high Λ of 250. In both cases, we observe that AnInfoNCE behaves much more stably compared to TriCL and reaches higher linear identifiability scores, especially when there is an anisotropic positive conditional.

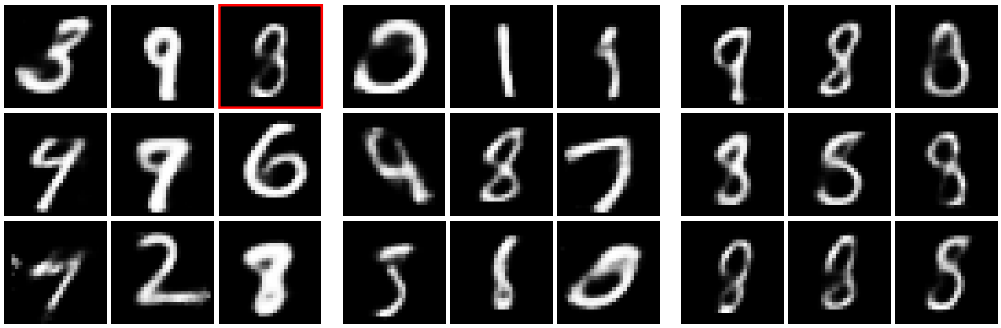


Figure 13: **Samples Generated by the MNIST-VAE**. **Left**: Samples from the marginal distribution, used as anchor samples for the contrastive objective. **Middle & Right**: For the highlighted sample, we show potential positive samples from the conditional distribution for two different values of $\mathbf{\Lambda}$. A value of 5 leads to too much variation (middle) and leads to obvious class changes in the conditional samples, whereas a value of 50 leads to reasonable variation. We note that training an encoder on images which were sampled according to a DGP with either ground-truth Λ-value leads to perfect linear disentanglement scores. In other words, a low concentration parameter of $\Lambda = 5$ which changes the class of the conditional view does not pose an issue.

arXiv:2006.16971v2 [cs.LG] 23 Oct 2020

Improving robustness against common corruptions by covariate shift adaptation

Steffen Schneider[*]
University of Tübingen &
IMPRS-IS

Evgenia Rusak[*]
University of Tübingen &
IMPRS-IS

Luisa Eck
LMU Munich

Oliver Bringmann[†]
University of Tübingen

Wieland Brendel[†]
University of Tübingen

Matthias Bethge[†]
University of Tübingen

Abstract

Today's state-of-the-art machine vision models are vulnerable to image corruptions like blurring or compression artefacts, limiting their performance in many real-world applications. We here argue that popular benchmarks to measure model robustness against common corruptions (like ImageNet-C) underestimate model robustness in many (but not all) application scenarios. The key insight is that in many scenarios, multiple unlabeled examples of the corruptions are available and can be used for unsupervised online adaptation. Replacing the activation statistics estimated by batch normalization on the training set with the statistics of the corrupted images consistently improves the robustness across 25 different popular computer vision models. Using the corrected statistics, ResNet-50 reaches 62.2% mCE on ImageNet-C compared to 76.7% without adaptation. With the more robust DeepAugment+AugMix model, we improve the state of the art achieved by a ResNet50 model up to date from 53.6% mCE to 45.4% mCE. Even adapting to a single sample improves robustness for the ResNet-50 and AugMix models, and 32 samples are sufficient to improve the current state of the art for a ResNet-50 architecture. We argue that results with adapted statistics should be included whenever reporting scores in corruption benchmarks and other out-of-distribution generalization settings.

1 Introduction

Deep neural networks (DNNs) are known to perform well in the independent and identically distributed (*i.i.d.*) setting when the test and training data are sampled from the same distribution. However, for many applications this assumption does not hold. In medical imaging, X-ray images or histology slides will differ from the training data if different acquisition systems are being used. In quality assessment, the images might differ from the training data if lighting conditions change or if dirt particles accumulate on the camera. Autonomous cars may face rare weather conditions like sandstorms or big hailstones. While human vision is quite robust to those deviations [1], modern machine vision models are often sensitive to such image corruptions.

We argue that current evaluations of model robustness underestimate performance in many (but not all) real-world scenarios. So far, popular image corruption benchmarks like ImageNet-C [IN-C; 2] focus only on ad hoc scenarios in which the tested model has zero prior knowledge about the corruptions it encounters during test time, even if it encounters the same corruption multiple times. In the example of medical images or quality assurance, the image corruptions do not change from sample to sample

[*]Equal contribution. [†] Equal contribution.; Online version and code: domainadaptation.org/batchnorm

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

but are continuously present over a potentially large number of samples. Similarly, autonomous cars will face the same weather condition over a continuous stream of inputs during the same sand- or hailstorm. These (unlabeled) observations can allow recognition models to adapt to the change in the input distribution.

Such unsupervised adaptation mechanisms are studied in the field of domain adaptation (DA), which is concerned with adapting models trained on one domain (the source, here clean images) to another for which only unlabeled samples exist (the target, here the corrupted images). Tools and methods from domain adaptation are thus directly applicable to increase model robustness against common corruptions, but so far no results on popular benchmarks have been reported. The overall goal of this work is to encourage stronger interactions between the currently disjoint fields of domain adaptation and robustness towards common corruptions.

We here focus on one popular technique in DA, namely adapting batch normalization [BN; 3] statistics [4–6]. In computer vision, BN is a popular technique for speeding up training and is present in almost all current state-of-the-art image recognition models. BN estimates the statistics of activations for the training dataset and uses them to normalize intermediate activations in the network.

By design, activation statistics obtained during training time do not reflect the statistics of the test distribution when testing in out-of-distribution settings like corrupted images. We investigate and corroborate the hypothesis that high-level distributional shifts from clean to corrupted images largely manifest themselves in a difference of first and second order moments in the internal representations of a deep network, which can be mitigated by adapting BN statistics, i.e. by estimating the BN statistics on the corrupted images. We demonstrate that this simple adaptation alone can greatly increase recognition performance on corrupted images.

Our contributions can be summarized as follows:

- We suggest to augment current benchmarks for common corruptions with two additional performance metrics that measure robustness after partial and full unsupervised adaptation to the corrupted images.
- We draw connections to domain adaptation and show that even adapting to a single corrupted sample improves the baseline performance of a ResNet-50 model trained on IN from 76.7% mCE to 71.4%. Robustness increases with more samples for adaptation and converges to a mCE of 62.2%.
- We show that the robustness of a variety of vanilla models trained on ImageNet [IN; 7, 8] substantially increases after adaptation, sometimes approaching the current state-of-the-art performance on IN-C without adaptation.
- Similarly, we show that the robustness of state-of-the-art ResNet-50 models on IN-C consistently increases when adapted statistics are used. We surpass the best non-adapted model (52.3% mCE) by almost 7% points.
- We show results on several popular image datasets and discuss both the generality and limitations of our approach.
- We demonstrate that the performance degradation of a non-adapted model can be well predicted from the Wasserstein distance between the source and target statistics. We propose a simple theoretical model for bounding the Wasserstein distance based on the adaptation parameters.

2 Measuring robustness against common corruptions

The ImageNet-C benchmark [2] consists of 15 test corruptions and four hold-out corruptions which are applied with five different severity levels to the $50\,000$ test images of the ILSVRC2012 subset of ImageNet [8]. During evaluation, model responses are assumed to be conditioned only on single samples, and are not allowed to adapt to e.g. a batch of samples from the same corruption. We call this the ad hoc or non-adaptive scenario. The main performance metric on IN-C is the mean corruption error (mCE), which is obtained by normalizing the model's top-1 errors with the top-1 errors of AlexNet [9] across the $C = 15$ test corruptions and $S = 5$ severities (cf. 2):

$$\text{mCE(model)} = \frac{1}{C}\sum_{c=1}^{C}\frac{\sum_{s=1}^{S}\text{err}_{c,s}^{\text{model}}}{\sum_{s=1}^{S}\text{err}_{c,s}^{\text{AlexNet}}}. \tag{1}$$

Note that mCE reflects only one possible averaging scheme over the IN-C corruption types. We additionally report the overall top-1 accuracies and report results for all individual corruptions in the supplementary material and the project repository.

In many application scenarios, this ad hoc evaluation is too restrictive. Instead, often many unlabeled samples with similar corruptions are available, which can allow models to adapt to the shifted data distribution. To reflect such scenarios, we propose to also benchmark the robustness of adapted models. To this end, we split the $50\,000$ validation samples with the same corruption and severity into batches with n samples each and allow the model to condition its responses on the complete batch of images. We then compute mCE and top-1 accuracy in the usual way.

We consider three scenarios: In the *ad hoc* scenario, we set $n = 1$ which is the typically considered setting. In the *full adaptation* scenario, we set $n = 50\,000$, meaning the model may adapt to the full set of unlabeled samples with the same corruption type before evaluation. In the *partial adaptation* scenario, we set $n = 8$ to test how efficiently models can adapt to a relatively small number of unlabeled samples.

3 Correcting Batch Normalization statistics as a strong baseline for reducing covariate shift induced by common corruptions

We propose to use a well-known tool from domain adaptation—adapting batch normalization statistics [5, 6]—as a simple baseline to increase robustness against image corruptions in the adaptive evaluation scenarios. IN trained models typically make use of batch normalization [BN; 3] for faster convergence and improved stability during training. Within a BN layer, first and second order statistics μ_c, σ_c^2 of the activation tensors $\mathbf{z}_c$ are estimated across the spatial dimensions and samples for each feature map c. The activations are then normalized by subtracting the mean μ_c and dividing by σ_c^2. During training, μ_c and σ_c^2 are estimated *per batch*. During evaluation, μ_c and σ_c^2 are estimated *over the whole training dataset*, typically using exponential averaging [10].

Using the BN statistics obtained during training for testing makes the model decisions deterministic but is also problematic if the input distribution changes. If the activation statistics μ_c, σ_c^2 change for samples from the test domain, then the activations of feature map c are no longer normalized to zero mean and unit variance, breaking a crucial assumption that all downstream layers depend on. Mathematically, this *covariate shift*[2] can be formalized as follows:

Definition 1 (Covariate Shift, cf. 12, 13)**.** *There exists covariate shift between a source distribution with density $p_s : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^+$ and a target distribution with density $p_t : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^+$, written as $p_s(\mathbf{x}, y) = p_s(\mathbf{x})p_s(y|\mathbf{x})$ and $p_t(\mathbf{x}, y) = p_t(\mathbf{x})p_t(y|\mathbf{x})$, if $p_s(y|\mathbf{x}) = p_t(y|\mathbf{x})$ and $p_s(\mathbf{x}) \neq p_t(\mathbf{x})$ where $y \in \mathcal{Y}$ denotes the class label.*

Removal of covariate shift. If covariate shift (Def. 1) only causes differences in the first and second order moments of the feature activations $\mathbf{z} = f(\mathbf{x})$, it can be removed by applying normalization:

$$p\left(\frac{f(\mathbf{x}) - \mathbb{E}_s[f(\mathbf{x})]}{\sqrt{\mathbb{V}_s[f(\mathbf{x})]}}\Big|\mathbf{x}\right) p_s(\mathbf{x}) \approx p\left(\frac{f(\mathbf{x}) - \mathbb{E}_t[f(\mathbf{x})]}{\sqrt{\mathbb{V}_t[f(\mathbf{x})]}}\Big|\mathbf{x}\right) p_t(\mathbf{x}). \tag{2}$$

Reducing the covariate shift in models with batch normalization is particularly straightforward: it suffices to estimate the BN statistics μ_t, σ_t^2 on (unlabeled) samples from the test data available for adaptation. If the number of available samples n is too small, the estimated statistics would be too unreliable. We therefore leverage the statistics μ_s, σ_s^2 already computed on the training dataset as a prior and infer the test statistics for each test batch as follows,

$$\bar{\mu} = \frac{N}{N+n}\mu_s + \frac{n}{N+n}\mu_t, \quad \bar{\sigma}^2 = \frac{N}{N+n}\sigma_s^2 + \frac{n}{N+n}\sigma_t^2. \tag{3}$$

[2]Note that our notion of internal covariate shift differs from previous work [3, 11]: In *i.i.d.* training settings, Ioffe and Szegedy [3] hypothesized that covariate shift introduced by changing lower layers in the network is reduced by BN, explaining the empirical success of the method. We do not provide evidence for this line of research in this work: Instead, we focus on the covariate shift introduced (by design) in datasets such as IN-C, and provide evidence for the hypothesis that high-level domain shifts in the input partly manifests in shifts and scaling of *internal* activations.

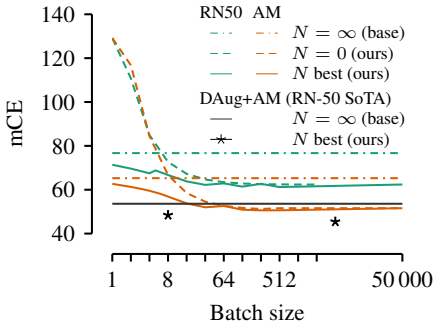


Figure 1: Sample size vs. performance tradeoff in terms of the mean corruption error (mCE) on IN-C for ResNet-50 and AugMix (AM). Black line corresponds to (non-adapted) ResNet50 state-of-the-art performance of DeepAug+AugMix.

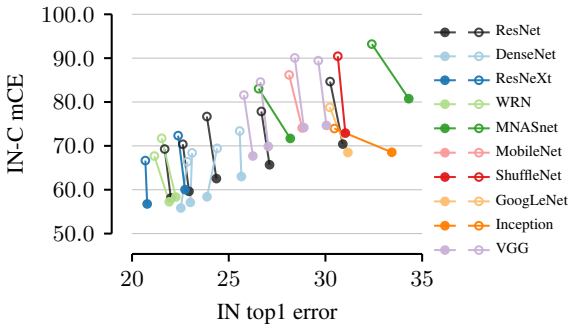


Figure 2: Across 25 model architectures in the `torchvision` library, the baseline mCE (○) improves with adaptation (●), often on the order of 10 points. Best viewed in color.

The hyperparameter N controls the trade-off between source and estimated target statistics and has the intuitive interpretation of a *pseudo sample size* (p. 117, 14) for samples from the training set. The case $N \to \infty$ ignores the test set statistics and is equivalent to the standard ad hoc scenario while $N = 0$ ignores the training statistics. Supported by empirical and theoretical results (see results section and appendix), we suggest using $N \in [8, 128]$ for practical applications with small $n < 32$.

4 Experimental Setup

Models. We consider a large range of models (cf. Table 2, §B,E) and evaluate pre-trained variants of DenseNet [15], GoogLeNet [16], Inception and GoogLeNet [17], MNASnet [18], MobileNet [19], ResNet [20], ResNeXt [21], ShuffleNet [22], VGG [23] and Wide Residual Network [WRN, 24] from the `torchvision` library [25]. All models are trained on the ILSVRC2012 subset of IN comprised of 1.2 million images in the training and a total of 1000 classes [7, 8]. We also consider a ResNeXt-101 variant pre-trained on a 3.5 billion image dataset and then fine-tuned on the IN training set [26]. We evaluate 3 models from the SimCLRv2 framework [27]. We additionally evaluate the four leading methods from the ImageNet-C leaderboard, namely Stylized ImageNet training [SIN; 28], adversarial noise training [ANT; 29] as well as a combination of ANT and SIN [29], optimized data augmentation using AutoAugment [AugMix; 30, 31] and Assemble Net [32]. For partial adaptation, we choose $N \in \{2^0, \cdots, 2^{10}\}$ and select the optimal value on the holdout corruption mCE.

Datasets. ImageNet-C [IN-C; 2] is comprised of corrupted versions of the 50 000 images in the IN validation set. The dataset offers five severities per corruption type, for a total of 15 "test" and 4 "holdout" corruptions. ImageNet-A [IN-A; 33] consists of unmodified real-world images which yield chance level classification performance in IN trained ResNet-50 models. ImageNet-V2 [IN-V2; 34] aims to mimic the test distribution of IN, with slight differences in image selection strategies. ObjectNet [ON; 35] is a test set containing 50 000 images like IN organized in 313 object classes with 109 unambiguously overlapping IN classes. ImageNet-R [IN-R; 36] contains 30 000 images with various artistic renditions of 200 classes of the original IN dataset. Additional information on the used models and datasets can be found in §B. For IN, we resize all images to 256×256px and take the center 224×224px crop. For IN-C, images are already cropped. We also center and re-scale the color values with $\mu_{RGB} = [0.485, 0.456, 0.406]$ and $\sigma = [0.229, 0.224, 0.225]$.

5 Results

Adaptation boosts robustness of a vanilla trained ResNet-50 model. We consider the pre-trained ResNet-50 architecture from the `torchvision` library and adapt the running mean and variance on all corruptions and severities of IN-C for different batch sizes. The results are displayed in Fig. 1 where different line styles of the green lines show the number of pseudo-samples N indicating the

Table 1: Adaptation improves mCE (lower is better) and Top1 accuracy (higher is better) on IN-C for different models and surpasses the previous state of the art without adaptation. We consider $n = 8$ for partial adaptation.

	IN-C mCE (↘)				Top1 accuracy (↗)			
Model	w/o adapt	partial adapt	full adapt	Δ	w/o adapt	partial adapt	full adapt	Δ
Vanilla ResNet-50	76.7	65.0	62.2	(−14.5)	39.2	48.6	50.7	(+11.5)
SIN [28]	69.3	61.5	59.5	(−9.8)	45.2	51.6	53.1	(+7.9)
ANT [29]	63.4	56.1	53.6	(−9.8)	50.4	56.1	58.0	(+7.6)
ANT+SIN [29]	60.7	55.3	53.6	(−7.0)	52.6	56.8	58.0	(+5.4)
AugMix [AM; 30]	65.3	55.4	51.0	(−14.3)	48.3	56.3	59.8	(+11.4)
Assemble Net [32]	52.3	–	50.1	(−1.2)	59.2	–	60.8	(+1.5)
DeepAug [36]	60.4	52.3	49.4	(−10.9)	52.6	59.0	61.2	(+8.6)
DeepAug+AM [36]	53.6	48.4	45.4	(−8.2)	58.1	62.2	64.5	(+6.4)
DeepAug+AM+RNXt101 [36]	**44.5**	40.7	**38.0**	(−6.6)	**65.2**	68.2	**70.3**	(+5.1)

influence of the prior given by the training statistics. With $N = 16$, we see that even adapting to a single sample can suffice to increase robustness, suggesting that even the ad hoc evaluation scenario can benefit from adaptation. If the training statistics are not used as a prior ($N = 0$), then it takes around 8 samples to surpass the performance of the non-adapted baseline model (76.7% mCE). After around 16 to 32 samples, the performance quickly converges to 62.2% mCE, considerably improving the baseline result. These results highlight the practical applicability of batch norm adaptation in basically all application scenarios, independent of the number of available test samples.

Adaptation consistently improves corruption robustness across IN trained models. To evaluate the interaction between architecture and BN adaptation, we evaluate all 25 pre-trained models in the `torchvision` package and visualize the results in Fig. 2. All models are evaluated with $N = 0$ and $n = 2000$. We group models into different families based on their architecture and observe consistent improvements in mCE for all of these families, typically on the order of 10% points. We observe that in both evaluation modes, DenseNets [15] exhibit higher corruption robustness despite having a comparable or even smaller number of trainable parameters than ResNets which are usually considered as the relevant baseline architecture. A take-away from this study is thus that model architecture alone plays a significant role for corruption robustness and the ResNet architecture might not be the optimal choice for practical applications.

Adaptation yields new state of the art on IN-C for robust models. We now investigate if BN adaptation also improves the most robust models on IN-C. The results are displayed in Table 1. All models are adapted using $n = 50\,000$ (vanilla) or $n = 4096$ (all other models) and $N = 0$. The performance of all models is considerably higher whenever the BN statistics are adapted. The DeepAugment+AugMix reaches a new state of the art on IN-C for a ResNet-50 architecture of 45.4% mCE. Evaluating the performance of AugMix over the number of samples for adaptation (Fig. 1, we find that as little as eight samples are sufficient to improve over AssembleNet [32], the current state-of-the-art ResNet-50 model on IN-C without adaptation. We have included additional results in §C.

6 Analysis and Ablation Studies

Severity of covariate shift correlates with performance degradation. The relationship between the performance degradation on IN-C and the covariate shift suggests an unsupervised way of estimating the classification performance of a model on a new corruption. Taking the normalized Wasserstein distance (cf. §A) between the statistics of the source and target domains[3] computed on all samples with the same corruption and severity and averaged across all network layers, we find a correlation with the top-1 error (Fig. 3 *i–iii*) of both non-adapted (*i*) and fully adapted model (*ii*) on IN-C corruptions. Within single corruption categories (noise, blur, weather, and digital), the relationship between top-1 error and Wasserstein distance is particularly striking: using linear

[3]For computing the Wasserstein metric we make the simplifying assumption that the empirical mean and covariances fully parametrize the respective distributions.

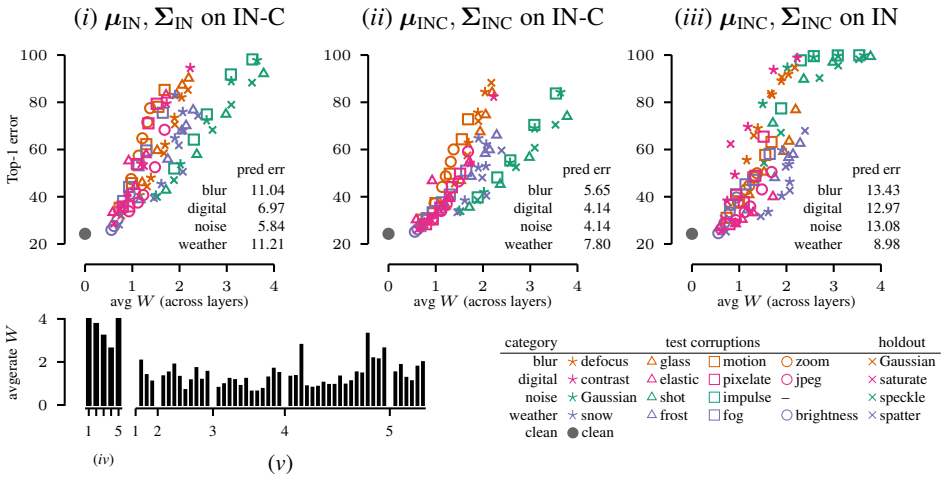


Figure 3: The Wasserstein metric between optimal source (IN) and target (IN-C) statistics correlates well with top-1 errors (*i*) of non-adapted models on IN-C, (*ii*) of adapted models on IN-C, indicating that even after reducing covariate shift, the metric is predictive of the remaining source–target mismatch (*iii*) IN-C adapted models on IN, the reverse case of (*i*). Holdout corruptions can be used to get a linear estimate on the prediction error of test corruptions (tables). We depict input and downsample (*iv*) as well as bottlneck layers (*v*) and notice the largest shift in early and late downsampling layers. The metric is either averaged across layers (*i–iii*) or across corruptions (*iv–v*).

Table 2: Improvements from adapting the BN parameters vanish for models trained with weakly supervised pre-training.

	IN-C mCE (↘)	
ResNeXt101	BN	BN+adapt
32x8d, IN	66.6	56.7 (−9.9)
32x8d, IG-3.5B	51.7	51.6 (−0.1)
32x48d, IG-3.5B	**45.7**	**47.3** (+1.6)

Table 3: Fixup and GN trained models perform better than non-adapted BN models but worse than adapted BN models.

	IN-C mCE (↘)			
Model	Fixup	GN	BN	BN+adapt
ResNet-50	72.0	72.4	76.7	**62.2**
ResNet-101	68.2	67.6	69.0	**59.1**
ResNet-152	67.6	65.4	69.3	**58.0**

regression, the top-1 accuracy of hold-out corruptions can be estimated with around 1–2% absolute mean deviation (cf. §C.5) within a corruption, and with around 5–15% absolute mean deviation when the estimate is computed on the holdout corruption of each category (see Fig. 3, typically, a systematic offset remains). In Fig. 3(*iv–v*), we display the Wasserstein distance across individual layers and observe that the covariate shift is particularly present in early and late downsampling layers of the ResNet-50.

Large scale pre-training alleviates the need for adaptation. Computer vision models based on the ResNeXt architecture [21] pretrained on a much larger dataset comprised of 3.5×10^9 Instagram images (IG-3.5B) achieve a 45.7% mCE on IN-C [26, 37]. We re-evaluate these models with our proposed paradigm and summarize the results in Table 2. While we see improvements for the small model pre-trained on IN, these improvements vanish once the model is trained on the full IG-3.5B dataset. This observation also holds for the largest model, suggesting that training on very large datasets might alleviate the need for covariate shift adaptation.

Group Normalization and Fixup Initialization performs better than non-adapted batch norm models, but worse than batch norm with covariate shift adaptation. So far, we considered image classification models with BN layers and concluded that using training dataset statistics in BN generally degrades model performance in out-of-distribution evaluation settings. We now consider models trained without BN and study the impact on corruption robustness, similar to Galloway et al. [38].

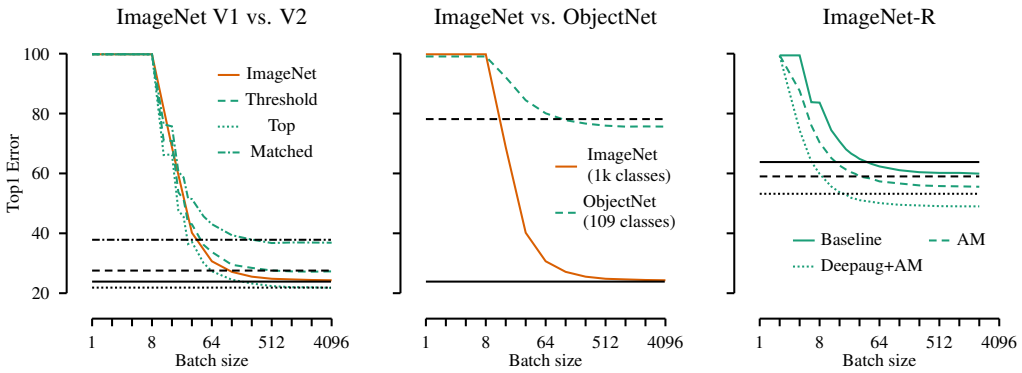


Figure 4: Batch size vs. performance trade-off for different natural image datasets with no covariate shift (IN, IN-V2), complex and shuffled covariate shift (ObjectNet), complex and systematic covariate shift (ImageNet-R). Straight black lines show baseline performance (no adaptation). ImageNet plotted for reference.

Table 4: GN and Fixup achieve the best results on ObjectNet (ON). After shuffling IN-C corruptions, BN adaptation does no longer decrease the error. Adaptation improves the performance of a vanilla ResNet50 on IN-R.

	ON		Mixed IN-C		IN-R
ResNet50	top-1	top-5	top-1	top-5	top-1
BN w/o adapt	78.2	60.9	61.1	40.8	63.8
BN w/ adapt	76.0	58.9	60.9	40.3	**59.9**
GroupNorm	**70.8**	**49.8**	57.3	36.0	61.2
Fixup	71.5	51.4	**56.8**	**35.4**	65.0

Table 5: Adaptation improves the performance (top-1 error) of robust models on IN-R (n=2048).

Model	base	adapt	Δ
ResNet50	63.8	59.9	-3.9
SIN	58.6	54.2	-4.4
ANT	61.0	58.0	-3.0
ANT+SIN	53.8	52.0	-1.8
AugMix (AM)	59.0	55.8	-3.2
DeepAug (DAug)	57.8	52.5	-5.3
DAug+AM	53.2	48.9	-4.3
DAug+AM+RNXt101	**47.9**	**44.0**	-3.9

First, using Fixup initialization [39] alleviates the need for BN layers. We train a ResNet-50 model on IN for 100 epochs to obtain a top-1 error of 24.2% and top-5 error of 7.6% (compared to 27.6% reported by Zhang et al. [39] with shorter training, and the 23.9% obtained by our ResNet-50 baseline trained with BN). The model obtains an IN-C mCE of 72.0% compared to 76.7% mCE of the vanilla ResNet-50 model and 62.2% mCE of our adapted ResNet-50 model (cf. Table 3). Additionally, we train a ResNet-101 and a ResNet-152 with Fixup initialization with similar results. Second, GroupNorm [GN; 40] has been proposed as a batch-size independent normalization technique. We train a ResNet-50, a ResNet-101 and a ResNet-152 architecture for 100 epochs and evaluate them on IN-C and find results very similar to Fixup.

Results on other datasets: IN-A, IN-V2, ObjectNet, IN-R We use $N = 0$ and vary n in all ablation studies in this subsection. The technique does not work for the case of "natural adversarial examples" of IN-A [33] and the error rate stays above 99%, suggesting that the covariate shift introduced in IN-A by design is more severe compared to the covariate shift of IN-C and can not be corrected by merely calculating the correct BN statistics. We are not able to increase performance neither on IN nor on IN-V2, since in these datasets, no domain shift is present by design (see Fig. 4). For ON, the performance increases slightly when computing statistics on more than 64 samples. In Table 4 (first and second column), we observe that the GroupNorm and Fixup models perform better than our BN adaptation scheme: while there is a dataset shift in ON compared to IN, BN adaptation is only helpful for *systematic* shifts across multiple inputs and this assumption is violated on ON. As a control experiment, we sample a dataset "Mixed IN-C" where we shuffle the corruptions and severities. In Table 4 (third and fourth column), we now observe that BN adaptation expectedly no longer improves performance. On IN-R, we achieve better results for the adapted model compared to the non-adapted model as well as the GroupNorm and Fixup models, see Table 4 (last column). Additionally, on IN-R, we decrease the top-1 error for a wide range of models through adaptation (see

Table 5). For IN-R, we observe performance improvements for the vanilla trained ResNet50 when using a sample size of larger than 32 samples for calculating the statistics (Fig. 4, right-most plot).

A model for correcting covariate shift effects. We evaluate how the batch size for estimating the statistics at test time affects the performance on IN, IN-V2, ON and IN-R in Fig. 4. As expected, for IN the adaptation to test time statistics converges to the performance of the train time statistics in the limit of large batch sizes, see Fig. 4 middle. For IN-V2, we find similar results, see Fig. 4 left. This observation shows that (*i*) there is no systematic covariate shift between the IN train set and the IN-V2 validation set that could be corrected by using the correct statistics and (*ii*) is further evidence for the *i.i.d.* setting pursued by the authors of IN-V2. In case of ON (Fig. 4 right), we see slight improvements when using a batch size bigger than 128.

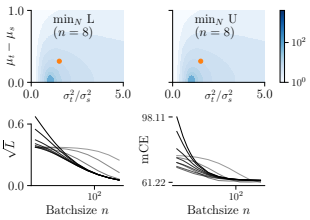


Figure 5: The bound suggests small optimal N for most parameters (i) and qualitatively explains our empirical observation (ii).

Choosing the number of pseudo-samples N offers an intuitive trade-off between estimating accurate target statistics (low N) and relying on the source statistics (large N). We propose a simple model to investigate optimal choices for N, disregarding all special structure of DNNs, and focusing on the statistical error introduced by estimating $\hat{\mu}_t$ and $\hat{\sigma}_t^2$ from a limited number of samples n. To this end, we estimate upper (U) and lower (L) bounds of the expected squared Wasserstein distance W_2^2 as a function of N and the covariate shift which provides good empirical fits between the estimated W and empirical performance for ResNet-50 for different N (Fig. 5; bottom row). Choosing N such that L or U are minimized (Fig. 5; example in top row) qualitatively matches the values we find, see §D for all details.

Proposition 1 (Bounds on the expected value of the Wasserstein distance between target and combined estimated target and source statistics)**.** *We denote the source statistics as μ_s, σ_s^2, the true target statistics as μ_t, σ_t^2 and the biased estimates of the target statistics as $\hat{\mu}_t, \hat{\sigma}_t^2$. For normalization, we take a convex combination of the source statistics and estimated target statistics as discussed in Eq. 3. At a confidence level $1-\alpha$, the expectation value of the Wasserstein distance $W_2^2(\bar{\mu}, \bar{\sigma}, \mu_t, \sigma_t)$ between ideal and estimated target statistics w.r.t. to the distribution of sample mean $\hat{\mu}_t$ and sample variance $\hat{\sigma}_t^2$ is bounded from above and below with $L \leq \mathbb{E}[W_2^2] \leq U$, where*

$$L = \left(\sigma_t - \sqrt{\frac{N}{N+n}\sigma_s^2 + \frac{n-1}{N+n}\sigma_t^2}\right)^2 + \frac{N^2}{(N+n)^2}\left(\mu_t - \mu_s\right)^2 + \frac{n}{(N+n)^2}\sigma_t^2$$

$$U = L + \sigma_t^5 \frac{(n-1)}{2(N+n)^2}\left(\frac{N}{N+n}\sigma_s^2 + \frac{1}{N+n}\chi^2_{1-\alpha/2,n-1}\sigma_t^2\right)^{-3/2}$$

The quantity $\chi^2_{1-\alpha/2,n-1}$ denotes the left tail value of a chi square distribution with $n-1$ degrees of freedom, defined as $P\left(X \leq \chi^2_{1-\alpha/2,n-1}\right) = \alpha/2$ for $X \sim \chi^2_{n-1}$. Proof: *See Appendix §D.*

7 Related Work

The IN-C benchmark [2] has been extended to MNIST [41], several object detection datasets [42] and image segmentation [43] reflecting the interest of the robustness community. Most proposals for improving robustness involve special training protocols, requiring time and additional resources. This includes data augmentation like Gaussian noise [44], optimized mixtures of data augmentations in conjunction with a consistency loss [30], training on stylized images [28, 42, 45] or against adversarial noise distributions [29]. Other approaches tweak the architecture, e.g. by adding shift-equivariance with an anti-aliasing module, [46] or assemble different training techniques [32].

Unsupervised domain adaptation (DA) is a form of transductive inference where additional information about the test dataset is used to adapt a model to the test distribution. Adapting feature statistics was proposed by Sun et al. [47] and follow up work evaluated the performance of adapting BN parameters in unsupervised [5, 6] and supervised DA settings [4]. As an application example in medical imaging, Bug et al. [48] show that adaptive normalization is useful for removing domain

shifts on histopathological data. More involved methods for DA include self-supervised domain adaptation on single examples [49] and pseudo-labeling French et al. [50]. Xie et al. [51] achieve the state of the art on IN-C with pseudo-labeling. In work concurrent to ours, Wang et al. [52] also show BN adaptation results on IN-C. They also perform experiments on CIFAR10-C and CIFAR100-C and explore other domain adaptation techniques.

Robustness scores obtained by adversarial training can be improved when separate BN or GroupNorm layers are used for clean and adversarial images [53]. The expressive power of adapting only affine BN parameters BN parameters was shown in multi-task [54] and DA contexts [4] and holds even for fine-tuning randomly initialized ResNets [55]. Concurrent work shows additional evidence that BN adaptation yields increased performance on ImageNet-C [56].

8 Discussion and Conclusion

We showed that reducing covariate shift induced by common image corruptions improves the robustness of computer vision models trained with BN layers, typically by 10–15% points (mCE) on IN-C. Current state-of-the-art models on IN-C can benefit from adaptation, sometimes drastically like AugMix (-14% points mCE). This observation underlines that current benchmark results on IN-C underestimate the corruption robustness that can be reached in many application scenarios where additional (unlabeled) samples are available for adaptation.

Robustness against common corruptions improves even if models are adapted only to a single sample, suggesting that BN adaptation should always be used whenever we expect machine vision algorithms to encounter out-of-domain samples. Most further improvements can be reaped by adapting to 32 to 64 samples, after which additional improvements are minor.

Our empirical results suggest that the performance degradation on corrupted images can mostly be explained by the difference in feature-wise first and second order moments. While this might sound trivial, the performance could also degrade because models mostly extract features susceptible to common corruptions [57], which could not be fixed without substantially adapting the model weights. The fact that model robustness increases after correcting the BN statistics suggests that the features upon which the models rely on are still present in the corrupted images. The opposite is true in other out-of-domain datasets like IN-A or ObjectNet where our simple adaptation scheme does not substantially improve performance, suggesting that here the main problem is in the features that models have learned to use for prediction.

Batch Norm itself is not the reason why models are susceptible to common corruptions. While alternatives like Group Normalization and Fixup initialization slightly increase robustness, the adapted BN models are still substantially more robust. This suggests that non-BN models still experience an internal covariate shift on corrupted images, but one that is now absorbed by the model parameters instead of being exposed in the BN layers, making it harder to fix.

Large-scale pre-training on orders of magnitude more data (like IG-3.5B) can remove the first- and second-order covariate shift between clean and corrupted image samples, at least partially explaining why models trained with weakly supervised training [26] generalize so well to IN-C.

Current corruption benchmarks emphasize ad hoc scenarios and thus focus and bias future research efforts on these constraints. Unfortunately, the ad hoc scenario does not accurately reflect the information available in many machine vision applications like classifiers in medical computer vision or visual quality inspection algorithms, which typically encounter a similar corruption continuously and could benefit from adaptation. This work is meant to spark more research in this direction by suggesting two suitable evaluation metrics—which we strongly suggest to include in all future evaluations on IN-C—as well as by highlighting the potential that even a fairly simple adaptation mechanism can have for increasing model robustness. We envision future work to also adopt and evaluate more powerful domain adaptation methods on IN-C and to develop new adaptation methods specifically designed to increase robustness against common corruptions.

Broader Impact

The primary goal of this paper is to increase the robustness of machine vision models against common corruptions and to spur further progress in this area. Increasing the robustness of machine vision

systems can enhance their reliability and safety, which can potentially contribute to a large range of use cases including autonomous driving, manufacturing automation, surveillance systems, health care and others. Each of these uses may have a broad range of societal implications: autonomous driving can increase mobility of the elderly and enhance safety, but could also enable more autonomous weapon systems. Manufacturing automation can increase resource efficiency and reduce costs for goods, but may also increase societal tension through job losses or increase consumption and thus waste. Of particular concern (besides surveillance) is the use of generative vision models for spreading misinformation or for creating an information environment of uncertainty and mistrust.

We encourage further work to understand the limitations of machine vision models in out-of-distribution generalization settings. More robust models carry the potential risk of automation bias, i.e., an undue trust in vision models. However, even if models are robust to common corruptions, they might still quickly fail on slightly different perturbations like surface reflections. Understanding under what conditions model decisions can be deemed reliable or not is still an open research question that deserves further attention.

Acknowledgments and Disclosure of Funding

We thank Julian Bitterwolf, Roland S. Zimmermann, Lukas Schott, Mackenzie W. Mathis, Alexander Mathis, Asim Iqbal, David Klindt, Robert Geirhos, other members of the Bethge and Mathis labs and four anonymous reviewers for helpful suggestions for improving our manuscript and providing ideas for additional ablation studies. We thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting E.R. and St.S.; St.S. acknowledges his membership in the European Laboratory for Learning and Intelligent Systems (ELLIS) PhD program. This work was supported by the German Federal Ministry of Education and Research (BMBF) through the Tübingen AI Center (FKZ: 01IS18039A), by the Deutsche Forschungsgemeinschaft (DFG) in the priority program 1835 under grant BR2321/5-2 and by SFB 1233, Robust Vision: Inference Principles and Neural Mechanisms (TP3), project number: 276693517. The authors declare no conflicts of interests.

References

[1] Robert Geirhos, Carlos R. M. Temme, Jonas Rauber, Heiko H. Schütt, Matthias Bethge, and Felix A. Wichmann. Generalisation in humans and deep neural networks. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 7538–7550. Curran Associates, Inc., 2018. URL `http://papers.nips.cc/paper/7982-generalisation-in-humans-and-deep-neural-networks.pdf`.

[2] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations (ICLR)*, 2019.

[3] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning (ICLR)*, 2015.

[4] Steffen Schneider, Alexander S Ecker, Jakob H Macke, and Matthias Bethge. Multi-task generalization and adaptation between noisy digit datasets: An empirical study. In *Neural Information Processing Systems (NeurIPS), Workshop on Continual Learning*, 2018.

[5] Fabio Maria Cariucci, Lorenzo Porzi, Barbara Caputo, Elisa Ricci, and Samuel Rota Bulo. Autodial: Automatic domain alignment layers. In *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017.

[6] Yanghao Li, Naiyan Wang, Jianping Shi, Jiaying Liu, and Xiaodi Hou. Revisiting batch normalization for practical domain adaptation. In *International Conference on Machine Learning (ICLR)*, 2017.

[7] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision (IJCV)*, 2015.

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Conference on computer vision and pattern recognition (CVPR)*, 2009.

[9] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[10] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in PyTorch. In *NIPS Autodiff Workshop*, 2017.

[11] Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry. How does batch normalization help optimization? In *Advances in Neural Information Processing Systems (NIPS)*, 2018.

[12] Masashi Sugiyama and Motoaki Kawanabe. *Machine learning in non-stationary environments: Introduction to covariate shift adaptation*. MIT press, 2012.

[13] Bernhard Schölkopf, Dominik Janzing, Jonas Peters, Eleni Sgouritsa, Kun Zhang, and Joris Mooij. On causal and anticausal learning. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, ICML'12, page 459–466, Madison, WI, USA, 2012. Omnipress. ISBN 9781450312851.

[14] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006. ISBN 0387310738.

[15] Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. Densely connected convolutional networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[16] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[17] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Conference on computer vision and pattern recognition (CVPR)*, 2016.

[18] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V Le. Mnasnet: Platform-aware neural architecture search for mobile. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[19] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Conference on computer vision and pattern recognition (CVPR)*, 2018.

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Conference on computer vision and pattern recognition (CVPR)*, 2016.

[21] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Conference on computer vision and pattern recognition (CVPR)*, 2017.

[22] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[23] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations (ICLR)*, 2015.

[24] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *CoRR*, abs/1605.07146, 2016.

[25] Sébastien Marcel and Yann Rodriguez. Torchvision the machine-vision package of torch. In *ACM International Conference on Multimedia*, 2010.

[26] Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens van der Maaten. Exploring the limits of weakly supervised pretraining. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[27] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey Hinton. Big self-supervised models are strong semi-supervised learners. *CoRR*, abs/2006.10029, 2020.

[28] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A. Wichmann, and Wieland Brendel. Imagenet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In *International Conference on Learning Representations (ICLR)*, 2019.

[29] Evgenia Rusak, Lukas Schott, Roland Zimmermann, Julian Bitterwolf, Oliver Bringmann, Matthias Bethge, and Wieland Brendel. Increasing the robustness of dnns against image corruptions by playing the game of noise. *CoRR*, abs/2001.06057, 2020.

[30] Dan Hendrycks, Norman Mu, Ekin D Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. In *International Conference on Learning Representations (ICLR)*, 2020.

[31] Ekin Dogus Cubuk, Barret Zoph, Dandelion Mané, Vijay Vasudevan, and Quoc V. Le. Autoaugment: Learning augmentation policies from data. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[32] Jungkyu Lee, Taeryun Won, and Kiho Hong. Compounding the performance improvements of assembled techniques in a convolutional neural network. *CoRR*, abs/2001.06268, 2020.

[33] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. *CoRR*, abs/1907.07174, 2019.

[34] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[35] Andrei Barbu, David Mayo, Julian Alverio, William Luo, Christopher Wang, Dan Gutfreund, Josh Tenenbaum, and Boris Katz. Objectnet: A large-scale bias-controlled dataset for pushing the limits of object recognition models. In *Advances in Neural Information Processing Systems 32*, 2019.

[36] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. *CoRR*, abs/2006.16241, 2020.

[37] A Emin Orhan. Robustness properties of facebook's resnext wsl models. *CoRR*, abs/1907.07640, 2019.

[38] Angus Galloway, Anna Golubeva, Thomas Tanay, Medhat Moussa, and Graham W Taylor. Batch normalization is a cause of adversarial vulnerability. *CoRR*, abs/1905.02161, 2019.

[39] Hongyi Zhang, Yann N Dauphin, and Tengyu Ma. Fixup initialization: Residual learning without normalization. *CoRR*, abs/1901.09321, 2019.

[40] Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[41] Norman Mu and Justin Gilmer. MNIST-C: A robustness benchmark for computer vision. *CoRR*, abs/1906.02337, 2019.

[42] Claudio Michaelis, Benjamin Mitzkus, Robert Geirhos, Evgenia Rusak, Oliver Bringmann, Alexander S Ecker, Matthias Bethge, and Wieland Brendel. Benchmarking robustness in object detection: Autonomous driving when winter is coming. *CoRR*, abs/1907.07484, 2019.

[43] Christoph Kamann and Carsten Rother. Benchmarking the robustness of semantic segmentation models. *CoRR*, abs/1908.05005, 2019.

[44] Nic Ford, Justin Gilmer, Nicolas Carlini, and Dogus Cubuk. Adversarial examples are a natural consequence of test error in noise. In *International Conference on Machine Learning (ICML)*, 2019.

[45] Agnieszka Mikołajczyk and Michał Grochowski. Data augmentation for improving deep learning in image classification problem. In *International Interdisciplinary PhD Workshop (IIPhDW)*, 2018.

[46] Richard Zhang. Making convolutional networks shift-invariant again. *International Conference on Machine Learning (ICML)*, 2019.

[47] Baochen Sun, Jiashi Feng, and Kate Saenko. Correlation alignment for unsupervised domain adaptation. In *Domain Adaptation in Computer Vision Applications*, pages 153–171. Springer, 2017.

[48] Daniel Bug, Steffen Schneider, Anne Grote, Eva Oswald, Friedrich Feuerhake, Julia Schüler, and Dorit Merhof. Context-based normalization of histological stains using deep convolutional features. In *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*. Springer, 2017.

[49] Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei A Efros, and Moritz Hardt. Test-time training for out-of-distribution generalization. *CoRR*, abs/1909.13231, 2019.

[50] Geoffrey French, Michal Mackiewicz, and Mark H. Fisher. Self-ensembling for domain adaptation. *CoRR*, abs/1706.05208, 2017.

[51] Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. Self-training with noisy student improves imagenet classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10687–10698, 2020.

[52] Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. Fully test-time adaptation by entropy minimization. *CoRR*, abs/2006.10726, 2020.

[53] Cihang Xie and Alan L. Yuille. Intriguing properties of adversarial training. In *International Conference on Learning Representations (ICLR)*, 2020.

[54] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. Learning multiple visual domains with residual adapters. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

[55] Jonathan Frankle, David J Schwab, and Ari S Morcos. Training batchnorm and only batchnorm: On the expressive power of random features in cnns. *CoRR*, abs/2003.00152, 2020.

[56] Zachary Nado, Shreyas Padhy, D Sculley, Alexander D'Amour, Balaji Lakshminarayanan, and Jasper Snoek. Evaluating prediction-time batch normalization for robustness under covariate shift. *CoRR*, abs/2006.10963, 2020.

[57] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. Shortcut learning in deep neural networks. *CoRR*, abs/2004.07780, 2020.

[58] Cédric Villani. *Optimal transport: old and new*, volume 338. Springer Science & Business Media, 2008.

[59] Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Jacob Steinhardt, and Aleksander Madry. Identifying statistical bias in dataset replication. *CoRR*, abs/2005.09619, 2020.

[60] Dirk Merkel. Docker: Lightweight linux containers for consistent development and deployment. *Linux J.*, 2014(239), March 2014. ISSN 1075-3583.

[61] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, CJ Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake Vand erPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1. 0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi: https://doi.org/10.1038/s41592-019-0686-2.

[62] O. Tange. Gnu parallel - the command-line power tool. *;login: The USENIX Magazine*, 36(1):42–47, Feb 2011. URL `http://www.gnu.org/s/parallel`.

[63] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pages 265–283, 2016.

[64] Ji Lin. *A PyTorch Converter for SimCLR Checkpoints*, 2020 (accessed October 21, 2020). URL `https://github.com/tonylins/simclr-converter`. Commit ID: 139d3cb0bd0c64b5ad32aab810e0bd0a0dddaae0.

[65] Eric Weisstein. Standard deviation distribution, 2020. URL `https://mathworld.wolfram.com/StandardDeviationDistribution.html`.

[66] Robert A. Becker. The variance drain and jensen's inequality. 2012-004, 2012.

[67] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NIPS)*. 2012.

Supplementary Material

A Distances and divergences for quantifying domain shift

Besides analyzing the performance drop when evaluating a model using source statistics on a target dataset, we consider the mismatch in model statistics directly. We first take an ImageNet trained model and adapt it to each of the 95 conditions in IN-C. To obtain a more exact estimate of the true statistics, we split the model into multiple stages with only few BN layers per stage and apply the following simple algorithm[4]:

- Start with image inputs $\mathbf{z}_n^0 \leftarrow \mathbf{x}_n$ from the validation set to adapt to, for each $n \in [50000]$.
- Split the model into multiple stages, $h(\mathbf{x}) = (f_m \circ \cdots \circ f_1)(\mathbf{x})$, where each module f_i can potentially contain one or multiple BN layers. We denote the number of BN layers in the i-th module as b_i.
- For each stage $i \in [m]$, repeat b_i times: $\mathbf{z}_n^i \leftarrow f_i(\mathbf{z}_n^{i-1})$ for each n, and update the BN statistics in module $f_i(\mathbf{z}_n^{i-1})$.
- Return h with adapted statistics.

Using this scheme, we get source statistics μ_s and Σ_s for each layer and μ_t and Σ_t for each layer and corruption. In total, we get 96 different collections of statistics across network layers (for IN and the 95 conditions in IN-C). For simplicity, we will not further index the statistics. Note that all covariance matrices considered here are diagonal, which is a further simplification. We expect that our domain shift estimates could be improved by considering the full covariance matrices.

In the following, we will introduce three possible distances and divergences which can be applied between source and target statistics to quantify the effect of common corruptions induced covariate shift. We consider the Wasserstein distance, a normalized version of the Wasserstein distance, and the Jeffrey divergence.

A.1 The Wasserstein distance

Given a baseline ResNet-50 model with source statistics μ_s, Σ_s on IN, the Wasserstein distance (cf. 58) between the train and test distribution with statistics μ_t, Σ_t is given as

$$W_2(p_s, p_t)^2 = \|\boldsymbol{\mu}_s - \boldsymbol{\mu}_t\|_2^2 + \operatorname{tr}\left(\boldsymbol{\Sigma}_s + \boldsymbol{\Sigma}_t - 2\left(\boldsymbol{\Sigma}_t^{1/2}\boldsymbol{\Sigma}_s\boldsymbol{\Sigma}_t^{1/2}\right)^{1/2}\right). \tag{4}$$

A.2 The source-normalized Wasserstein distance

When estimated for multiple layers across the network, the Wasserstein distance between source and target depends on the overall magnitude of the statistics. Practically, this means the metric is dominated by features with large magnitude (e.g. in the first layer of a neural network, which receives larger inputs).

To mitigate this issue, we normalize both statistics with the source statistics and define the normalized Wasserstein distance as

$$\widetilde{W}_2^2 = W_2^2\left(\boldsymbol{\Sigma}_s^{-1/2}\boldsymbol{\mu}_s, \mathbf{I}, \boldsymbol{\Sigma}_s^{-1/2}\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_s^{-1}\boldsymbol{\Sigma}_t\right) \tag{5}$$

$$= \operatorname{Tr}\left(\mathbf{I} + \boldsymbol{\Sigma}_t\boldsymbol{\Sigma}_s^{-1} - 2\boldsymbol{\Sigma}_t^{1/2}\boldsymbol{\Sigma}_s^{-1/2}\right) + (\boldsymbol{\mu}_t - \boldsymbol{\mu}_s)^T\boldsymbol{\Sigma}_s^{-1}(\boldsymbol{\mu}_t - \boldsymbol{\mu}_s). \tag{6}$$

In the uni-variate case, the normalized Wasserstein distance $\widetilde{W}_2^2$ is equal to the Wasserstein distance W_2^2 between source and target statistics divided by σ_s^2:

$$\widetilde{W}_2^2 = W_2^2\left(\frac{\mu_s}{\sigma_s}, 1, \frac{\mu_t}{\sigma_s}, \frac{\sigma_t^2}{\sigma_s^2}\right) = 1 + \frac{\sigma_t^2}{\sigma_s^2} - 2\frac{\sigma_t}{\sigma_s} + \frac{(\mu_t - \mu_s)^2}{\sigma_s^2} = \frac{1}{\sigma_s^2}W_2^2(\mu_s, \sigma_s^2, \mu_t, \sigma_t^2). \tag{7}$$

[4]Note that for simplicity, we do not reset the statistics of the remaining $(b_i - i)$ BN layers. This could potentially be adapted in future work.

A.3 The Jeffrey divergence

The Jeffrey divergence $J(p_s, p_t)$ between source distribution p_s and target distribution p_t is the symmetrized version of the Kullback-Leibler divergence D_{KL}:

$$J(p_s, p_t) = \frac{1}{2} \left(D_{KL}(p_s \| p_t) + D_{KL}(p_t \| p_s) \right) \tag{8}$$

The Kullback-Leibler divergence between the D-dimensional multivariate normal source and target distributions is defined as

$$D_{KL}(\mathcal{N}_t \| \mathcal{N}_s) = \frac{1}{2} \left(\mathrm{Tr}\left(\boldsymbol{\Sigma}_s^{-1} \boldsymbol{\Sigma}_t \right) + (\boldsymbol{\mu}_s - \boldsymbol{\mu}_t)^\top \boldsymbol{\Sigma}_s^{-1} (\boldsymbol{\mu}_s - \boldsymbol{\mu}_t) - D + \ln \left(\frac{\det \boldsymbol{\Sigma}_s}{\det \boldsymbol{\Sigma}_t} \right) \right). \tag{9}$$

The Jeffrey divergence between the D-dimensional multivariate normal source and target distributions then follows as

$$J(\mathcal{N}_t, \mathcal{N}_s) = \frac{1}{4} \left(\mathrm{Tr}\left(\boldsymbol{\Sigma}_s^{-1} \boldsymbol{\Sigma}_t \right) + \mathrm{Tr}\left(\boldsymbol{\Sigma}_t^{-1} \boldsymbol{\Sigma}_s \right) + (\boldsymbol{\mu}_s - \boldsymbol{\mu}_t)^\top \left(\boldsymbol{\Sigma}_s^{-1} + \boldsymbol{\Sigma}_t^{-1} \right) (\boldsymbol{\mu}_s - \boldsymbol{\mu}_t) - 2D \right). \tag{10}$$

A.4 Summary statistics and quantification of covariate shift between different IN-C conditions

Given the 95 distances/divergences between the baseline (IN) statistics and 95 IN-C conditions, we first perform a layer-wise analysis of the statistics and depict the results in Figure 6. The unnormalized Wasserstein distance is sensitive to the magnitude of the source statistics and hence differs qualitatively from the results on the normalized Wasserstein distance and Jeffrey Divergence. We appreciate that the most notable difference between source and target domains is visible in the ResNet-50 downsampling layers. All three metrics suggest that the shift is mainly present in the first and final layers of the network, supporting the hypothesis that within the common corruption dataset, we have both superficial covariate shift which can be corrected by simple means (such as brightness or contrast variations) in the first layers, and also more "high-level" domain shifts which can only be corrected in the later layers of the network.

In Figure 7, we more closely analyze this relationship for different common corruptions. We can generally appreciate the increased measures as the corruption severity increases.

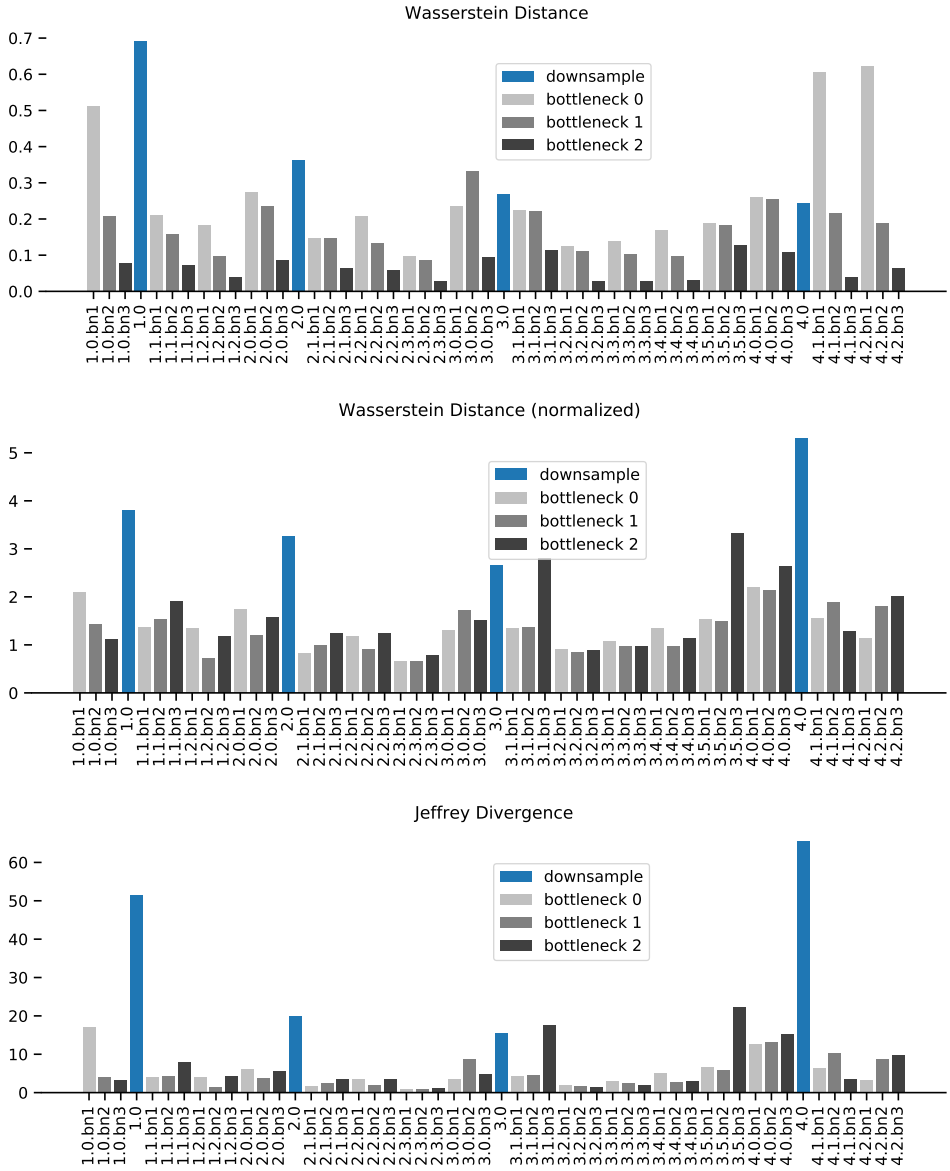


Figure 6: Wasserstein distance, normalized Wasserstein distance and Jeffrey divergence estimated among source and target statistics between different network layers. We report the respective metric w.r.t. to the difference between baseline (IN) and target (IN-C) statistics and show the value averaged across all corruptions. We note that for a ResNet-50 model, downsampling layers contribute most to the overall error.

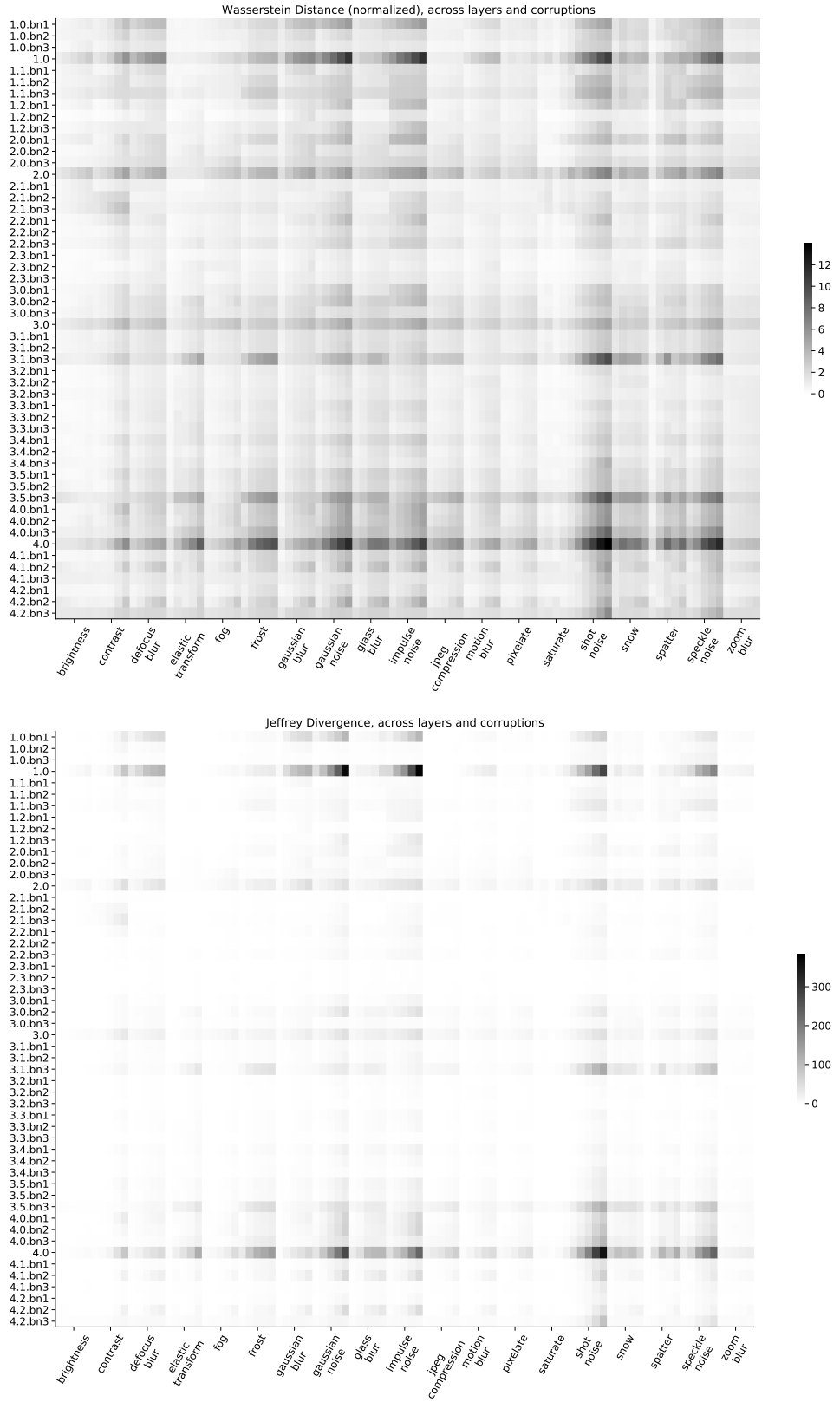


Figure 7: Normalized Wasserstein distance and Jeffrey divergence across corruptions and layers in a ResNet-50.

B Notes on the experimental setup

B.1 Practical considerations for implementing the method

Our method is conceptually very easy to implement. We generally recommend to first explore the easier variant of the algorithm where $N = 0$, i.e., no source statistics are used. As shown in our experiments, this setting works well if 100 or more target samples are available.

In this case, implementing the method boils down to enabling the training mode for all BN layers across the network. We will discuss this option along with two variants important for application to practical problems: Using exponential moving averaging (EMA) to collect target statistics across multiple batches, and using the source statistics as a prior.

Example implementation in PyTorch and caveats We encourage authors of robust models to always evaluate their models, and in particular baseline algorithms on both the train and test set statistics. Implementation in both PyTorch, Tensorflow and other machine learning libraries is straightforward and adds only minimal overhead. For PyTorch, adaptation is possible by simply adding

```
def use_test_statistics(module):
  if isisinstance(module, nn._BatchNorm):
    module.train()
model.eval()
model.apply(use_test_statistics)
```

before starting a model evaluation. For the adaptation to a full dataset, we provide a reference implementation with the source code release of this paper. Also, in contrast to the convention of not shuffling examples during test time, *make sure to enable dataset shuffling also during test time* in order to compute the correct statistics marginalized over class assignment.

Exponential moving averaging In practice, it might be beneficial to keep track of samples already encountered and use a running mean and variance on the test set to normalize new samples. We can confirm that this technique closely matches the full-dataset adaptation case even when evaluating with batch size 1 and is well suited for settings with less powerful hardware, or in general settings where access to the full batch of samples is not possible. Variants of this technique include the adaptation of the decay factor to discard statistics of samples encountered in the past (e.g. when the data domain slowly drifts over time).

B.2 Notes on models

Note that we only re-evaluate existing model checkpoints, and hence do not perform any hyperparameter tuning or adaptations to model training except for selecting the pseudo batchsize N for the source domain. Depending on the batch size and the architecture, model evaluations are done on one to eight Nvidia RTX 2080 GPUs (i.e., using 12 to 96 GB of memory) or up to four Nvidia V100 GPUs (128 GB of memory). Since we merely re-evaluate trained models, it is also possible to work on less powerful hardware with less memory. In these cases, the aggregation of batch normalization statistics has to be done across several batches using a variant of EMA.

B.3 Hyperparameter tuning

Our method is generally parameter-free if only target statistics should be considered for normalization. This approach is generally preferred for larger batch sizes n and should also be adapted in practice when a sufficient amount of samples is available. For tuning N, we consider the pre-defined holdout corruptions in IN-C, including speckle noise, saturation, Gaussian blur and spatter using a grid search across different values for N.

B.4 Notes on datasets

In the main paper, we have used several datasets and provide more relevant information here:

ImageNet-C (IN-C) For the evaluation on IN-C, we use the JPEG compressed images from github.com/hendrycks/robustness as is advised by the authors to ensure reproducibility. We note that Ford et al. [44] report a decrease in performance when the compressed JPEG files are used as opposed to applying the corruptions directly in memory without compression artefacts.

ObjectNet (ON) We find that there are 9 classes with multiple possible mappings from ON to IN (see the list in Table 6); we discard these classes in our evaluation. Models trained on IN experience a large performance drop on the order of 40–45% when tested on ON. ON is an interesting test case for unsupervised domain adaptation since IN and ON are likely sampled from different distributions. ON intentionally shows objects from new viewpoints on new backgrounds.

ImageNet-V2 (IN-V2) There are three test sets in IN-V2 that differ in *selection frequencies* of the MTurk workers. The selection frequency is given by the fraction of MTurk workers who selected an image for its target class. For the "MatchedFrequency" dataset, images were sampled according to the estimated selection frequency of sampling of the original IN validation dataset. For the "Threshold0.7" variant of IN-V2, images were sampled with a selection frequency of at least 0.7. The "TopImages" was sampled from images with the highest selection frequency. Although all three test sets were sampled from the same Flickr candidate pool and were labeled correctly and selected by more than 70% of MTurk workers, the model accuracies on these datasets vary by 14%. The authors observe a systematic accuracy drop when comparing model performance on the original IN validation set and IN-V2 and attribute it to the distribution gap between their datasets and the original IN dataset. They quantify the distribution gap by how much the change from the original distribution to the new distribution affects the considered model. Engstrom et al. analyze the creation process of IN-V2 and identify statistical bias resulting from noisy readings of the selection frequency statistic as a main source of dropping performance [59]. After correcting the bias, [59] find that the accuracy drop between IN and IN-V2 measures only 3.6% ± 1.5% of the original 11.7% ± 1.0%.

ON class	IN classes
wheel	wheel; paddlewheel, paddle wheel
helmet	football helmet; crash helmet
chair	barber chair; folding chair; rocking chair, rocker
still_camera	Polaroid camera, Polaroid Land camera; reflex camera
alarm_clock	analog clock; digital clock
tie	bow tie, bow-tie, bowtie; Windsor tie
pen	ballpoint, ballpoint pen, ballpen, Biro; quill, quill pen; fountain pen
bicycle	mountain bike, all-terrain bike, off-roader; bicycle-built-for-two, tandem bicycle, tandem
skirt	hoopskirt, crinoline; miniskirt, mini; overskirt

Table 6: Mapping between 9 ambiguous ON classes and the possible correspondences in IN. Different IN classes are separated with a semicolon.

B.5 Overview of models in torchvision

In Table 7, we provide a list of the models we evaluate in the main paper, along with numbers of trainable parameters and BN parameters. Note that the fraction of BN parameters is at most at 1% compared to all trainable parameters in all considered models.

B.6 Baseline corruption errors

In Table 8, we report the scores used for converting top-1 error into the mean corruption error (mCE) metric proposed by Hendrycks and Dietterich [2].

B.7 Software stack

We use various open source software packages for our experiments, most notably Docker [60], scipy and numpy [61], GNU parallel [62], Tensorflow [63], PyTorch [10] and torchvision [25].

Model	Parameter Count	BN Parameters	Fraction (%)
densenet121	7.98×10^6	8.36×10^4	0.010
densenet161	2.87×10^7	2.20×10^5	0.008
densenet169	1.41×10^7	1.58×10^5	0.011
densenet201	2.00×10^7	2.29×10^5	0.011
googlenet	1.30×10^7	1.51×10^4	0.001
inception-v3	2.72×10^7	3.62×10^4	0.001
mnasnet0-5	2.22×10^6	2.06×10^4	0.009
mnasnet0-75	3.17×10^6	2.98×10^4	0.009
mnasnet1-0	4.38×10^6	3.79×10^4	0.009
mnasnet1-3	6.28×10^6	4.88×10^4	0.008
mobilenet-v2	3.50×10^6	3.41×10^4	0.010
resnet101	4.45×10^7	1.05×10^5	0.002
resnet152	6.02×10^7	1.51×10^5	0.003
resnet18	1.17×10^7	9.60×10^3	0.001
resnet34	2.18×10^7	1.70×10^4	0.001
resnet50	2.56×10^7	5.31×10^4	0.002
resnext101-32x8d	8.88×10^7	2.03×10^5	0.002
shufflenet-v2-x0-5	1.37×10^6	7.95×10^3	0.006
shufflenet-v2-x1-0	2.28×10^6	1.62×10^4	0.007
shufflenet-v2-x1-5	3.50×10^6	2.34×10^4	0.007
shufflenet-v2-x2-0	7.39×10^6	3.37×10^4	0.005
vgg11-bn	1.33×10^8	5.50×10^3	4.142×10^{-5}
vgg13-bn	1.33×10^8	5.89×10^3	4.425×10^{-5}
vgg16-bn	1.38×10^8	8.45×10^3	6.106×10^{-5}
vgg19-bn	1.44×10^8	1.10×10^4	7.662×10^{-5}
wide-resnet101-2	1.27×10^8	1.38×10^5	0.001
wide-resnet50-2	6.89×10^7	6.82×10^4	0.001

Table 7: Overview of different models with parameter counts. We show the total number of BN parameters, which is a sum of affine parameters.

Category	Corruption	top1 error
Noise	Gaussian Noise	0.886428
	Shot Noise	0.894468
	Impulse Noise	0.922640
Blur	Defocus Blur	0.819880
	Glass Blur	0.826268
	Motion Blur	0.785948
	Zoom Blur	0.798360
Weather	Snow	0.866816
	Frost	0.826572
	Fog	0.819324
	Brightness	0.564592
	Contrast	0.853204
Digital	Elastic Transform	0.646056
	Pixelate	0.717840
	JPEG Compression	0.606500
Hold-out Noise	Speckle Noise	0.845388
Hold-out Digital	Saturate	0.658248
Hold-out Blur	Gaussian Blur	0.787108
Hold-out Weather	Spatter	0.717512

Table 8: AlexNet top1 errors on ImageNet-C

Table 9: After converting the checkpoints from TensorFlow to Pytorch, we notice a slight degradation in performance on the IN val set.

IN val top-1 accuracy in %.

Model	TF	PyTorch
SimCLRv2 ResNet50	76.3	75.6
SimCLRv2 ResNet101	78.2	77.5
SimCLRv2 ResNet152	79.3	78.6

Table 10: Adaptation improves the performance of the ResNet50 and the ResNet101 model but hurts the performance of the ResNet152 model.

ImageNet-C (n=4096), mCE.

Model, adaptation:	base	adapt	Δ
SimCLRv2 ResNet50	72.4	68.0	-4.2
SimCLRv2 ResNet101	66.6	65.1	-0.9
SimCLRv2 ResNet152	63.7	64.2	+0.5

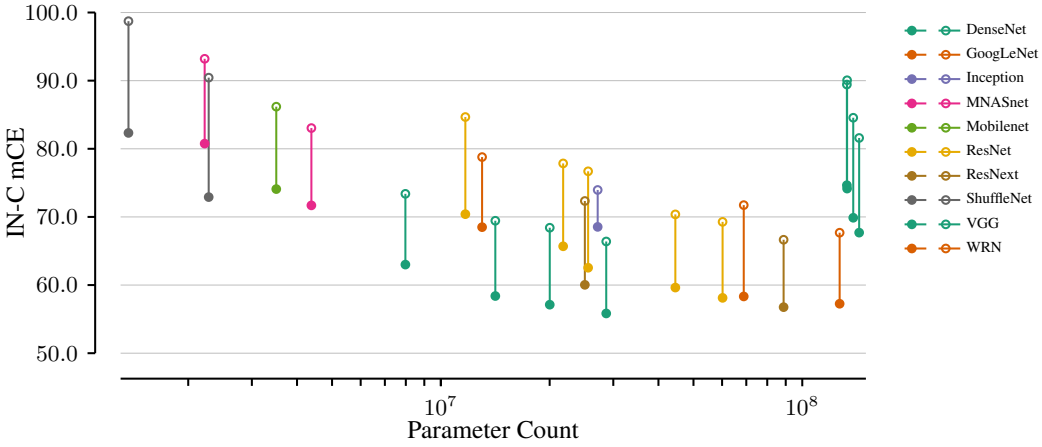


Figure 8: Adaptation (●) improves baseline (○) mCE across all 25 model architectures in the `torchvision` library, often on the order of 10% points. Best viewed in color.

C Additional results

C.1 Performance of SimCLRv2 models

We evaluate the performance of 3 models from the SimCLRv2 framework with and without batchnorm adaptation. We test a ResNet50, a ResNet101 and a ResNet152, finetuned on 100% of IN training data. Since our code-base is in PyTorch, we use the Pytorch-SimCLR-Converter [64] to convert the provided checkpoints from Tensorflow to PyTorch. We notice a slight decline in performance when comparing the top-1 accuracy on the IN validation set, see Table 9. For preprocessing, we disable the usual PyTorch normalization and use the PIL.Image.BICUBIC interpolation for resizing because this interpolation is used in the TensorFlow code (instead of the default PIL.Image.BILINEAR in PyTorch).

The BN adaptation results for the converted models are shown in Table 10. Adaptation improves the performance of the ResNet50 and the ResNet101 model, but hurts the performance of the ResNet152 model.

C.2 Relationship between parameter count and IN-C improvements

In addition to Fig. 3 in the main paper, we show the relationship between parameter count and IN-C mCE. In general, we see that the parameter counts correlates with corruption robustness since larger models have smaller mCE values.

C.3 Per-corruption results on IN-C

We provide more detailed results on the individual corruptions of IN-C for the most important models considered in our study in Fig. 9. The results are shown for models where the BN parameters are

adapted on the full test sets. The adaptation consistently improves the error rates on all corruptions for both vanilla and AugMix.

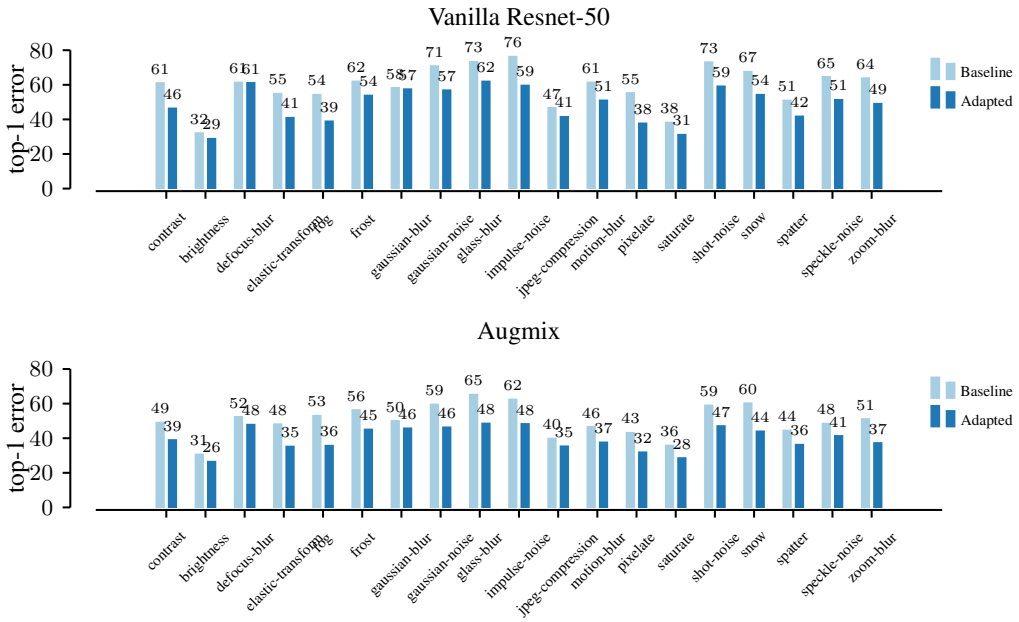


Figure 9: Results on the individual corruptions of IN-C for the vanilla trained ResNet-50 and the AugMix model with and without adaptation. Adaptation reduces the error on all corruptions.

C.4 Qualitative analysis of similarities between common corruptions

In this analysis, we compute a t-SNE embedding of the Wasserstein distances between the adapted models and the non-adapted model from Section 5, Fig. 4(i) of the main paper. The results are displayed in Fig. 10. We observe that the different corruption categories indicated by the different colors are grouped together except for the 'digital' category (pink). This visualization shows that corruption categories mostly induce similar shifts in the BN parameters. This might be an explanation why training a model on Gaussian noise generalizes so well to other noise types as has been observed by Rusak et al. [29]: By training on Gaussian noise, the BN statistics are adapted to the Gaussian noise corruption and from Fig. 10, we observe that these statistics are similar to the BN statistics of other noises.

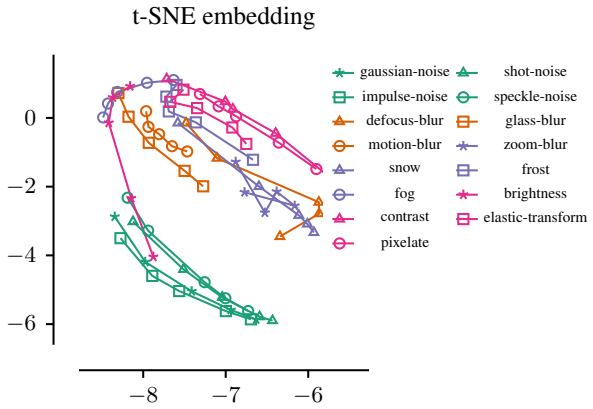


Figure 10: t-SNE embeddings of the Wasserstein distances between BN statistics adapted on the different corruptions. This plot shows evidence on the similarities between different corruption types.

C.5 Error prediction based on the Wasserstein distance

In Section 5, Fig. 4(i), we observe that the relationship between the Wasserstein distance and the top-1 error on IN-C is strikingly linear in the considered range of the Wasserstein distance. Similar corruptions and corruption types (indicated by color) exhibit similar slope, allowing to approximate the expected top-1 error rate without any information about the test domain itself. Using the split of the 19 corruptions into 15 test and 4 holdout corruptions [2], we compute a linear regression model on the five data points we get for each of the holdout corruptions (corresponding to the five severity levels), and use this model to predict the expected top-1 error rates for the remaining corruptions within the corruption family. This scheme works particularly for the "well defined" corruption types such as noise and digital (4.1% points absolute mean deviation from the real error. The full results are depicted in Table 11.

	test error			holdout (train) error			model	
	true	pred	$\lvert\Delta\rvert$	true	pred	$\lvert\Delta\rvert$	coef	intercept
Fig. 3 (i)								
blur	64.89	54.53	11.04	58.13	58.13	3.24	37.59	-0.70
digital	54.37	51.96	6.97	38.08	38.08	0.60	37.20	6.39
noise	73.29	69.68	5.84	64.51	64.51	0.65	24.66	1.68
weather	53.87	42.92	11.21	50.84	50.84	5.48	25.80	6.33
Fig. 3 (ii)								
blur	55.68	53.28	5.65	57.38	57.38	4.01	42.74	-9.51
digital	41.53	39.80	4.14	31.05	31.05	0.34	23.44	11.09
noise	58.43	55.04	4.14	51.24	51.24	1.01	18.13	5.06
weather	43.84	36.16	7.80	41.63	41.63	4.32	17.80	10.91
Fig. 3 (iii)								
blur	57.10	69.84	13.43	74.01	74.01	3.96	43.50	5.93
digital	46.16	38.06	12.97	36.22	36.22	10.52	4.94	32.01
noise	93.60	85.84	13.08	81.10	81.10	3.52	22.56	23.65
weather	43.74	36.90	8.98	44.05	44.05	6.20	23.29	3.87

Table 11: Estimating top-1 error of unseen corruptions within the different corruption classes. We note that especially for well defined corruptions (like noise or digital corruptions), the estimation scheme works well. We follow the categorization originally proposed by Hendrycks and Dietterich [2].

C.6 Training details on the models trained with Fixup initialization and GroupNorm

In Section 5 of the main paper, we consider IN models trained with GroupNorm and Fixup initialization. For these models, we consider the original reference implementations provided by the authors. We train ResNet-50, ResNet-101 and ResNet-152 models with stochastic gradient descent with momentum (learning rate 0.1, momentum 0.9), with batch size 256 and weight decay 1×10^{-4} for 100 epochs.

C.7 Effect of Pseudo Batchsize

We show the full results for considering different choices of N for ResNet-50, Augmix, ANT, ANT+SIN and SIN models and display the result in Fig. 12. We observe a characteristic shape which we believe can be attributed to the way statistics are estimated. We provide evidence for this view by proposing an analytical model which we discuss in §D.

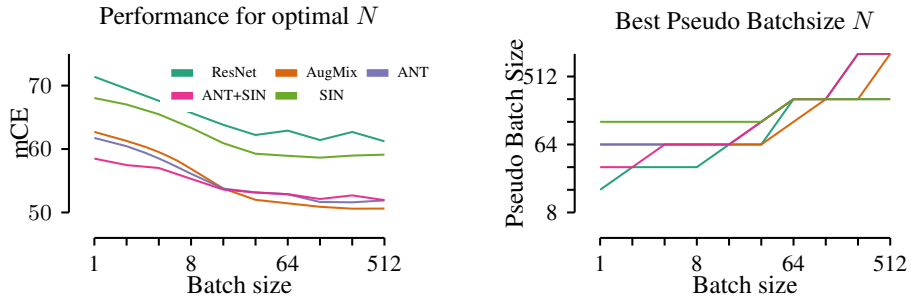


Figure 11: Left: Performance for all the considered ResNet-50 variants based on the sample batch size. The optimal N is chosen according to the mCE on the holdout corruptions. Right: Best choice for N depending on the input batchsize n. Note that in general for high values n, the model is generally more robust to the choice of N.

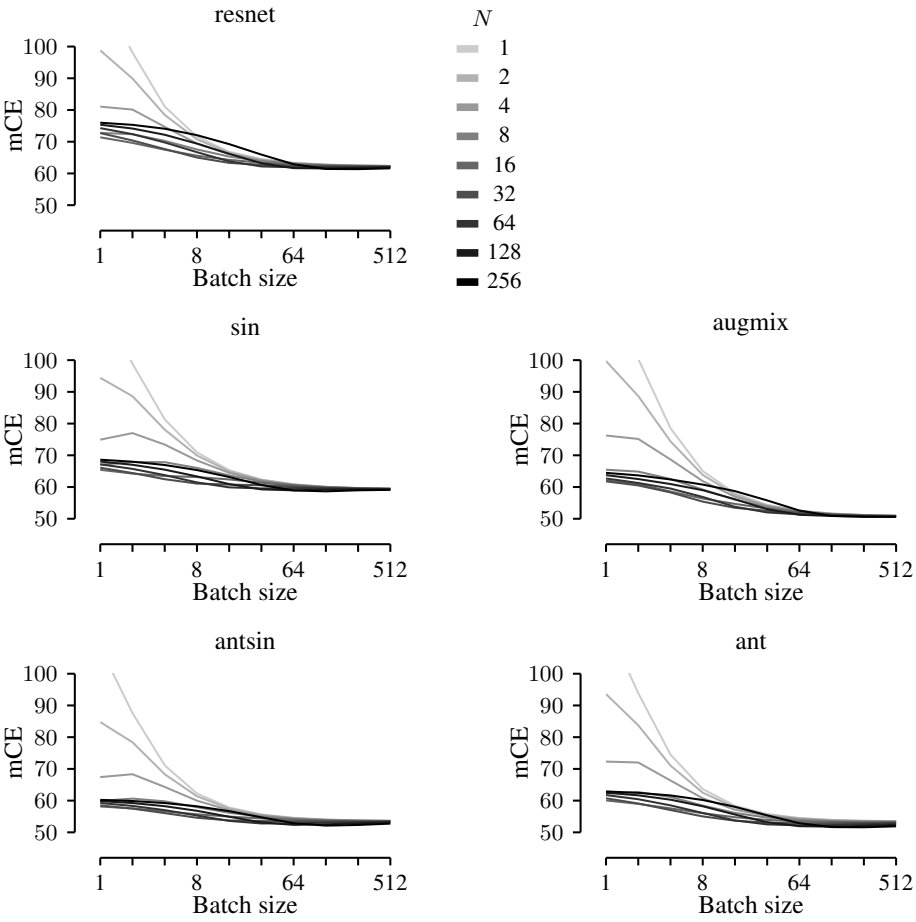


Figure 12: Effects of batch size n and pseudo batch size N for the various considered models. We report mCE averaged across 15 test corruptions.

ResNet-50	1	2	4	8	16	32	64	128	256
1	117.76	98.78	81.06	72.80	71.39	72.72	74.28	75.36	75.99
2	98.11	89.92	80.13	72.36	69.63	70.39	72.39	74.16	75.32
4	81.10	78.45	74.70	70.27	67.48	67.69	69.77	72.19	74.10
8	71.56	70.74	69.44	67.56	65.60	65.02	66.70	69.41	72.07
16	66.82	66.52	66.06	65.32	64.29	63.32	63.81	66.19	69.24
32	64.51	64.39	64.19	63.87	63.38	62.72	62.21	63.22	65.94
64	63.33	63.28	63.19	63.05	62.81	62.43	61.95	61.68	62.90
128	62.78	62.75	62.69	62.62	62.50	62.29	62.00	61.56	61.42
256	62.51	62.49	62.44	62.41	62.32	62.22	62.01	61.73	61.35
512	62.36	62.36	62.33	62.29	62.26	62.17	62.06	61.90	61.62

AugMix	1	2	4	8	16	32	64	128	256
1	122.56	99.72	76.23	65.46	62.08	61.78	62.70	63.75	64.47
2	100.39	88.69	75.16	64.86	60.93	60.51	61.28	62.52	63.67
4	78.55	74.41	68.69	62.52	58.58	58.30	59.53	60.94	62.39
8	65.02	63.81	61.86	59.21	56.39	55.40	56.87	59.00	60.77
16	58.02	57.55	56.96	56.02	54.69	53.44	53.78	56.15	58.71
32	54.37	54.20	53.99	53.68	53.21	52.50	51.99	53.01	55.78
64	52.55	52.50	52.38	52.24	52.07	51.83	51.39	51.25	52.59
128	51.64	51.60	51.54	51.47	51.38	51.26	51.10	50.88	50.89
256	51.18	51.17	51.12	51.08	51.02	50.95	50.86	50.76	50.60
512	50.96	50.95	50.93	50.90	50.86	50.80	50.72	50.65	50.61

ANT	1	2	4	8	16	32	64	128	256
1	116.10	93.58	72.31	62.28	60.07	60.73	61.75	62.48	62.90
2	93.88	83.74	72.01	62.69	58.97	59.10	60.44	61.67	62.44
4	74.51	71.06	66.34	61.15	57.55	57.03	58.51	60.29	61.64
8	63.65	62.50	60.74	58.43	56.04	55.02	56.10	58.22	60.20
16	58.37	57.87	57.14	56.11	54.77	53.67	53.76	55.61	58.06
32	55.78	55.54	55.20	54.66	53.91	53.06	52.50	53.18	55.35
64	54.51	54.41	54.21	53.88	53.42	52.84	52.23	51.94	52.87
128	53.92	53.85	53.71	53.53	53.28	52.85	52.29	51.80	51.65
256	53.66	53.61	53.50	53.37	53.20	52.96	52.54	52.04	51.60
512	53.53	53.49	53.41	53.33	53.21	53.02	52.78	52.38	51.90

ANT+SIN	1	2	4	8	16	32	64	128	256
1	108.24	84.75	67.42	59.91	58.15	58.49	59.24	59.85	60.23
2	87.60	78.40	68.32	60.63	57.54	57.47	58.33	59.23	59.87
4	71.12	68.32	64.31	59.78	56.63	56.06	57.01	58.24	59.23
8	62.23	61.38	59.98	57.93	55.69	54.59	55.30	56.79	58.21
16	57.83	57.51	57.00	56.17	54.96	53.76	53.61	54.92	56.68
32	55.62	55.51	55.33	54.96	54.38	53.55	52.80	53.13	54.73
64	54.57	54.49	54.40	54.25	53.98	53.51	52.84	52.36	52.89
128	54.02	53.98	53.95	53.85	53.72	53.49	53.07	52.53	52.12
256	53.76	53.74	53.71	53.67	53.59	53.47	53.23	52.85	52.33
512	53.64	53.63	53.60	53.57	53.51	53.45	53.35	53.12	52.75

SIN	1	2	4	8	16	32	64	128	256
1	119.11	94.43	74.93	67.03	65.43	66.08	67.16	68.04	68.62
2	98.85	88.62	76.99	67.88	64.23	64.42	65.72	67.02	67.99
4	81.35	78.10	73.38	67.84	63.49	62.47	63.76	65.48	66.94
8	70.92	69.94	68.38	66.02	63.14	61.09	61.45	63.35	65.35
16	65.29	64.97	64.48	63.68	62.39	60.78	59.90	60.92	63.16
32	62.34	62.25	62.08	61.80	61.36	60.55	59.55	59.26	60.65
64	60.84	60.80	60.74	60.61	60.47	60.15	59.67	58.96	58.93
128	60.07	60.04	60.02	59.96	59.87	59.77	59.57	59.18	58.64
256	59.68	59.66	59.64	59.62	59.59	59.53	59.43	59.27	58.97
512	59.48	59.47	59.46	59.44	59.42	59.40	59.33	59.26	59.11

DeepAugment	1	2	4	8	16	32	64	128	256
8	65.37	63.87	61.37	58.11	54.48	52.17	52.33	54.18	56.36

DeepAugment+AugMix	1	2	4	8	16	32	64	128	256
8	52.59	51.98	51.05	49.83	48.5	47.81	48.36	49.72	51.12

ResNext+DeepAugment+Augmix	1	2	4	8	16	32	64	128	256
8	42.09	41.74	41.29	40.67	39.96	39.69	40.35	41.55	42.69

Table 12: Test mCE for various batch sizes (rows) vs. pseudo batch sizes (columns)

D Analytical error model

We consider a univariate model in §D.1–D.3 and discuss a simple extension to the multivariate diagonal case in §D.4. As highlighted in the main text, the model qualitatively explains the overall characteristics of our experimental data. Note that we assume a linear relationship between the Wasserstein distance and the error under domain shift, as suggested by our empirical findings.

Univariate model. We denote the source statistics as μ_s, σ_s^2, the true target statistics as μ_t, σ_t^2 and the estimated target statistics as $\hat{\mu}_t, \hat{\sigma}_t^2$. For normalization, we take a convex combination of the source statistics and estimated target statistics:

$$\bar{\mu} = \frac{N}{N+n}\mu_s + \frac{n}{N+n}\hat{\mu}_t,\ \bar{\sigma}^2 = \frac{N}{N+n}\sigma_s^2 + \frac{n}{N+n}\hat{\sigma}_t^2. \tag{11}$$

We now analyze the trade-off between using an estimate closer to the source or closer to the estimated target statistics. In the former case, the model will suffer under the covariate shift present between target and source distribution. In the latter case, small batch sizes n will yield unreliable estimates for the true target statistics, which might hurt the performance even more than the source-target mismatch. Hence, we aim to gain understanding in the trade-off between both options, and potential optimal choices of N for a given sample size n.

As a metric of domain shift with good properties for our following derivation, we leverage the Wasserstein distance. In §5 and §C.5, we already established an empirical link between domain shift measured in terms of the top-1 performance vs. the Wasserstein distance between model statistics and observed a linear relationship for case of common corruptions.

Proposition 1 (Bounds on the expected value of the Wasserstein distance between target and combined estimated target and source statistics)**.** *We denote the source statistics as μ_s, σ_s^2, the true target statistics as μ_t, σ_t^2 and the biased estimates of the target statistics as $\hat{\mu}_t, \hat{\sigma}_t^2$. For normalization, we take a convex combination of the source statistics and estimated target statistics as discussed in Eq. 11. At a confidence level $1-\alpha$, the expectation value of the squared Wasserstein distance $W_2^2(\bar{\mu}, \bar{\sigma}, \mu_t, \sigma_t)$ between ideal and estimated target statistics w.r.t. to the distribution of sample mean $\hat{\mu}_t$ and sample variance $\hat{\sigma}_t^2$ is bounded from above and below with $L \leq \mathbb{E}[W_2^2] \leq U$, where*

$$\begin{aligned} L &= \left(\sigma_t - \sqrt{\frac{N}{N+n}\sigma_s^2 + \frac{n-1}{N+n}\sigma_t^2}\right)^2 + \frac{N^2}{(N+n)^2}\left(\mu_t - \mu_s\right)^2 + \frac{n}{(N+n)^2}\sigma_t^2 \\ U &= L + \sigma_t^5 \frac{(n-1)}{2(N+n)^2}\left(\frac{N}{N+n}\sigma_s^2 + \frac{1}{N+n}\chi^2_{1-\alpha/2,n-1}\sigma_t^2\right)^{-3/2} \end{aligned} \tag{12}$$

The quantity $\chi^2_{1-\alpha/2,n-1}$ denotes the left tail value of a chi square distribution with $n-1$ degrees of freedom, defined as $P\left(X \leq \chi^2_{1-\alpha/2,n-1}\right) = \alpha/2$ for $X \sim \chi^2_{n-1}$.

D.1 Proof sketch

We are interested in the expected value of the Wasserstein distance defined in (A.1) between the target statistics μ_t, σ_t^2 and the mixed statistics $\bar{\mu}, \bar{\sigma}^2$ introduced above in equation (11), taken with respect to the distribution of the sample moments $\hat{\mu}_t$, $\hat{\sigma}_t^2$. The expectation value itself cannot be evaluated in closed form because the Wasserstein distance contains a term proportional to $\bar{\sigma}$ being the square root of the convex combination of target and source variance.

In Lemma 3, the square root term is bounded from above and below using Jensen's inequality and Holder's defect formula which is reviewed in Lemma 2. After having bounded the problematic square root term, the proof of Proposition 1 reduces to inserting the expectation values of sample mean and sample variance reviewed in Lemma 1.

D.2 Prerequisites

Lemma 1 (Mean and variance of sample moments, following [65])**.** *The sample moments $\hat{\mu}_t, \hat{\sigma}_t^2$ are random variables depending on the sample size n.*

$$\hat{\mu}_t = \frac{1}{n}\sum_{j=1}^{n} x_j, \quad \hat{\sigma}_t^2 = \frac{1}{n}\sum_{j=1}^{n}(x_j - \hat{\mu}_t)^2 \;\; \textit{with } x_j \sim \mathcal{N}\left(\mu_t, \sigma_t^2\right). \tag{13}$$

For brevity, we use the shorthand $\mathbb{E}[\cdot]$ for all expectation values with respect to the distribution of $p(\hat{\mu}_t, \hat{\sigma}_t^2|n)$. In particular, our computation uses mean and variance of $\hat{\mu}_t$ and $\hat{\sigma}_t^2$ which are well known for a normal target distribution:

$$\hat{\mu}_t \sim \mathcal{N}\left(\mu_t, \frac{1}{n}\sigma_t^2\right), \; \mathbb{E}[\hat{\mu}_t] = \mu_t, \; \mathbb{V}[\hat{\mu}_t] = \frac{1}{n}\sigma_t^2 \tag{14}$$

$$\frac{\hat{\sigma}_t^2}{\sigma_t^2/n} \sim \chi^2_{n-1}, \; \mathbb{E}[\hat{\sigma}_t^2] = \frac{n-1}{n}\sigma_t^2, \; \mathbb{V}[\hat{\sigma}_t^2] = \frac{\sigma_t^4}{n^2}\mathbb{V}\left[\frac{\hat{\sigma}_t^2}{\sigma_t^2/n}\right] = \frac{\sigma_t^4}{n^2}\,2(n-1). \tag{15}$$

The derivation of the variance $\mathbb{V}[\hat{\sigma}_t^2]$ in the last line uses the fact that the variance of a chi square distributed variable with $(n-1)$ degrees of freedom is equal to $2(n-1)$.

Lemma 2 (Holder's defect formula for concave functions in probabilistic notation, following Becker [66])**.** *If the concave function $f : [a, b] \to \mathbb{R}$ is twice continuously differentiable and there are finite bounds m and M such that*

$$-M \leq f''(x) \leq -m \leq 0 \; \forall x \in [a, b], \tag{16}$$

then the defect between Jensen's inequality estimate $f\left(\mathbb{E}[X]\right)$ for a random variable X taking values $x \in [a, b]$ and the true expectation value $\mathbb{E}[f(X)]$ is bounded from above by a term proportional to the variance of X:

$$f\left(\mathbb{E}[X]\right) - \mathbb{E}[f(X)] \leq \frac{1}{2}M\mathbb{V}[X]. \tag{17}$$

Lemma 3 (Upper and lower bounds on the expectation value of $\bar{\sigma}$)**.** *The expectation value of the square root of the random variable $\bar{\sigma}^2$ defined as*

$$\bar{\sigma}^2 = \frac{N}{N+n}\sigma_s^2 + \frac{n}{N+n}\hat{\sigma}_t^2, \tag{18}$$

is bounded from above and below at a confidence level $1-\alpha$ by

$$\sqrt{\mathbb{E}\left[\bar{\sigma}^2\right]} - \frac{1}{2}M\mathbb{V}[\bar{\sigma}^2] \leq \mathbb{E}\left[\sqrt{\bar{\sigma}^2}\right] \leq \sqrt{\mathbb{E}\left[\bar{\sigma}^2\right]} \tag{19}$$

$$\sqrt{\mathbb{E}\left[\bar{\sigma}^2\right]} = \sqrt{\frac{N}{N+n}\sigma_s^2 + \frac{n-1}{N+n}\sigma_t^2}, \tag{20}$$

$$\frac{1}{2}M\mathbb{V}[\bar{\sigma}^2] = \frac{(n-1)}{4(N+n)^2}\sigma_t^4\left(\frac{N}{N+n}\sigma_s^2 + \frac{1}{N+n}\chi^2_{1-\alpha/2,n-1}\sigma_t^2\right). \tag{21}$$

The quantity $\chi^2_{1-\alpha/2,n-1}$ denotes the left tail value of a chi square distribution with $n-1$ degrees of freedom, defined as $P\left(X \leq \chi^2_{1-\alpha/2,n-1}\right) = \alpha/2$ for $X \sim \chi^2_{n-1}$.

Proof. The square root function is concave, therefore Jensen's inequality implies the upper bound

$$\mathbb{E}\left[\sqrt{\bar{\sigma}^2}\right] \leq \sqrt{\mathbb{E}[\bar{\sigma}^2]}. \tag{22}$$

The square root of the expectation value of $\bar{\sigma}^2$ is computed using the expectation value of the sample variance as given in Lemma 1.

$$\sqrt{\mathbb{E}[\bar{\sigma}^2]} = \sqrt{\frac{N}{N+n}\sigma_s^2 + \frac{n}{N+n}\frac{n-1}{n}\sigma_t^2} = \sqrt{\frac{N}{N+n}\sigma_s^2 + \frac{n-1}{N+n}\sigma_t^2}. \tag{23}$$

To state a lower bound, we use Holder's defect formula in probabilistic notation stated in Lemma 2. Holder's formula for concave functions requires that the random variable $\bar{\sigma}^2$ can take values in

the compact interval $[a, b]$ and that the second derivative of the square root function $f(\bar{\sigma}^2) = \sqrt{\bar{\sigma}^2}$, exists and is strictly smaller than zero in $[a, b]$. Regarding the interval of $\bar{\sigma}^2$, we provide probabilistic upper and lower bounds. The ratio of sample variance and true variance divided by n follows a chi square distribution with $n-1$ degrees of freedom. At confidence level $1-\alpha$, this ratio lies between $\chi^2_{1-\alpha/2,n-1}$ and $\chi^2_{\alpha/2,n-1}$ which are defined as follows:

$$\chi^2_{1-\alpha/2,n-1} \leq \frac{\hat{\sigma}_t^2}{\sigma_t^2/n} \leq \chi^2_{\alpha/2,n-1}, \tag{24}$$

$$Pr(X \leq \chi^2_{1-\alpha/2,n-1}) = \frac{\alpha}{2},\ Pr(X \geq \chi^2_{\alpha/2,n-1}) = \frac{\alpha}{2}. \tag{25}$$

Then at the same confidence level, the sample variance itself lies between the two quantiles multiplied by σ_t^2/n,

$$\chi^2_{1-\alpha/2,n-1}\frac{\sigma_t^2}{n} \leq \hat{\sigma}_t^2 \leq \chi^2_{\alpha/2,n-1}\frac{\sigma_t^2}{n}, \tag{26}$$

and the random variable $\bar{\sigma}^2$ lies in the interval

$$\bar{\sigma}^2 \in [a, b] \text{ with } a = \frac{N}{N+n}\sigma_s^2 + \frac{1}{N+n}\chi^2_{1-\alpha/2,n-1}\sigma_t^2, \tag{27}$$

$$\text{and } b = \frac{N}{N+n}\sigma_s^2 + \frac{1}{N+n}\chi^2_{\alpha/2,n-1}\sigma_t^2. \tag{28}$$

The variances and chi square values are all positive and therefore both a and b are positive as well, implying that the second derivative of the square root is strictly negative in the interval $[a, b]$.

$$f(\bar{\sigma}^2) = \sqrt{\bar{\sigma}^2},\ f'(\bar{\sigma}^2) = \frac{1}{2}(\bar{\sigma}^2)^{-1/2},\ f''(\bar{\sigma}^2) = -\frac{1}{4}(\bar{\sigma}^2)^{-3/2} < 0 \in [a, b]. \tag{29}$$

Consequently the second derivative is in the interval $[M, m]$ at the given confidence level:

$$-M \leq f''(\bar{\sigma}^2) \leq -m \leq 0 \text{ for } \bar{\sigma}^2 \in [a, b] \text{ with } M = \frac{1}{4}a^{-3/2},\ m = \frac{1}{4}b^{-3/2}. \tag{30}$$

The defect formula 2 states that the defect is bounded by

$$\sqrt{\mathbb{E}[\bar{\sigma}^2]} - \mathbb{E}[\sqrt{\bar{\sigma}^2}] \leq \frac{1}{2}M\mathbb{V}[\bar{\sigma}^2]. \tag{31}$$

The constant M was computed above in (30), and the variance of $\bar{\sigma}^2$ is calculated in the next lines, using the first and second moment of the sample variance as stated in 1.

$$\begin{aligned}
\mathbb{V}[\bar{\sigma}^2] &= \mathbb{E}[(\bar{\sigma}^2 - \mathbb{E}[\bar{\sigma}^2])^2] = \mathbb{E}\left[\left(\frac{n}{N+n}\hat{\sigma}_t^2 - \frac{n}{N+n}\frac{n-1}{n}\sigma_t^2\right)^2\right] \\
&= \frac{n^2}{(N+n)^2}\mathbb{E}\left[\left(\hat{\sigma}_t^2 - \mathbb{E}[\hat{\sigma}_t^2\right)^2\right] = \frac{n^2}{(N+n)^2}\mathbb{V}\left[\hat{\sigma}_t^2\right] \\
&= \frac{n^2}{(N+n)^2}\frac{2(n-1)}{n^2}\sigma_t^4 = \frac{2(n-1)}{(N+n)^2}\sigma_t^4.
\end{aligned} \tag{32}$$

Inserting $\mathbb{V}[\bar{\sigma}^2]$ computed in (32) and M defined in (30) with a as defined in (27) into the defect formula (31) yields the lower bound:

$$\begin{aligned}
&\sqrt{\mathbb{E}[\bar{\sigma}^2]} - \frac{1}{2}M\mathbb{V}[\bar{\sigma}^2] \leq \mathbb{E}[\sqrt{\bar{\sigma}^2}] \\
&\sqrt{\mathbb{E}[\bar{\sigma}^2]} - \frac{1}{2}M\mathbb{V}[\bar{\sigma}^2] \\
&= \sqrt{\mathbb{E}[\bar{\sigma}^2]} - \frac{1}{2}\cdot\frac{1}{4}a^{-3/2}\frac{2(n-1)}{(N+n)^2}\sigma_t^4 \\
&= \sqrt{\mathbb{E}[\bar{\sigma}^2]} - \frac{(n-1)}{4(N+n)^2}\sigma_t^4\left(\frac{N}{N+n}\sigma_s^2 + \frac{1}{N+n}\chi^2_{1-\alpha/2,n-1}\sigma_t^2\right)^{-3/2}.
\end{aligned} \tag{33}$$

Assuming that source and target variance are of the same order of magnitude σ, the defect will be of order of magnitude σ: The factor $\mathbb{V}[X]$ scales with σ^4 and M with σ^{-3}. □

D.3 Proof of Proposition 1

Proof. For two univariate normal distributions with moments μ_t, σ_t^2 and $\bar{\mu}, \bar{\sigma}^2$, the Wasserstein distance as defined in (A.1) reduces to

$$W_2^2 = \sigma_t^2 + \bar{\sigma}^2 - 2\bar{\sigma}\sigma_t + (\bar{\mu} - \mu)^2. \tag{34}$$

The expected value of the Wasserstein distance across many batches is given as

$$\begin{aligned}\mathbb{E}[W_2^2] &= \sigma_t^2 + \mathbb{E}[\bar{\sigma}^2] - 2\mathbb{E}[\bar{\sigma}]\sigma_t + \mathbb{E}[(\mu_t - \bar{\mu})^2] \\ &= \sigma_t^2 + \frac{N}{N+n}\sigma_s^2 + \frac{n}{N+n}\frac{n-1}{n}\sigma_t^2 - 2\sigma_t\mathbb{E}\left[\sqrt{\frac{N}{N+n}\sigma_s^2 + \frac{n}{N+n}\hat{\sigma}_t^2}\right] \\ &\quad + \mathbb{E}\left[\left(\mu_t - \frac{N}{N+n}\mu_s - \frac{n}{N+n}\hat{\mu}_t\right)^2\right]\end{aligned} \tag{35}$$

which can already serve as the basis for our numerical simulations. To arrive at a closed form analytical solution, we invoke Lemma 3 to bound the expectation value $\mathbb{E}\left[\bar{\sigma}\right]$ in equation (35).

$$-2\sigma_t\sqrt{\mathbb{E}\left[\bar{\sigma}^2\right]} \leq -2\sigma_t\mathbb{E}\left[\sqrt{\bar{\sigma}^2}\right] \leq -2\sigma_t\sqrt{\mathbb{E}\left[\bar{\sigma}^2\right]} - 2\sigma_t\left(-\frac{1}{2}M\mathbb{V}[\bar{\sigma}^2]\right) \tag{36}$$

Apart from the square root term bounded in equation (36) above, the expectation value of the Wasserstein distance can be computed exactly. Hence the bounds on $\mathbb{E}\left[\bar{\sigma}\right]$ multiplied by a factor of $(-2\sigma_t^2)$ coming from equation (35) determine lower and upper bounds L and U on the expected value of W_2^2:

$$L \leq \mathbb{E}\left[W_2^2\right] \leq U = L + \sigma_t M\mathbb{V}[\bar{\sigma}^2] \tag{37}$$

In the next lines, the lower bound is calculated:

$$\begin{aligned}L = \; & \sigma_t^2 + \frac{N}{N+n}\sigma_s^2 + \frac{n-1}{N+n}\sigma_t^2 - 2\sigma_t\sqrt{\mathbb{E}\left[\frac{N}{N+n}\sigma_s^2 + \frac{n-1}{N+n}\sigma_t^2\right]} \\ & + \left(\mu_t - \frac{N}{N+n}\mu_s\right)^2 - 2\left(\mu_t - \frac{N}{N+n}\mu_s\right)\frac{n}{N+n}\mathbb{E}[\hat{\mu}_t] + \frac{n^2}{(N+n)^2}\left(\mathbb{V}[\hat{\mu}_t] + (\mathbb{E}[\hat{\mu}_t])^2\right) \\ = \; & \sigma_t^2 + \frac{N}{N+n}\sigma_s^2 + \frac{n-1}{N+n}\sigma_t^2 - 2\sigma_t\sqrt{\frac{N}{N+n}\sigma_s^2 + \frac{n-1}{N+n}\sigma_t^2} \\ & + \left(\mu_t - \frac{N}{N+n}\mu_s\right)^2 - 2\left(\mu_t - \frac{N}{N+n}\mu_s\right)\frac{n}{N+n}\mu_t + \frac{n^2}{(N+n)^2}\left(\frac{1}{n}\sigma_t^2 + \mu_t^2\right) \\ = \; & \left(\sigma_t - \sqrt{\frac{N}{N+n}\sigma_s^2 + \frac{n-1}{N+n}\sigma_t^2}\right)^2 + \left(\mu_t - \frac{N}{N+n}\mu_s - \frac{n}{N+n}\mu_t\right)^2 + \frac{n}{(N+n)^2}\sigma_t^2 \\ = \; & \left(\sigma_t - \sqrt{\frac{N}{N+n}\sigma_s^2 + \frac{n-1}{N+n}\sigma_t^2}\right)^2 + \frac{N^2}{(N+n)^2}\left(\mu_t - \mu_s\right)^2 + \frac{n}{(N+n)^2}\sigma_t^2\end{aligned} \tag{38}$$

After having derived the lower bound, the upper bound is the sum of the lower bound and the defect term as computed in Lemma 3.

$$\begin{aligned}\mathbb{E}[W^2] \geq U &= L + \sigma_t M\mathbb{V}[\bar{\sigma}^2] \\ &= L + \sigma_t\frac{1}{4}\left(\frac{N}{N+n}\sigma_s^2 + \frac{n}{N+n}\chi^2_{1-\alpha/2,n-1}\frac{\sigma_t^2}{n}\right)^{-3/2}\frac{2(n-1)}{(N+n)^2}\sigma_t^4 \\ &= L + \left(\frac{N}{N+n}\sigma_s^2 + \frac{1}{N+n}\chi^2_{1-\alpha/2,n-1}\sigma_t^2\right)^{-3/2}\frac{(n-1)}{2(N+n)^2}\sigma_t^5.\end{aligned} \tag{39}$$

□

Based on choices of the model parameters, the model qualitatively matches our experimental results. We plot different choices in Fig. 13.

Figure 13: Overview of different parametrizations of the model. We denote each plot with $(\mu_t - \mu_s, \sigma_t/\sigma_s)$ and report the lower bound $\sqrt{L}$ on the Wasserstein distance. Parametrizations in columns four to seven produce qualitatively similar results we observed in our experiments, assuming a linear relationship between the Wasserstein distance and the error rate.

D.4 Extension to multivariate distributions.

We now derive a multivariate variant that can be fit to data from a DNN. Due to the estimation of running statistics in the network, we have access to a diagonal approximation of the true covariance matrix.

We denote the diagonal covariance matrices with matrix elements σ_i^2 as

$$(\Sigma_t)_{ii} = (\sigma_t^2)_i,\ (\hat{\Sigma}_t)_{ii} = (\hat{\sigma}_t^2)_i,\ (\Sigma_s)_{ii} = (\sigma_s^2)_i \tag{40}$$

and extend our definition of the statistics used for normalization to $\bar{\boldsymbol{\mu}}$ and $\bar{\boldsymbol{\Sigma}}$:

$$\bar{\boldsymbol{\mu}} = \frac{N}{N+n}\boldsymbol{\mu}_s + \frac{n}{N+n}\hat{\boldsymbol{\mu}}_t,\ \bar{\boldsymbol{\Sigma}} = \frac{N}{N+n}\boldsymbol{\Sigma}_s + \frac{n}{N+n}\hat{\boldsymbol{\Sigma}}_t. \tag{41}$$

The Wasserstein distance between $\bar{\boldsymbol{\mu}}, \bar{\boldsymbol{\Sigma}}$ and $\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t$ is then defined as

$$\begin{aligned} W_2^2 =&\ \operatorname{Tr} \boldsymbol{\Sigma}_t + \bar{\boldsymbol{\Sigma}} - 2\boldsymbol{\Sigma}_t^{1/2}\bar{\boldsymbol{\Sigma}}^{1/2} + (\boldsymbol{\mu}_t - \bar{\boldsymbol{\mu}})^T(\boldsymbol{\mu}_t - \bar{\boldsymbol{\mu}}) \\ =&\ \sum_{i=1}^{D}(\sigma_t^2)_i + (\bar{\sigma}^2)_i - 2(\bar{\sigma})_i(\sigma_t)_i + ((\mu_t)_i - (\bar{\mu}_t)_i)^2 = \sum_{i=1}^{D}(W_2^2)_i \end{aligned} \tag{42}$$

Every component $(W_2^2)_i$ in the sum above is bounded by the univariate bound discussed above. The multivariate Wasserstein distance which sums over the diagonal covariance matrix entries is then bounded by the sums over the individual bounds L_i and U_i given in (12).

$$L_i \leq (W_2^2)_i \leq U_i \Rightarrow \sum_{i=1}^{D} L_i \leq W_2^2 \leq \sum_{i=1}^{D} U_i. \tag{43}$$

D.5 Limits of Proposition 1

Limit $n \to \infty$ In the limit of infinite batch size $n \to \infty$, upper and lower bounds on the expected Wasserstein distance between $\bar{\mu}, \bar{\sigma}^2$ and μ_t, σ_t^2 both go to zero.

$$\begin{aligned} \lim_{n\to\infty} L =& \lim_{n\to\infty} \left(\sigma_t - \sqrt{\frac{N}{N+n}\sigma_s^2 + \frac{n-1}{N+n}\sigma_t^2}\right)^2 + \frac{N^2}{(N+n)^2}\left(\mu_t - \mu_s\right)^2 + \frac{n}{(N+n)^2}\sigma_t^2 \\ =&(\sigma_t - \sigma_t)^2 = 0 \\ \lim_{n\to\infty} U =& \lim_{n\to\infty} L + \lim_{n\to\infty} \sigma_t^5 \frac{(n-1)}{2(N+n)^2}\left(\frac{N}{N+n}\sigma_s^2 + \frac{1}{N+n}\chi^2_{1-\alpha/2,n-1}\sigma_t^2\right)^{-3/2} = 0. \end{aligned} \tag{44}$$

The intuition behind this limit is that if a large number of samples from the target domain is given, $\hat{\mu}$ and $\hat{\sigma}^2$ approximate the true target statistics very well. As $\hat{\mu}$ and $\hat{\sigma}^2$ dominate $\bar{\mu}$ and $\bar{\sigma}^2$ for large n, the expected Wasserstein distance has to vanish.

Limit $N \to \infty$ In the opposite limit $N \to \infty$, the expected value of the Wasserstein distance reduces to the Wasserstein distance between source and target statistics.

$$\lim_{N\to\infty} \bar{\mu} = \mu_s,\ \lim_{N\to\infty} \bar{\sigma}^2 = \sigma_s^2, \tag{45}$$

$$\Rightarrow \lim_{N\to\infty} \mathbb{E}[W_2^2] = \sigma_t^2 + \sigma_s^2 - 2\sigma_t\sigma_s + (\mu_t - \mu_s)^2 = W_2^2\left(\mu_s, \sigma_s^2, \mu_t, \sigma_t^2\right). \tag{46}$$

Limiting case $\mu_t = \mu_s$ **and** $\sigma_t^2 = \sigma_s^2$ When source and target domain coincide, and the statistics $\sigma_s^2 = \sigma_t^2$ and $\mu_s = \mu_t$ are known, then the source target mismatch is not an error source.

However, one might assume that source and target domain are different even though they actually coincide. In this case, proceeding with our proposed strategy and using the statistics $\bar{\mu}$ and $\bar{\sigma}^2$, the bounds on the expected Wasserstein distance follow from setting σ_t^2 to σ_s^2 and μ_t to μ_s in

Proposition 1.

$$\bar{\mu} = \frac{N}{N+n}\mu_t + \frac{n}{N+n}\hat{\mu}_t,\ \bar{\sigma}^2 = \frac{N}{N+n}\sigma_t^2 + \frac{n}{N+n}\hat{\sigma}_t^2,\ L \leq \mathbb{E}[W_2^2] \leq U$$
$$L = \sigma_t^2\left(\frac{2N^2+4Nn-N+2n^2}{(N+n)^2} - 2\sqrt{1-\frac{1}{N+n}}\right), \tag{47}$$
$$U = L + \sigma_t^2\frac{n-1}{2(N+n)^2}\left(\frac{N+\chi^2_{1-\alpha/2,n-1}}{N+n}\right)^{-3/2}.$$

It could also be the case that the equality of source and target statistics is known but the concrete values of the statistics are unknown. In our model, this amounts to setting the number of pseudo samples N to zero and assuming that source and target statistics are equal. Setting $N = 0$ in equation (47) and keeping n finite yields

$$L = 2\sigma_t^2\left(1-\sqrt{1-\frac{1}{n}}\right),\ U = L + \sigma_t^2\frac{n-1}{2n^2}\left(\frac{\chi^2_{1-\alpha/2,n-1}}{n}\right)^{-3/2}. \tag{48}$$

D.6 Bounds on the normalized Wasserstein distance

The Wasserstein distance (cf. §A.1) between the interpolating statistics $\bar{\mu}$, $\bar{\sigma}^2$ and the target statistics can also be normalized by a factor of σ_s^{-2}. Because σ_s^{-2} is constant, the bounds on the expectation value of the unnormalized Wasserstein distance discussed in the previous subsections just have to be multiplied by σ_s^{-2} to obtain bounds on the normalized Wasserstein distance (cf. §A.2):

$$\frac{L}{\sigma_s^2} \leq \widetilde{W}_2^2 = W_2^2\left(\frac{\bar{\mu}}{\sigma_s},,\frac{\bar{\sigma}^2}{\sigma_s^2},\frac{\mu_t}{\sigma_s},\frac{\sigma_t^2}{\sigma_s^2}\right) = \frac{1}{\sigma_s^2}W_2^2(\bar{\mu},\bar{\sigma}^2,\mu_t,\sigma_t^2) \leq \frac{U}{\sigma_s^2}. \tag{49}$$

E Full list of models evaluated on IN

The following lists contains all models we evaluated on various datasets with references and links to the corresponding source code.

E.1 Torchvision models trained on IN

Weights were taken from `https://github.com/pytorch/vision/tree/master/torchvision/models`

1. `alexnet` [67]
2. `densenet121` [15]
3. `densenet161` [15]
4. `densenet169` [15]
5. `densenet201` [15]
6. `densenet201` [15]
7. `googlenet` [16]
8. `inception_v3` [17]
9. `mnasnet0_5` [18]
10. `mnasnet1_0` [18]
11. `mobilenet_v2` [19]
12. `resnet18` [20]
13. `resnet34` [20]
14. `resnet50` [20]
15. `resnet101` [20]
16. `resnet152` [20]
17. `resnext50_32x4d` [21]
18. `resnext101_32x8d` [21]
19. `shufflenet_v2_x0_5` [22]
20. `shufflenet_v2_x1_0` [22]
21. `vgg11_bn` [23]
22. `vgg13_bn` [23]
23. `vgg16_bn` [23]
24. `vgg19_bn` [23]
25. `wide_resnet101_2` [24]
26. `wide_resnet50_2` [24]

E.2 Robust ResNet50 models

1. `resnet50 AugMix` [30] `https://github.com/google-research/augmix`
2. `resnet50 SIN+IN` [28] `https://github.com/rgeirhos/texture-vs-shape`
3. `resnet50 ANT` [29] `https://github.com/bethgelab/game-of-noise`
4. `resnet50 ANT+SIN` [29] `https://github.com/bethgelab/game-of-noise`
5. `resnet50 DeepAugment` [36] `https://github.com/hendrycks/imagenet-r`
6. `resnet50 DeepAugment+AugMix` [36] `https://github.com/hendrycks/imagenet-r`

E.3 SimCLRv2 models [27]

We used the checkpoints from `https://github.com/google-research/simclr` and converted them from TensorFlow to PyTorch with `https://github.com/tonylins/simclr-converter`, commit ID: 139d3cb0bd0c64b5ad32aab810e0bd0a0dddaae0.

1. `resnet50` FT100 SK=0 width=1
2. `resnet101` FT100 SK=0 width=1
3. `resnet152` FT100 SK=0 width=1

E.4 Robust ResNext models [21]

Note that the baseline `resnext50_32x4d` model trained on ImageNet is available as part of the `torchvision` library.

1. `resnext50_32x4d WSL` [26] `https://github.com/facebookresearch/WSL-Images/blob/master/hubconf.py`
2. `resnext101_32x4d WSL` [26] `https://github.com/facebookresearch/WSL-Images/blob/master/hubconf.py`
3. `resnext101_32x8d Deepaugment+AugMix` [36] `https://github.com/hendrycks/imagenet-r`

E.5 ResNet50 with Group Normalization [40]

Model weights and training code was taken from `https://github.com/ppwwyyxx/GroupNorm-reproduce`

1. `resnet50 GroupNorm`
2. `resnet101 GroupNorm`
3. `resnet152 GroupNorm`

E.6 ResNet50 with Fixup initialization [39]

Model weights and training code was taken from `https://github.com/hongyi-zhang/Fixup/tree/master/imagenet`. For training, we keep all hyperparameters at their default values and note that in particular the batchsize of 256 is a sensitive parameter.

1. `resnet50 FixUp`
2. `resnet101 FixUp`
3. `resnet152 FixUp`

arXiv:2104.12928v4 [cs.CV] 7 Dec 2023

If your data distribution shifts, use self-learning

Evgenia Rusak[*] *evgenia.rusak@bethgelab.org*
University of Tübingen

Steffen Schneider[*,†] *steffen.schneider@bethgelab.org*
University of Tübingen

George Pachitariu *george.pachitariu@tue.ai*
University of Tübingen

Luisa Eck *luisa.eck@physics.ox.ac.uk*
University of Oxford

Peter Gehler *pgehler@amazon.com*
Amazon Tübingen

Oliver Bringmann *oliver.bringmann@uni-tuebingen.de*
University of Tübingen

Wieland Brendel[‡] *wieland.brendel@tuebingen.mpg.de*
Max-Planck Institute for Intelligent Systems Tübingen

Matthias Bethge[‡] *matthias.bethge@bethgelab.org*
University of Tübingen

Abstract

We demonstrate that self-learning techniques like entropy minimization and pseudo-labeling are simple and effective at improving performance of a deployed computer vision model under systematic domain shifts. We conduct a wide range of large-scale experiments and show consistent improvements irrespective of the model architecture, the pre-training technique or the type of distribution shift. At the same time, self-learning is simple to use in practice because it does not require knowledge or access to the original training data or scheme, is robust to hyperparameter choices, is straight-forward to implement and requires only a few adaptation epochs. This makes self-learning techniques highly attractive for any practitioner who applies machine learning algorithms in the real world. We present state-of-the-art adaptation results on CIFAR10-C (8.5% error), ImageNet-C (22.0% mCE), ImageNet-R (17.4% error) and ImageNet-A (14.8% error), theoretically study the dynamics of self-supervised adaptation methods and propose a new classification dataset (ImageNet-D) which is challenging even with adaptation.

[*]Equal contribution.
[†]Work initiated during an internship at Amazon Tübingen.
[‡]Joint senior authors.

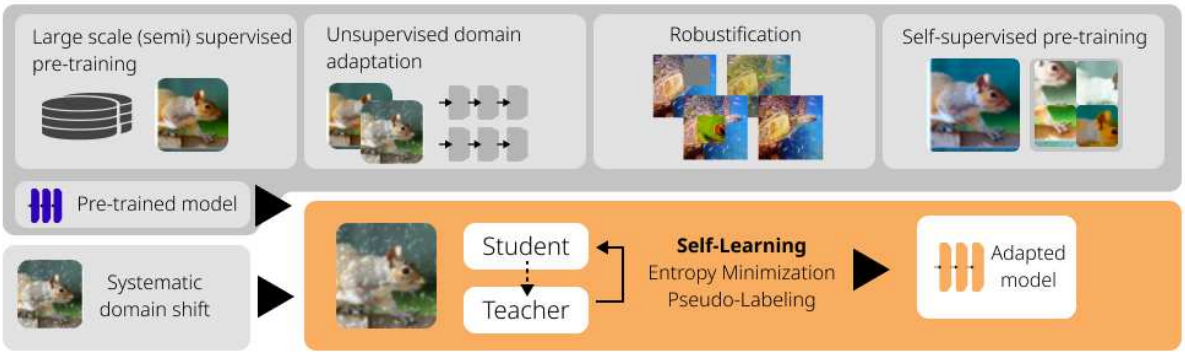


Figure 1: Robustness and adaptation to new datasets has traditionally been achieved by robust pre-training (with hand-selected/data-driven augmentation strategies, or additional data), unsupervised domain adaptation (with access to unlabeled samples from the test set), or, more recently, self-supervised learning methods. We show that on top of these different pre-training tasks, it is always possible (irrespective of architecture, model size or pre-training algorithm) to further adapt models to the target domain with simple self-learning techniques.

1 Introduction

Deep Neural Networks (DNNs) can reach human-level performance in complex cognitive tasks (Brown et al., 2020; He et al., 2016a; Berner et al., 2019) if the distribution of the test data is sufficiently similar to the training data. However, DNNs are known to struggle if the distribution of the test data is shifted relatively to the training data (Geirhos et al., 2018; Dodge & Karam, 2017).

Two largely distinct communities aim to increase the performance of models under test-time distribution shifts: The *robustness community* generally considers ImageNet-scale datasets and evaluates models in an *ad-hoc* scenario. Models are trained on a clean source dataset like ImageNet (Deng et al., 2009), using heavy data augmentation (Hendrycks et al., 2020a; Rusak et al., 2020; Geirhos et al., 2019) and/or large-scale pre-training (Xie et al., 2020a; Mahajan et al., 2018). The trained models are not adapted in any way to test-time distribution shifts. This evaluation scenario is relevant for applications in which very different distribution shifts are encountered in an unpredictable order, and hence misses out on the gains of adaptation to unlabeled samples of the target distribution.

The *unsupervised domain adaptation (UDA) community* often considers smaller-scale datasets and assumes that both the source and the (unlabeled) target dataset are known. Models are trained on both datasets, e.g., with an adversarial objective (Ganin et al., 2016; Tzeng et al., 2017; Hoffman et al., 2018), before evaluation on the target domain data. This evaluation scenario provides optimal conditions for adaptation, but the reliance on the source dataset makes UDA more computationally expensive, more impractical and prevents the use of pre-trained models for which the source dataset is unknown or simply too large. We refer the reader to Farahani et al. (2021) for a review of UDA.

In this work, we consider the *source-free domain adaptation setting*, a middle ground between the classical ad-hoc robustness setting and UDA in which models can adapt to the target distribution but without using the source dataset (Kundu et al., 2020; Kim et al., 2021; Li et al., 2020; Liang et al., 2020). This evaluation scenario is interesting for many practitioners and applications as an extension of the ad-hoc robustness scenario. It evaluates the possible performance of a *deployed* model on a systematic, unseen distribution shift at inference time: an embedded computer vision system in an autonomous car should adapt to changes without being trained on all available training data; an image-based quality control software may not necessarily open-source the images it has been trained on, but still has to be adapted to the lighting conditions at the operation location; a computer vision system in a hospital should perform robustly when tested on a scanner different from the one used for producing the training images—importantly, it might not be known at development time which scanner the vision system will be tested on, and it might be prohibited to share images from many hospitals to run UDA.

Can self-learning methods like *pseudo-labeling* and *entropy-minimization* also be used in this *source-free* domain adaptation setting?[1] To answer this question, we perform an extensive study of several self-learning variants, and find consistent and substantial gains in test-time performance across several robustness and out-of-domain benchmarks and a wide range of models and pre-training methods, including models trained with UDA methods that do not use self-learning, see Figure 1. We also find that self-learning outperforms state-of-the-art source-free domain adaptation methods, namely Test-Time Training which is based on a self-supervised auxiliary objective and continual training (Sun et al., 2020), test-time entropy minimization (Wang et al., 2021), Meta Test-Time Training (Bartler et al., 2022), and (gradient-free) BatchNorm adaptation (Schneider et al., 2020; Nado et al., 2020), and can be improved with additional techniques such as diversity regularization (Mummadi et al., 2021). We perform a large number of ablations to study important design choices for self-learning methods in source-free domain adaptation. Furthermore, we show that a variant of pseudo-labeling with a robust loss function consistently outperforms entropy minimization on ImageNet-scale datasets.

We begin by positioning our work in the existing literature (§ 2) and proceed with an overview of various self-learning variants that have been applied over the past years, and propose a new technique for robust pseudo-labeling (§ 3). We then outline a rigorous experimental protocol that aims to highlight the strengths (and shortcomings) of various self-learning methods. We test various model architectures, with different pre-training schemes covering the most important models in unsupervised domain adaptation, robustness, and large-scale pre-training (§ 4). Using this protocol, we show the effectiveness of self-learning across architectures, models and pre-training schemes (§ 5). We proceed with an in-depth analysis of self-learning, both empirical (§ 6) and theoretical (§ 7). Since the outlined results on ImageNet-C (22.0% mCE), ImageNet-R (17.4% error) and ImageNet-A (14.8%) approach clean performance (11.6% error for our baseline), we propose ImageNet-D as a new benchmark, which we analyse in § 8. We conclude by proposing a set of best practices for evaluating test-time adaptation techniques in the future to ensure scientific rigor and to enable fair model and method comparisons (§ 9).

2 Related Work

Test-Time Adaptation The main question we ask in this work is whether self-learning methods such as entropy minimization and different variants of pseudo-labeling can improve the performance of models when adapted at test-time to data with a distribution shift relative to the training data. Our work is most similar to test-time entropy minimization (TENT; Wang et al., 2021) since entropy minimization is one of our studied test-time adaptation techniques. The conceptual difference between our experiments and Wang et al. (2021) is that Wang et al. (2021) compare TENT to UDA while we argue that UDA can be regarded as a pretraining step, and self-learning can be used on top of any checkpoint pretrained with UDA. Wang et al. (2021) study the effectiveness of entropy minimization across different models, datasets and tasks. We expand upon their experimental setup and show that entropy minimization is effective across a large range of model architectures (convolutional neural networks, Vision Transformers (Dosovitskiy et al., 2021; Caron et al., 2021)), sizes (ResNet50 to EfficientNet-L2 (Xie et al., 2020a; Tan & Le, 2019)), and are orthogonal to robustification (e.g., DeepAugment (Hendrycks et al., 2020a)) and other pre-training schemes. Finally, we perform a large hyperparameter study for test-time entropy minimization, and thereby further improve upon the results reported by Wang et al. (2021).

We also compare our self-learning results to gradient-free adaptation of batch normalization (BN; Ioffe & Szegedy, 2015) statistics (BN adapt; Schneider et al., 2020) who proposed re-estimating the BN statistics on the shifted data distribution.

In Test-Time Training (TTT; Sun et al., 2020), the softmax cross-entropy loss is combined with a rotation prediction task (Gidaris et al., 2018) during pretraining on the source dataset. At test-time, the model is fine-tuned on the unlabeled test data with the self-supervised task. Sun et al. (2020) report results when

[1] Self-learning was defined by Tsypkin (1968) to denote "learning when there is no external indication concerning the correctness of the response of the automatic system to the presented patterns". We opted to use this term to name the superset of pseudo-labeling, entropy minimization and self-supervised learning variants to highlight the fact these methods do not require ground truth labels for adaptation.

adapting to a single test example, and also for online adaptation where the model successively adapts to a stream of data, where the data can either come from the same or a gradually changing distribution. We added a detailed comparison to Sun et al. (2020) in Appendix C.6. Liu et al. (2021) (TTT+++) replace the rotation prediction task with SimCLR (Chen et al., 2020a), and find this modification improves performance.

Eastwood et al. (2022) propose Feature Restoration where the approximate feature distribution under the target data is realigned with the feature distribution under source data. Eastwood et al. (2022) report results on CIFAR10 (among other datasets) for a ResNet18 architecture which allows us to include their method as a baseline. In a concurrent publication, Niu et al. (2022) propose an anti-forgetting test-time adaptation method called EATA, which combine sample-efficient entropy minimization with anti-forgetting weight regularization. They report accuracy numbers on ImageNet-C for a ResNet50 which we include as a baseline. Mummadi et al. (2021) introduce a novel loss to stabilize entropy minimization by replacing the entropy by a non-saturating surrogate and a diversity regularizer based on batch-wise entropy maximization that prevents convergence to trivial collapsed solutions. To partially undo the distribution shift at test time, they additionally propose to add an input transformation module to the model. Mummadi et al. (2021) report results on the highest severity of ImageNet-C for a ResNet50 which we include as a comparison to our results. We note that Mummadi et al. (2021) constitute concurrent unpublished work.

Meta Test-Time Training (MT3; Bartler et al., 2022) combine meta-learning, self-supervision and test-time training to adapt a model trained on clean CIFAR10 (Krizhevsky et al., 2009) to CIFAR10-C (Hendrycks & Dietterich, 2019). We added a detailed comparison to Bartler et al. (2022) in Appendix C.7.

The following papers also consider the setting of test-time adaptation, but are not used as direct baselines in our work, because they study other datasets or tasks. Azimi et al. (2022) show performance improvements when using test-time adaptation on video data. They show adaptation results for the popular test-time adaptation techniques of BN adaptation (Schneider et al., 2020), Test-Time Training (Sun et al., 2020) and test-time entropy minimization (Wang et al., 2021). MEMO (Zhang et al., 2021) maximizes the prediction consistency of different augmented copies regarding a given test sample. We do not consider the single-sample adaptation setting, and comparing our self-learning techniques to MEMO is not a fair setting for MEMO; unsurprisingly, techniques benefiting from multiple samples such as TENT (Wang et al., 2021) outperform MEMO. AdaContrast by Chen et al. (2022) combines pseudo-labeling with other techniques, such as self-supervised contrastive learning on the target domain, soft k-nearest neighbors voting to stabilize the pseudo-labels, as well as consistency and diversity regularization. A direct comparison with their results is difficult because they evaluate on VISDA-C and DomainNet, and so we would need to train our methods on either of the datasets and perform a full hyperparameter search for a fair comparison. We do have one point of comparison: in their paper, Chen et al. (2022) perform better than TENT (Wang et al., 2021). However, we note that Chen et al. (2022) used the default hyperparameters of TENT and did not perform hyperparameter tuning on the new dataset, thus, we would expect the performance of properly tuned TENT to be better than reported in the paper. We expect the additional changes of AdaContrast to further improve upon our simple self-learning baselines. Our work is conceptually similar to virtual adversarial domain adaptation in the fine-tuning phase of DIRT-T (Shu et al., 2018). In contrast to DIRT-T, our objective is simpler and we scale the approach to considerably larger datasets on ImageNet scale. Iwasawa & Matsuo (2021) propose a test-time adaptation algorithm for the task of domain generalization based on computing the distance of each test sample and pseudo-prototypes for each class. Kumar et al. (2020) study the setting of self-learning for gradual domain adaptation. They find that self-learning works better if the data distribution changes slowly. The gradual domain adaptation setting differs from ours: instead of a gradual shift over time, we focus on a fixed shift at test time.

Self-learning for domain adaptation Xie et al. (2020b) introduce "In-N-Out" which uses auxiliary information to boost both in- and out-of-distribution performance. AdaMatch (Berthelot et al., 2021) builds upon FixMatch (Sohn et al., 2020) and can be used for the tasks of unsupervised domain adaptation, semi-supervised learning and semi-supervised domain adaptation as a general-purpose algorithm. Prabhu et al. (2021) propose SENTRY, an algorithm based on judging the predictive consistency of samples from the target domain under different image transformations. Zou et al. (2019) show that different types of confidence regularization can improve the performance of self-learning. A theoretically motivated framework

for self-learning in domain adaptation based on consistency regularization has been proposed by Wei et al. (2020) and then extended by Cai et al. (2021).

The main differences from these works to ours are that they 1) utilize both source and target data during training (i.e., the classical UDA setup) whereas we only require access to unlabeled target data (source-free setup), 2) train their models from scratch whereas we adapt pretrained checkpoints to the unlabeled target data, and 3) are oftentimes more complicated (also in terms of the number of hyperparameters) than our approach due to using more than one term in the objective function. We would like to highlight that utilizing source data should always result in better performance compared to not using source data. Our contribution is to show that self-learning can still be very beneficial with a small compute budget and no access to source data. Our setup targets "deployed systems", e.g., a self-driving car or a detection algorithm in a production line which adapts to the distribution shift "on-the-fly" and cannot (or should not) be retrained from scratch for every new domain shift.

Model selection Gulrajani & Lopez-Paz (2021) show that model selection for hyperparameter tuning is non-trivial for the task of domain generalization, and propose model selection criteria under which models should be selected for this task. Following their spirit, we identify a model selection criterion for test-time adaptation, and rigorously use it in all our experiments. We outperform state-of-the-art techniques which did not disclose their hyperparameter selection protocols.

3 Self-learning for Test-Time Adaptation

Tsypkin (1968) defines self-learning as "learning when there is no external indication concerning the correctness of the response of the automatic system to the presented patterns", and thus, we use this term as a superset of different variants of pseudo-labeling and entropy minimization to highlight that these methods can be used for adaptation to unlabeled data. Different versions of self-learning have been used in both unsupervised domain adaptation (French et al., 2018; Shu et al., 2018), self-supervised representation learning (Caron et al., 2021), and in semi-supervised learning (Xie et al., 2020a). In a typical self-learning setting, a *teacher* network $\mathbf{f}^t$ trained on the source domain predicts labels on the target domain. Then, a *student* model $\mathbf{f}^s$ is fine-tuned on the predicted labels.

In the following, let $\mathbf{f}^t(\mathbf{x})$ denote the logits for sample $\mathbf{x}$ and let $p^t(j|\mathbf{x}) \equiv \sigma_j(\mathbf{f}^t(\mathbf{x}))$ denote the probability for class j obtained from a softmax function $\sigma_j(\cdot)$. Similarly, $\mathbf{f}^s(\mathbf{x})$ and $p^s(j|\mathbf{x})$ denote the logits and probabilities for the student model $\mathbf{f}^s$. For all techniques, one can optionally only admit samples where the probability $\max_j p^t(j|\mathbf{x})$ exceeds some threshold. We consider three popular variants of self-learning: Pseudo-labeling with hard or soft labels, as well as entropy minimization.

Hard Pseudo-Labeling (Lee, 2013; Galstyan & Cohen, 2007). We generate labels using the teacher and train the student on pseudo-labels i using the softmax cross-entropy loss,

$$\ell_H(\mathbf{x}) := -\log p^s(i|\mathbf{x}), \quad i = \operatorname{argmax}_j p^t(j|\mathbf{x}) \tag{1}$$

Usually, only samples with a confidence above a certain threshold are considered for training the student. We test several thresholds but note that thresholding means discarding a potentially large portion of the data which leads to a performance decrease in itself. The teacher is updated after each epoch.

Soft Pseudo-Labeling (Lee, 2013; Galstyan & Cohen, 2007). In contrast to the hard pseudo-labeling variant, we here train the student on class probabilities predicted by the teacher,

$$\ell_S(\mathbf{x}) := -\sum_j p^t(j|\mathbf{x}) \log p^s(j|\mathbf{x}). \tag{2}$$

Soft pseudo-labeling is typically not used in conjunction with thresholding, since it already incorporates the certainty of the model. The teacher is updated after each epoch.

Entropy Minimization (ENT; Grandvalet & Bengio, 2004; Wang et al., 2021). This variant is similar to soft pseudo-labeling, but we no longer differentiate between a teacher and student network. It

corresponds to an "instantaneous" update of the teacher. The training objective becomes

$$\ell_E(\mathbf{x}) := -\sum_j p^s(j|\mathbf{x}) \log p^s(j|\mathbf{x}). \tag{3}$$

Intuitively, self-learning with entropy minimization leads to a sharpening of the output distribution for each sample, making the model more confident in its predictions.

Robust Pseudo-Labeling (RPL). Virtually all introduced self-learning variants use the softmax cross-entropy classification objective. However, the softmax cross-entropy loss has been shown to be sensitive to label noise (Zhang & Sabuncu, 2018; Zhang et al., 2017). In the setting of domain adaptation, inaccuracies in the teacher predictions and, thus, the labels for the student, are inescapable, with severe repercussions for training stability and hyperparameter sensitivity as we show in the results.

As a straight-forward solution to this problem, we propose to replace the cross-entropy loss by a robust classification loss designed to withstand certain amounts of label noise (Ghosh et al., 2017; Song et al., 2020; Shu et al., 2020; Zhang & Sabuncu, 2018). A popular candidate is the *Generalized Cross Entropy (GCE)* loss which combines the noise-tolerant Mean Absolute Error (MAE) loss (Ghosh et al., 2017) with the CE loss. We only consider the hard labels and use the robust GCE loss as the training loss for the student,

$$i = \text{argmax}_j\, p^t(j|\mathbf{x}), \quad \ell_{GCE}(\mathbf{x}, i) := q^{-1}(1 - p^s(i|\mathbf{x})^q), \tag{4}$$

with $q \in (0, 1]$. For the limit case $q \to 0$, the GCE loss approaches the CE loss and for $q = 1$, the GCE loss is the MAE loss (Zhang & Sabuncu, 2018). We test updating the teacher both after every update step of the student (RPL) and once per epoch (RPL$^{\text{ep}}$).

Adaptation parameters. Following Wang et al. (2021), we only adapt the affine scale and shift parameters γ and β following the batch normalization layers (Ioffe & Szegedy, 2015) in most of our experiments. We verify that this type of adaptation works better than full model adaptation for large models in an ablation study in Section 6.

Additional regularization in self-learning Different regularization terms have been proposed as a means to stabilize entropy minimization. Niu et al. (2022) propose an anti-forgetting weight regularization term, Liang et al. (2020); Mummadi et al. (2021); Chen et al. (2022) add a diversity regularizer, and Chen et al. (2022) use an additional consistency regularizer. These methods show improved performance with these regularization terms over simple entropy minimization, but also introduce additional hyperparameters, the tuning of which significantly increases compute requirements. In this work, we do not experiment with additional regularization, as the main point of our analysis is to show that pure self-learning is effective at improving the performance over the unadapted model across model architectures/sizes and pre-training schemes. For practitioners, we note that regularization terms can further improve the performance if the new hyperparameters are tuned properly.

4 Experiment design

Datasets. ImageNet-C (IN-C; Hendrycks & Dietterich, 2019) contains corrupted versions of the 50 000 images in the ImageNet validation set. There are fifteen test and four hold-out corruptions, and there are five severity levels for each corruption. The established metric to report model performance on IN-C is the mean Corruption Error (mCE) where the error is normalized by the AlexNet error, and averaged over all corruptions and severity levels, see Eq. 20, Appendix C.1. ImageNet-R (IN-R; Hendrycks et al., 2020a) contains 30 000 images with artistic renditions of 200 classes of the ImageNet dataset. ImageNet-A (IN-A; Hendrycks et al., 2019) is composed of 7500 unmodified real-world images on which standard ImageNet-trained ResNet50 (He et al., 2016b) models yield chance level performance. CIFAR10 (Krizhevsky et al., 2009) and STL10 (Coates et al., 2011) are small-scale image recognition datasets with 10 classes each, and training sets of 50 000/5000 images and test sets of 10 000/8000 images, respectively. The digit datasets MNIST (Deng, 2012) and MNIST-M (Ganin et al., 2016) both have 60 000 training and 10 000 test images.

Table 1: Self-learning decreases the error on ImageNet-scale robustness datasets. Robust pseudo-labeling generally outperforms entropy minimization.

mCE [%] on IN-C test (↘)	number of parameters	w/o adapt	w/ adapt (Δ) RPL	w/ adapt (Δ) ENT
ResNet50 vanilla (He et al., 2016b)	2.6×10^7	76.7	50.5 (-26.2)	51.6 (-25.1)
ResNet50 DAug+AM (Hendrycks et al., 2020a)	2.6×10^7	53.6	41.7 (-11.9)	42.6 (-11.0)
DenseNet161 vanilla (Huang et al., 2017)	2.8×10^7	66.4	47.0 (-19.4)	47.7 (-18.7)
ResNeXt101$_{32\times8d}$ vanilla (Xie et al., 2017)	8.8×10^7	66.6	43.2 (-23.4)	44.3 (-22.3)
ResNeXt101$_{32\times8d}$ DAug+AM (Hendrycks et al., 2020a)	8.8×10^7	44.5	34.8 (-9.7)	35.5 (-9.0)
ResNeXt101$_{32\times8d}$ IG-3.5B (Mahajan et al., 2018)	8.8×10^7	51.7	40.9 (-10.8)	40.8 (-10.9)
EfficientNet-L2 Noisy Student (Xie et al., 2020a)	4.8×10^8	28.3	**22.0** (-6.3)	23.0 (-5.3)
top1 error [%] on IN-R (↘)				
ResNet50 vanilla (He et al., 2016b)	2.6×10^7	63.8	54.1 (-9.7)	56.1 (-7.7)
EfficientNet-L2 Noisy Student (Xie et al., 2020a)	4.8×10^8	23.5	**17.4** (-6.1)	19.7 (-3.8)
top1 error [%] on ImageNet-A (↘)				
EfficientNet-L2 Noisy Student (Xie et al., 2020a)	4.8×10^8	16.5	**14.8** (-1.7)	15.5 (-1.0)

Hyperparameters. The different self-learning variants have a range of hyperparameters such as the learning rate or the stopping criterion. Our goal is to give a realistic estimation on the performance to be expected in practice. To this end, we optimize hyperparameters for each variant of pseudo-labeling on a hold-out set of IN-C that contains four types of image corruptions ("speckle noise", "Gaussian blur", "saturate" and "spatter") with five different strengths each, following the procedure suggested in Hendrycks & Dietterich (2019). We refer to the hold-out set of IN-C as our *dev* set. On the small-scale datasets, we use the hold-out set of CIFAR10-C for hyperparameter tuning. On all other datasets, we use the hyperparameters obtained on the hold-out sets of IN-C (for large-scale datasets) or CIFAR10-C (on small-scale datasets).

Models for ImageNet-scale datasets. We consider five popular model architectures: ResNet50 (He et al., 2016b), DenseNet161 (Huang et al., 2017), ResNeXt101 (Xie et al., 2017), EfficientNet-L2 (Tan & Le, 2019), and the Vision Transformer (ViT; Dosovitskiy et al., 2021) (see Appendix B.1 for details on the used models). For ResNet50, DenseNet and ResNeXt101, we include a simple *vanilla* version trained on ImageNet only. For ResNet50 and ResNeXt101, we additionally include a state-of-the-art robust version trained with DeepAugment and Augmix (DAug+AM; Hendrycks et al., 2020a)[2]. For the ResNeXt model, we also include a version that was trained on 3.5 billion weakly labeled images (IG-3.5B; Mahajan et al., 2018). For EfficientNet-L2 we select the current state of the art on IN-C which was trained on 300 million images from JFT-300M (Chollet, 2017; Hinton et al., 2014) using a noisy student-teacher protocol (Xie et al., 2020a). Finally, for the ViT, we use the model pretrained with DINO (Caron et al., 2021). We validate the ImageNet and IN-C performance of all considered models and match the originally reported scores (Schneider et al., 2020). For EfficientNet-L2, we match ImageNet top-1 accuracy up to 0.1% points, and IN-C up to 0.6% points mCE.

Models for CIFAR10/ MNIST-scale datasets. For CIFAR10-C experiments, we use three WideResNets (WRN; Zagoruyko & Komodakis, 2016): the first one is trained on clean CIFAR10 and has a depth of 28 and a width of 10, the second one is trained with CIFAR10 with the AugMix protocol (Hendrycks et al., 2020b) and has a depth of 40 and a width of 2, and the third one has a depth of 26 layers, and is pre-trained on clean CIFAR10 using the default training code from `https://github.com/kuangliu/pytorch-cifar`. We used this code-base to also train the ResNet18 and the ResNet50 models on CIFAR10. The remaining small-scale models are trained with UDA methods. We propose to regard any UDA method which requires joint training with source and target data as a pre-training step, similar to regular pre-training on ImageNet, and use self-learning on top of the final checkpoint. We consider two popular UDA methods: self-supervised domain adaptation (UDA-SS; Sun et al., 2019) and Domain-Adversarial Training of Neural Networks (DANN; Ganin et al., 2016). In UDA-SS, the authors seek to align the representations of both domains by performing an auxiliary self-supervised task on both domains simultaneously. In all UDA-SS experiments, we use a WideResNet with a depth of 26 and a width of 16. In DANN, the authors learn a domain-invariant embedding

[2]see leaderboard at `github.com/hendrycks/robustness`

Table 2: Self-learning decreases the error on small-scale datasets, for models pre-trained using data augmentation and unsupervised domain adaptation. Entropy minimization outperforms robust pseudo-labeling.

top1 error [%] on CIFAR10-C (↘)	number of parameters	w/o adapt	w/ adapt (Δ) RPL	w/ adapt (Δ) ENT
WRN-28-10 vanilla (Zagoruyko & Komodakis, 2016)	3.6×10^7	26.5	13.7 (-12.8)	13.3 (-13.2)
WRN-40-2 AM (Hendrycks et al., 2020b)	2.2×10^6	11.2	9.0 (-2.2)	8.5 (-2.7)
WRN-26-1-GN (Bartler et al., 2022)	1.5×10^6	18.6	18.0 (-0.6)	18.4 (0.2)
WRN-26-1-BN (Zagoruyko & Komodakis, 2016)	1.5×10^6	25.8	15.1 (-10.7)	13.1 (-12.7)
WRN-26-16 vanilla (Zagoruyko & Komodakis, 2016)	9.3×10^7	24.2	11.8 (-12.4)	11.2 (-13.0)
WRN-26-16 UDA-SS (Sun et al., 2019)	9.3×10^7	27.7	18.2 (-9.5)	16.7 (-11.0)
WRN-26-16 DANN (Ganin et al., 2016)	9.3×10^7	29.7	28.6 (-1.1)	28.5 (-1.2)
UDA CIFAR10→STL10, top1 error on target [%](↘)				
WRN-26-16 UDA-SS (Sun et al., 2019)	9.3×10^7	28.7	22.9 (-5.8)	21.8 (-6.9)
WRN-26-16 DANN (Ganin et al., 2016)	9.3×10^7	25.0	24.0 (-1.0)	23.9 (-1.1)
UDA MNIST→MNIST-M, top1 error on target [%](↘)				
WRN-26-16 UDA-SS (Sun et al., 2019)	9.3×10^7	4.8	2.4 (-2.4)	2.0 (-2.8)
WRN-26-2 DANN (Ganin et al., 2016)	1.5×10^6	11.4	5.2 (-6.2)	5.1 (-6.3)

by optimizing a minimax objective. For all DANN experiments except for MNIST→MNIST-M, we use the same WRN architecture as above. For the MNIST→MNIST-M experiment, the training with the larger model diverged and we used a smaller WideResNet version with a width of 2. We note that DANN training involves optimizing a minimax objective and is generally harder to tune.

5 Self-learning universally improves models

Self-learning is a powerful learning scheme, and in the following section we show that it allows to perform test-time adaptation on robustified models, models obtained with large-scale pre-training, as well as (already) domain adapted models across a wide range of datasets and distribution shifts. Our main results on large-scale and small-scale datasets are shown in Tables 1 and 2. These summary tables show final results, and all experiments use the hyperparameters we determined separately on the dev set. The self-learning loss function, i.e. soft- or hard-pseudo-labeling / entropy minimization / robust pseudo-labeling, is a hyperparameter itself, and thus, in Tables 1 and 2, we show the overall best results. Results for the other loss functions can be found in Section 6 and in Appendix C.

Self-learning successfully adapts ImageNet-scale models across different model architectures on IN-C, IN-A and IN-R (Table 1). We adapt the vanilla ResNet50, ResNeXt101 and DenseNet161 models to IN-C and decrease the mCE by over 19 percent points in all models. Further, self-learning works for models irrespective of their size: Self-learning substantially improves the performance of the ResNet50 and the ResNext101 trained with DAug+AM, on IN-C by 11.9 and 9.7 percent points, respectively. Finally, we further improve the current state of the art model on IN-C—the EfficientNet-L2 Noisy Student model—and report a new state-of-the-art result of 22% mCE (which corresponds to a top1 error of 17.1%) on this benchmark with test-time adaptation (compared to 28% mCE without adaptation).

Self-learning is not limited to the distribution shifts in IN-C like compression artefacts or blur. On IN-R, a dataset with renditions, self-learning improves both the vanilla ResNet50 and the EfficientNet-L2 model, the latter of which improves from 23.5% to a new state of the art of 17.4% top-1 error. For a vanilla ResNet50, we improve the top-1 error from 63.8% (Hendrycks et al., 2020a) to 54.1%. On IN-A, adapting the EfficientNet-L2 model using self-learning decreases the top-1 error from 16.5% (Xie et al., 2020a) to 14.8% top-1 error, again constituting a new state of the art with test-time adaptation on this dataset. Self-learning can also be used in an online adaptation setting, where the model continually adapts to new samples on IN-C in Fig. 7(i) or IN-R Fig. 7(ii), Appendix C.10.

Adapting a ResNet50 on IN-A with RPL increases the error from 0% to 0.13% (chance level: 0.1%). Thus, an unadapted ResNet50 has 0% accuracy on IN-A by design and this error "is increased" to chance-level with

self-learning. Since all labels on ImageNet-A are wrong by design, predicting wrong labels as pseudo-labels does not lead to improvements beyond restoring chance-level performance.

The finding that self-learning can be effective across model architectures has also been made by Wang et al. (2021) who show improved adaptation performance on CIFAR100-C for architectures based on self-attention (Zhao et al., 2020) and equilibrium solving (Bai et al., 2020), and also by Mummadi et al. (2021) who showed adaptation results for TENT and TENT+ which combines entropy minimization with a diversity regularizer for a DenseNet121 (Huang et al., 2017), a MobileNetV2 (Sandler et al., 2018), a ResNeXt50 (Xie et al., 2017), and a robust model trained with DAug+AM on IN-C and IN-R.

The improvements of self-learning are very stable: In Table 30, Appendix C.9, we show the averaged results across three different seeds for a ResNet50 model adapted with ENT and RPL. The unbiased std is roughly two orders of magnitude lower than any improvement we report in our paper, showcasing the robustness of our results to random initialization.

Self-learning improves robustified and domain adapted models on small-scale datasets (Table 2). We test common domain adaptation techniques like DANN (Ganin et al., 2016) and UDA-SS (Sun et al., 2019), and show that self-learning is effective at further tuning such models to the target domain. We suggest to view unsupervised source/target domain adaptation as a step comparable to pre-training under corruptions, rather than an adaptation technique specifically tuned to the target set—indeed, we can achieve error rates using, e.g., DANN + target adaptation previously only possible with source/target based pseudo-labeling, across different common domain adaptation benchmarks.

For the UDA-SS experiments, we additionally trained a vanilla version with the same architecture using the widely used training code for CIFAR10 at `https://github.com/kuangliu/pytorch-cifar` (1.9k forks, 4.8k stars), and find that the vanilla trained model performs better both with and without adaptation compared to the UDA-SS model. We think that the UDA-SS model would need hyperparameter tuning; we did not perform any tuning for this model, especially because the authors provided scripts with hyperparameters they found to be optimal for different setups. In addition, the clean accuracy of the vanilla model on CIFAR10 (96.5%) is much higher than the average clean accuracy of the UDA-SS model (82.6%), which may explain or imply generally higher robustness under distribution shift (Miller et al., 2021). The finding that self-learning is more effective in the vanilla model compared to the UDA-SS model points towards the hypothesis that the network weights of the vanilla model trained on the source distribution are sufficiently general and can be tuned successfully using only the affine BN statistics, while the weights of the UDA-SS model are already co-adapted to both the source and the target distribution, and thus, self-learning is less effective.

Self-learning also decreases the error on CIFAR10-C of the Wide ResNet model trained with AugMix (AM, Hendrycks et al., 2020b) and reaches a new state of the art on CIFAR10-C of 8.5% top1 error with test-time adaptation.

Self-learning also improves large pre-trained models (Table 3). Unlike BatchNorm adaptation (Schneider et al., 2020), we show that self-learning transfers well to models pre-trained on a large amount of unlabeled data: self-learning decreases the mCE on IN-C of the ResNeXt101 trained on 3.5 billion weakly labeled samples (IG-3.5B, Mahajan et al., 2018) from 51.7% to 40.9%.

Self-learning outperforms other test-time adaptation techniques on IN-C (Tables 4 and 5). The main point of our paper is showing that self-learning is effective across datasets, model sizes and pretraining methods. Here, we analyze whether our simple techniques can compete with other state-of-the-art adaptation methods. Overall, we find that self-learning outperforms several state-of-the-art techniques, but underperforms in some cases, especially when self-learning is combined with other techniques.

By rigorous and fair model selection, we are able to improve upon TENT (Wang et al., 2021), and find that RPL performs better than entropy minimization on IN-C. We also compare to BN adaptation (Schneider et al., 2020), and find that 1) self-learning further improves the performance upon BN adapt, and 2) self-learning improves performance of a model pretrained on a large amount of data (Mahajan et al., 2018) (Table 3) which is a setting where BN adapt failed.

Table 3: Unlike batch norm adaptation, self-learning adapts large-scale models trained on external data.

mCE, test [%] (↘)	w/o adapt	BN adapt	RPL
ResNeXt101 vanilla	66.6	56.8	43.2
ResNeXt101 IG-3.5B	51.7	51.8	**40.9**

Table 4: Self-learning outperforms other test-time-adaptation techniques on IN-C.

mCE [%] on IN-C test (↘)	w/o adapt	BN Adapt	TENT	EATA(lifelong)	RPL	ENT
ResNet50	76.7	62.2	53.5	51.2	**50.5**	51.6

Table 5: Self-learning can further be improved when combining it with other techniques.

	literature results		our results		
	w/o adapt	w/ adapt	w/o adapt	w/ adapt (RPL)	w/ adapt (ENT)
top1 error [%] on IN-C test, sev. 5(↘)					
TTT, ResNet18 (Sun et al., 2020)	86.6	66.3	85.4	**61.9**	62.8
SLR, ResNet50 (Mummadi et al., 2021)	82.0	(46.9)	82.0	54.6	54.7
top1 error [%] on CIFAR10-C (↘)					
TTT+++, ResNet50 (Liu et al., 2021)	29.1	**9.8**	24.9	14.6	12.4
top1 error [%] on CIFAR10-C, sev. 5 (↘)					
MT3, WRN-26-1-GN (Bartler et al., 2022)	35.7	**24.4**	35.6	28.2	27.8
BUFR, ResNet18 (Eastwood et al., 2022)	42.4	**10.6**	42.9	18.4	16.3

Table 6: Vision Transformers can be adapted with self-learning.

mCE on IN-C test [%] (↘)	w/o adapt	w/ adapt affine layers	w/ adapt bottleneck layers	w/ adapt lin. layers	w/ adapt all weights
ViT-S/16	62.3	51.8	46.8	45.2	**43.5**

ENT and RPL outperform the recently published EATA method (Niu et al., 2022). In EATA, high entropy samples are excluded from optimization, and a regularization term is added to prevent the model from forgetting the source distribution. EATA requires tuning of two additional parameters: the threshold for high entropy samples to be discarded and the regularization trade-off parameter β.

RPL, ENT and simple hard PL outperform TTT (Sun et al., 2020); in particular, note that TTT requires a special loss function at training time, while our approach is agnostic to the pre-training phase. A detailed comparison to TTT is included in Appendix C.6. Mummadi et al. (2021) (SLR) is unpublished work and performs better than ENT and RPL on the highest severity of IN-C. SLR is an extension of entropy minimization where the entropy minimization loss is replaced with a version to ensure non-vanishing gradients of high confidence samples, as well as a diversity regularizer; in addition, a trainable module is prepended to the network to partially undo the distribution shift. Mummadi et al. (2021) introduce the hyperparameters δ as the trade-off parameter between their two losses, as well as κ as the momentum in their diversity regularization term. The success of SLR over ENT and RPL shows the promise of extending self-learning methods by additional objectives, and corroborates our findings on the effectiveness of self-learning.

We also compare our approach to Meta Test-Time Training (MT3, Bartler et al., 2022), which combines meta-learning, self-supervision and test-time training for test-time adaptation. We find that both ENT and RPL perform better than MT3: using the same architecture as Bartler et al. (2022), our best error on CIFAR10-C is 18.0% compared to their best result of 24.4%. When exchanging GroupNorm layers (Wu & He, 2018) for BN layers, the error further reduces to 13.1% (Table 11). We thus find that self-learning is more effective when adapting affine BN parameters instead of GN layers, which is consistent with the findings in Schneider et al. (2020). We included a detailed comparison to Bartler et al. (2022) in Appendix C.7.

TTT+++ (Liu et al., 2021) outperforms both ENT and RPL on CIFAR10-C. Since Liu et al. (2021) do not report results on IN-C, it is impossible to judge whether their gains would generalize, although they do report much better results compared to TENT on Visda-C, so TTT+++ might also be effective on IN-C. Similar to TTT, TTT+++ requires a special loss function during pretraining and thus, cannot be used as an out-of-the-box adaptation technique on top of any pretrained checkpoint.

Table 7: RPL (ENT) performs better on IN-C (CIFAR10-C).

	mCE, IN-C dev			err, C10-C
	ResNet50	ResNeXt-101	EffNet-L2	WRN-40
ENT	50.0 ± 0.04	43.0	22.2	**8.5**
RPL	**48.9** ± 0.02	**42.0**	**21.3**	9.0

Table 8: RPL performs best without a threshold.

threshold	0.0	0.5	0.9
mCE on IN-C dev [%]			
no adapt	69.5		
soft PL	60.1		
hard PL	53.8	51.9	52.4
RPL	**49.7**	49.9	51.8

Bottom-Up Feature Restoration (BUFR; Eastwood et al., 2022) outperforms self-learning on CIFAR10-C. The authors note that BUFR is applicable to dataset shifts with measurement shifts which stem from measurement artefacts, but not applicable to more complicated shifts where learning new features would be necessary.

Self-supervised methods based on self-learning allow out-of-the-box test-time adaptation (Table 6). The recently published DINO method (Caron et al., 2021) is another variant of self-supervised learning that has proven to be effective for unsupervised representation learning. At the core, the method uses soft pseudo-labeling. Here, we test whether a model trained with DINO on the source dataset can be test-time adapted on IN-C using DINO to further improve out-of-distribution performance. We highlight that we specifically test the self-supervised DINO objective for its practicality as a test-time adaptation method, and did not switch the DINO objective for ENT or RPL to do test-time adaptation. Since the used model is a vision transformer model, we test different choices of adaptation parameters and find considerable performance improvements in all cases, yielding an mCE of 43.5% mCE at a parameter count comparable to a ResNet50 model. For adapting the affine layers, we follow Houlsby et al. (2019).

6 Understanding test-time adaptation with self-learning

In the following section, we show ablations and interesting insights of using self-learning for test-time adaptation. If not specified otherwise, all ablations are run on the hold-out corruptions of IN-C (our dev set) with a vanilla ResNet50.

Robust pseudo-labeling outperforms entropy minimization on large-scale datasets while the reverse is true on small-scale datasets (Table 7). We find that robust pseudo-labeling consistently improves over entropy minimization on IN-C, while entropy minimization performs better on smaller scale data (CIFAR10, STL10, MNIST). The finding highlights the importance of testing both algorithms on new datasets. The improvement is typically on the order of one percent point.

Robust pseudo-labeling allows usage of the full dataset without a threshold (Table 8). Classical hard labeling needs a confidence threshold (T) for best performance, thereby reducing the dataset size, while best performance for RPL is reached for full dataset training with a threshold T of 0.0. We corroborate the results of Wang et al. (2021) who showed that TENT outperforms standard hard labeling with a threshold on the highest severity of CIFAR10-C and CIFAR100-C. We show that this result transfers to IN-C, for a variety of thresholds and pseudo-labeling variants.

Short update intervals are crucial for fast adaptation (Table 9). Having established that RPL generally performs better than soft- and hard-labeling, we vary the update interval for the teacher. We find that instant updates are most effective. In entropy minimization, the update interval is instant per default.

Adaptation of only affine layers is important in CNNs (Table 10). On IN-C, adapting only the affine parameters after the normalization layers (i.e., the rescaling and shift parameters β and γ) works better on a ResNet50 architecture than adapting all parameters or only the last layer. We indicate the number of adapted parameters in brackets. Note that for Vision Transformers, full model adaptation works better than affine adaptation (see Table 6). We also noticed that on convolutional models with a smaller parameter count like ResNet18, full model adaptation is possible. Wang et al. (2021) also used the affine BN parameters for test-time adaptation with TENT, and report that last layer optimization can improve performance but degrades with further optimization. They suggest that full model optimization does not improve performance

Table 9: RPL performs best with instantaneous updates (ResNet50).

Update interval (RPL)	w/o adapt	none	epoch	instant
mCE, IN-C dev [%]	69.5	54.0	49.7	**49.2**

Table 10: RPL performs best when affine BN parameters are adapted (ResNet50).

Mechanism	w/o adapt	last layer	full model	affine
mCE, IN-C dev [%]	69.5	60.2	51.5	**48.9**
adapted parameters	0	2M	22.6M	5.3k

Table 11: Self-learning works better in models with BN layers compared to models with GN layers, although un-adapted models with GN are more stable under distribution shift compared to models with BN layers.

top1 error [%] on CIFAR10-C (↘)	number of parameters	w/o adapt	w/ adapt (Δ) RPL	w/ adapt (Δ) ENT
WRN-26-1-BN (Zagoruyko & Komodakis, 2016)	1.5×10^6	25.8	15.1 (-10.7)	13.1 (-12.7)
WRN-26-1-GN (Bartler et al., 2022)	1.5×10^6	18.6	18.4 (-0.2)	18.0 (-0.6)
mCE [%] on IN-C test (↘)				
ResNet50 BigTransfer (Kolesnikov et al., 2020)	2.6×10^7	55.0	54.4 (-0.6)	56.4 (+1.4)

Table 12: Our expanded analysis confirms hyperparameter choices from the literature.

Method	short updates	adapt affine params	use BN instead of GN	adapt to full dataset
TENT (Wang et al., 2021)	✓	✓	✓	✓
EATA (Niu et al., 2022)	✓	✓	✓	✗
SRL (Mummadi et al., 2021)	✓	✓	✓	✓
MEMO (Zhang et al., 2021)	✗	✓	✓	✗
RPL/ENT(ours)	✓	✓	✓	✓

at all. In contrast, we find gains with full model adaptation, but stronger gains with adaptation of only affine parameters.

Affine BN parameters work better for test-time adaptation compared to GN parameters. (Table 11). Schneider et al. (2020) showed that models with batch normalization layers are less robust to distribution shift compared to models with group normalization (Wu & He, 2018) layers. However, after adapting BN statistics, the adapted model outperformed the non-adapted GN model. Here, we show that these results also hold for test-time adaptation when adapting a model with GN or BN layers. We show that a WideResNet-26-1 (WRN-26-1) vanilla model with BN layers pretrained on clean CIFAR10 has a much higher error on CIFAR10-C than the same model with GN layers, but it has a much lower error after adaptation. The full results for the WRN-26-1 model can be found in Appendix C.7. Further, we test the pretrained BigTransfer (Kolesnikov et al., 2020) models which have GN layers, and find only small improvements with RPL , and no improvements with ENT. There are no pretrained weights released for the BigTransfer models which have BN layers, thus, a comparison similar to the WRN-26-1 model is not possible. A more detailed discussion on our BigTransfer results as well as a hyperparameter selection study can be found in Appendix E.2.

Hyperparameters obtained on corruption datasets transfer well to real world datasets. When evaluating models, we select the hyperparameters discussed above (the learning rate and the epoch used for early stopping are the most critical ones) on the dev set (full results in Appendix C.2). We note that this technique transfers well to IN-R and -A, highlighting the practical value of corruption robustness datasets for adapting models on real distribution shifts.

Learning rate and number of training epochs are important hyperparameters We tune the learning rate as well as the number of training epochs for all models, except for the EfficientNet-L2 model where we only train for one epoch due to computational constraints. In Tables 18, 19, 20, and 21 in Appendix C.2, we show that both ENT and RPL collapse after a certain number of epochs, showing the inherent instability of pseudo-labeling. While Wang et al. (2021) trained their model only for one epoch with one learning rate,

Table 13: Self-learning leads to an increased ECE compared to the unadapted model.

adaptation	IN-C full	IN-C sev 1	IN-C sev 2	IN-C sev 3	IN-C sev 4	IN-C sev 5
w/o adapt	2.3 ± 0.8	2.3 ± 0.3	2.1 ± 0.3	2.1 ± 0.3	2.2 ± 0.4	2.9 ± 1.6
RPL	10.6 ± 7.3	6.6 ± 0.5	7.9 ± 1.5	8.9 ± 2.1	11.2 ± 3.3	18.7 ± 12.6
ENT	10.6 ± 7.3	6.6 ± 0.5	7.9 ± 1.5	8.9 ± 2.1	11.2 ± 3.3	18.7 ± 12.6

we rigorously perform a full hyperparameter selection on a hold-out set (our dev set), and use the optimal hyperparameters on all tested datasets. We believe that this kind of experimental rigor is essential in order to be able to properly compare methods.

Our analysis confirms hyperparameter choices from the literature Having searched over a broad space of hyperparameters and self-learning algorithms, we identified ENT and RPL as the best performing variants across different model sizes, architectures and pretraining techniques. We summarize the most important hyperparameter choices in Table 12 and compare them to those used in the literature when applying self-learning for test-time adaptation. Wang et al. (2021) identified that on CIFAR-C, entropy minimization outperforms hard PL, and found that affine adaptation of BN parameters works better than full model adaptation, and adapted to the full dataset since TENT does not have a threshold in contrast to hard PL. EATA (Niu et al., 2022) and SRL (Mummadi et al., 2021) are extensions of TENT, and thus, followed their hyperparameter choices. EATA does not adapt to the full dataset as they do not adapt to very similar samples or samples with high entropy values. MEMO (Zhang et al., 2021) adapts to a single sample and adapts all model weights. It would be interesting to study whether the performance of MEMO can be improved when adapting only affine BN parameters.

Pure self-learning hurts calibration, (Table 13) Eastwood et al. (2022) show that approaches based on entropy minimization and pseudo-labeling hurt calibration due to the objective of confidence maximization. We corroborate their results and report the Expected Calibration Error (ECE; Naeini et al., 2015) when adapting a vanilla ResNet50 model with RPL and ENT. We report the mean ECE (lower is better) and standard deviation across corruptions (and severities) below. We observe that both methods increase ECE compared to the unadapted model. The increase in ECE is higher for more severe corruptions which can be explained by a successively stronger distribution shift compared to the source dataset.

Self-learning leads to slightly decreased accuracy on the source dataset (Table 14) To judge the forgetting effect on the source distribution, we calculated the accuracy on clean ImageNet for our adapted ENT and RPL checkpoints. We note that success of self-learning can partially be attributed to correcting the BN statistics of the vanilla model with respect to the distribution shift (Schneider et al., 2020). Thus, we only wish to examine the effect of fine-tuning of the affine BN parameters to the target distribution with respect to the source distribution. Thus, we again correct the mismatched statistics to the source dataset when calculating the accuracy.

We report the mean and standard deviation across corruptions (and severities) below. We observe that both ENT and RPL lead to a decrease in performance on the source dataset, an effect which has also been observed by Niu et al. (2022). The decrease in performance is higher for more severe corruptions which can be explained by a increasingly stronger distribution shift compared to the source dataset. We find that the effect of forgetting is less pronounced in RPL compared to ENT.

Additional experiments and ablation studies, as well as detailed results for all models and datasets can be found in Appendix C. We discuss additional proof-of-concept implementations on the WILDS benchmark (Koh et al., 2021), BigTransfer (BiT; Chen et al., 2020b) models and on self-learning based UDA models in Appendix E. On WILDS, self-learning is effective for the Camelyon17 task with a systematic shift between train, validation and test sets (each set is comprised of different hospitals), while self-learning fails to improve on tasks with mixed domains, such as on the RxRx1 and the FMoW tasks.These results support our claim that self-learning is effective, while showing the important limitation when applied to more diverse shifts.

Table 14: Self-learning leads to slightly decreased accuracy on the source dataset (clean IN).

model	top1 accuracy on ImageNet val [%]
w/o adapt	74.2
RPL adapted to IN-C (avg over corruptions, sev. 1)	74.7 ± 0.6
RPL adapted to IN-C (avg over corruptions, sev. 2)	73.9 ± 0.9
RPL adapted to IN-C (avg over corruptions, sev. 3)	73.0 ± 1.3
RPL adapted to IN-C (avg over corruptions, sev. 4)	71.8 ± 1.8
RPL adapted to IN-C (avg over corruptions, sev. 5)	69.8 ± 3.0
ENT adapted to IN-C (avg over corruptions, sev. 1)	74.3 ± 0.6
ENT adapted to IN-C (avg over corruptions, sev. 2)	73.2 ± 1.1
ENT adapted to IN-C (avg over corruptions, sev. 3)	72.3 ± 1.7
ENT adapted to IN-C (avg over corruptions, sev. 4)	70.7 ± 2.3
ENT adapted to IN-C (avg over corruptions, sev. 5)	68.0 ± 3.9

7 A simple model of stability in self-learning

We observed that different self-learning schemes are optimal for small-scale vs. large-scale datasets and varying amount of classes. We reconsider the used loss functions, and unify them into

$$\begin{aligned} \ell(\mathbf{x}) &= -\sum_j \sigma_j \left(\frac{\mathbf{f}^t(\mathbf{x})}{\tau_t} \right) \log \left(\sigma_j \left(\frac{\mathbf{f}^s(\mathbf{x})}{\tau_s} \right) \right), \\ \mathbf{f}^t(\mathbf{x}) &= \begin{cases} \mathbf{f}(\mathbf{x}), & \text{entropy minimization} \\ \text{sg}(\mathbf{f}(\mathbf{x})), & \text{pseudo-labeling.} \end{cases} \end{aligned} \tag{5}$$

where we introduced student and teacher temperature τ_s and τ_t as parameters in the softmax function and the stop gradient operation sg. To study the learning dynamics, we consider a linear student network $\mathbf{f}^s = \mathbf{w}^s \in \mathbb{R}^d$ and a linear teacher network $\mathbf{f}^t = \mathbf{w}^t \in \mathbb{R}^d$ which are trained on N data points $\{\mathbf{x}_i\}_{i=1}^N$ with a binary cross-entropy loss function $\mathcal{L}$ defined as

$$\begin{aligned} \mathcal{L} &= -\sum_{i=1}^N \ell(\mathbf{x}_i) = -\sum_{i=1}^N \left(\sigma_t(\mathbf{x}_i^\top \mathbf{w}^t) \log \sigma_s(\mathbf{x}_i^\top \mathbf{w}^s) + \sigma_t(-\mathbf{x}_i^\top \mathbf{w}^t) \log \sigma_s(-\mathbf{x}_i^\top \mathbf{w}^s) \right), \\ &\text{where } \sigma_t(z) = \frac{1}{1+e^{-z/\tau_t}} \text{ and } \sigma_s(z) = \frac{1}{1+e^{-z/\tau_s}}. \end{aligned} \tag{6}$$

With stop gradient, student and teacher evolve in time according to

$$\dot{\mathbf{w}}^s = -\nabla_{\mathbf{w}^s} \mathcal{L}\left(\mathbf{w}^s, \mathbf{w}^t\right), \quad \dot{\mathbf{w}}^t = \alpha(\mathbf{w}^s - \mathbf{w}^t), \tag{7}$$

where α is the learning rate of the teacher. Without stop gradient, student and teacher are set equal to each other (following the instant updates we found to perform empirically best), and they evolve as

$$\dot{\mathbf{w}} = -\nabla_{\mathbf{w}} \mathcal{L}(\mathbf{w}), \text{ where } \mathbf{w}^s = \mathbf{w}^t = \mathbf{w}. \tag{8}$$

We restrict the theoretical analysis to the time evolution of the components of $\mathbf{w}^{s,t}$ in direction of two data points $\mathbf{x}_k$ and $\mathbf{x}_l$, $y_k^{s,t} \equiv \mathbf{x}_k^\top \mathbf{w}^{s,t}$ and $y_l^{s,t} \equiv \mathbf{x}_l^\top \mathbf{w}^{s,t}$. All other components $y_i^{s,t}$ with $i \neq k, l$ are neglected to reduce the dimensionality of the equation system. It turns out that the resulting model captures the neural network dynamics quite well despite the drastic simplification of taking only two data points into account (see Figure 2 for a comparison of the model vs. self-learning on the CIFAR-C dataset). We obtain the dynamics:

$$\begin{aligned} \text{with stop gradient: } \dot{y}_k^s &= -\mathbf{x}_k^\top \nabla_{\mathbf{w}^s} \left(\ell(\mathbf{x}_k) + \ell(\mathbf{x}_l) \right), \quad \dot{y}_l^s = -\mathbf{x}_l^\top \nabla_{\mathbf{w}^s} \left(\ell(\mathbf{x}_k) + \ell(\mathbf{x}_l) \right), \\ \dot{y}_k^t &= \alpha(y_k^t - y_k^s), \quad \dot{y}_l^t = \alpha(y_l^t - y_l^s), \\ \text{without stop gradient: } \dot{y}_k &= -\mathbf{x}_k^\top \nabla_{\mathbf{w}} \left(\ell(\mathbf{x}_k) + \ell(\mathbf{x}_l) \right), \quad \dot{y}_l = -\mathbf{x}_l^\top \nabla_{\mathbf{w}} \left(\ell(\mathbf{x}_k) + \ell(\mathbf{x}_l) \right). \end{aligned} \tag{9}$$

With this setup in place, we can derive

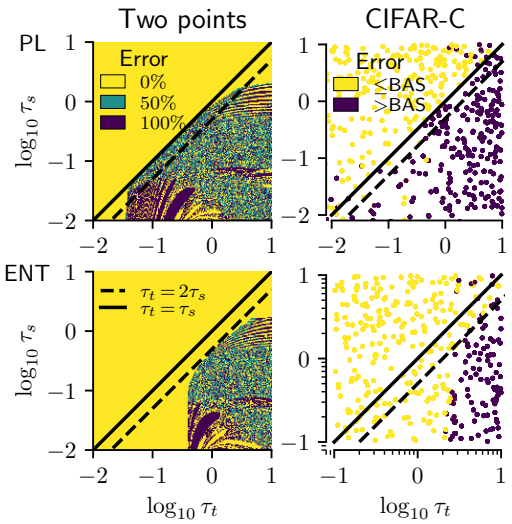


Figure 2: For the two point model, we show accuracy, and for the CIFAR10-C simulation, we show improvement (yellow) vs. degradation (purple) over the non-adapted baseline (BAS). An important convergence criterion for pseudo-labeling (top row) and entropy minimization (bottom row) is the ratio of student and teacher temperatures; it lies at $\tau_s = \tau_t$ for PL, and $2\tau_s = \tau_t$ for ENT. Despite the simplicity of the two-point model, the general convergence regions transfer to CIFAR10-C.

Proposition 7.1 (Collapse in the two-point model)**.** *The student and teacher networks* $\mathbf{w}_s$ *and* $\mathbf{w}_t$ *trained with stop gradient do not collapse to the trivial representation* $\forall \mathbf{x} : \mathbf{x}^\top \mathbf{w}^s = 0, \mathbf{x}^\top \mathbf{w}^t = 0$ *if* $\tau_s > \tau_t$*. The network* $\mathbf{w}$ *trained without stop gradient does not collapse if* $\tau_s > \tau_t/2$*. Proof. see § A.1.* □

We validate the proposition on a simulated two datapoint toy dataset, as well as on the CIFAR-C dataset (Figure 2). In general, the size and location of the region where collapse is observed in the simulated model also depends on the initial conditions, the learning rate and the optimization procedure. An in depth discussion, as well as additional simulations are given in Appendix A.

Entropy minimization with standard temperatures ($\tau_s = \tau_t = 1$) and hard pseudo-labeling ($\tau_t \to 0$) are hence stable. The two-point learning dynamics vanish for soft pseudo-labeling with $\tau_s = \tau_t$, suggesting that one would have to analyze a more complex model with more data points. While this does not directly imply that the learning is unstable at this point, we empirically observe that both entropy minimization and hard labeling outperform soft-labeling in practice.

The finding aligns with empirical work: For instance, Caron et al. (2021) fixed τ_s and varied τ_t during training, and empirically found an upper bound for τ_t above which the training was no longer stable. It also aligns with our findings suggesting that hard-labeling tends to outperform soft-labeling approaches, and soft-labeling performs best when selecting lower teacher temperatures.

In practice, the result suggests that *student temperatures should always exceed the teacher temperatures for pseudo-labeling*, and *student temperatures should always exceed half the teacher temperature for entropy minimization*, which narrows the search space for hyperparameter optimization considerably.

8 Adapting models on a wider range of distribution shifts reveals limitations of robustification and adaptation methods

Robustness datasets on ImageNet-scale have so far been limited to a few selected domains (image corruptions in IN-C, image renditions in IN-R, difficult images for ResNet50 classifiers in IN-A). In order to test our approach on a wider range of complex distribution shifts, we re-purpose the dataset from the Visual Domain Adaptation Challenge 2019 (DomainNet; Saenko et al., 2019) as an additional robustness benchmark.

Table 15: Self-learning decreases the top1 error on IN-D domains with strong initial performance, but fails to improve performance on challenging domains.

domain	Real		Painting		Clipart		Sketch		Infograph		Quickdraw		ImageNet
adapt	w/o	w/	w/o	w/	w/o	w/	w/o	w/	w/o	w/	w/o	w/	w/o
model													
EffNet-L2 Noisy Student	29.2	**27.9**	42.7	**40.9**	45.0	**37.9**	56.4	**51.5**	**77.9**	94.3	**98.4**	99.4	11.6
ResNet50 DAug+AM	39.2	36.5	58.7	53.4	68.4	57.0	75.2	61.3	88.1	83.2	98.2	99.1	23.3
ResNet50 vanilla	40.1	37.3	65.1	57.8	76.0	63.6	82.0	73.0	89.6	85.1	99.2	99.8	23.9

Creation of ImageNet-D The original DomainNet dataset comes with six image styles: "Clipart", "Real", "Infograph", "Painting", "Quickdraw" and "Sketch", and has 345 classes in total, out of which 164 overlap with ImageNet. We map these 164 DomainNet classes to 463 ImageNet classes, e.g., for an image from the "bird" class in DomainNet, we accept all 39 bird classes in ImageNet as valid predictions. ImageNet also has ambiguous classes, e.g., it has separate classes for "cellular telephone" and "dial phone". For these cases, we accept both predictions as valid. In this sense, the mapping from DomainNet to ImageNet is a one-to-many mapping. We refer to the smaller version of DomainNet that is now compatible with ImageNet-trained models as ImageNet-D (IN-D). The benefit of IN-D over DomainNet is this re-mapping to ImageNet classes which allows robustness researchers to easily benchmark on this dataset, without the need of re-training a model (as is common in UDA). We show example images from IN-D in Table 15. The detailed evaluation protocol on IN-D, our label-mapping procedure from DomainNet to ImageNet along with justifications for our design choices and additional analysis are outlined in Appendix D.

The most similar robustness dataset to IN-D is IN-R which contains renditions of ImageNet classes, such as art, cartoons, deviantart, graffiti, embroidery, graphics and others. The benefit of IN-D over IN-R is that in IN-D, the images are separated according to the domain allowing for studying of systematic domain shifts, while in IN-R, the different domains are not distinguished. ImageNet-Sketch (Wang et al., 2019) is a dataset similar to the "Sketch" domain of IN-D.

More robust models perform better on IN-D. To test whether self-learning is helpful for more complex distribution shifts, we adapt a vanilla ResNet50, several robust IN-C models and the EfficientNet-L2 Noisy Student model on IN-D. We use the same hyperparameters we obtained on IN-C dev for all our IN-D experiments[3]. We show our main results in Table 15. Comparing the performance of the vanilla ResNet50 model to its robust DAug+AM variant, we find that the DAug+AM model performs better on all domains, with the most significant gains on the "Clipart", "Painting" and "Sketch" domains. We show detailed results for all domains and all tested models in Appendix D.3, along with results on IN-C and IN-R for comparison. We find that the best performing models on IN-D are also the strongest ones on IN-C and IN-R which indicates good generalization capabilities of the techniques combined for these models, given the large differences between the three considered datasets. The Spearman's rank correlation coefficient between IN-C and IN-D (averaged over all domains) is 0.54, and 0.73 between IN-R and IN-D. Thus, the errors on IN-R are strongly correlated to errors on IN-D which can be explained by the similarity of IN-D and IN-R. We show Spearman's rank correlation cofficients for the individual domains versus IN-C/IN-R in Fig. 9 in Appendix D.5, and find correlation values above 0.8 between IN-R and IN-D for all domains except for the "Real" domain where the coefficient is almost zero. Further, we find that even the best models perform 20 to 30 percentage points worse on IN-D compared to their performance on IN-C or IN-R, indicating that IN-D might be a more challenging benchmark.

All models struggle with some domains of IN-D. The EfficientNet-L2 Noisy Student model obtains the best results on most domains. However, we note that the overall error rates are surprisingly high compared to the model's strong performance on the other considered datasets (IN-A: 14.8% top-1 error, IN-R: 17.4% top-1

[3]In regards to hyperparameter selection, we performed a control experiment where we selected hyperparameters with leave-one-out cross validation—this selection scheme actually performed worse than IN-C parameter selection (see Appendix D.2).

error, IN-C: 22.0% mCE). Even on the "Real" domain closest to clean ImageNet where the EfficientNet-L2 model has a top-1 error of 11.6%, the model only reaches a top-1 error of 29.2%. Self-learning decreases the top-1 error on all domains except for "Infograph" and "Quickdraw". We note that both domains have very high error rates from the beginning and thus hypothesize that the produced pseudo-labels are of low quality.

Error analysis on IN-D. We investigate the errors a ResNet50 model makes on IN-D by analyzing the most frequently predicted classes for different domains to reveal systematic errors indicative of the encountered distribution shifts. We find most errors interpretable: the classifier assigns the label "comic book" to images from the "Clipart" or "Painting" domains, "website" to images from the "Infograph" domain, and "envelope" to images from the "Sketch" domain. Thus, the classifier predicts the domain rather than the class. We find no systematic errors on the "Real" domain which is expected since this domain should be similar to ImageNet. Detailed results on the most frequently predicted classes for different domains can be found in Fig. 9, Appendix D.5.

IN-D should be used as an additional robustness benchmark. While the error rates on IN-C, -R and -A are at a well-acceptable level for our largest EfficientNet-L2 model after adaptation, IN-D performance is consistently worse for all models. We propose to move from isolated benchmark settings like IN-R (single domain) to benchmarks more common in domain adaptation (like DomainNet) and make IN-D publicly available as an easy to use dataset for this purpose.

9 Best practices and evaluation in test-time adaptation

Based on our results as well as our discussion on previous work, we arrive at several proposals on how test-time adaptation should be evaluated in future work to ensure scientific rigor:

1. **Cross-validation:** We propose using the hold-out set of IN-C for model selection of all relevant hyperparameters, and then using these hyperparameters for testing on different datasets.
2. **Comparison to simple baselines:** With proper hyperparameter tuning, very simple baselines can perform on par with sophisticated approaches. This insight is also discussed by Gulrajani & Lopez-Paz (2021) for the setting of domain generalization and by Rusak et al. (2020) for robustness to common corruptions.
3. **Using more robust models:** Test-time adaptation can further improve upon robust models, which were pre-trained with more data or with UDA, or using protocols to increase robustness. A test-time adaptation method will be much more relevant to practitioners if it can improve upon the most robust model they can find for their task.
4. **Important hyperparameters:** We identify several important hyperparameters which affect the final performance in a crucial way, and thus, should be tuned to ensure fair comparisons:
    - **Number of adaptation epochs and learning rate**: The final performance crucially depends on both of these parameters for all models and all methods that we have studied.
    - **Adaptation parameters** While adaptation of affine batch normalization parameters works best for adaptation of CNNs, full adaptation performs best for ViTs. Therefore, it is important to benchmark test-time adaptation for different model architectures and adaptation parameters.

10 Conclusion

We evaluated and analysed how self-learning, an essential component in many unsupervised domain adaptation and self-supervised pre-training techniques, can be applied for adaptation to both small and large-scale image recognition problems common in robustness research. We demonstrated new state-of-the-art adaptation results with the EfficientNet-L2 model on the benchmarks ImageNet-C, -R, and -A, and introduced a new benchmark dataset (ImageNet-D) which remains challenging even after adaptation. Our theoretical analysis shows the influence of the temperature parameter in the self-learning loss function on the training stability and provides guidelines how to choose a suitable value. Based on our extensive experiments, we formulate best practices for future research on test-time adaptation. Across the large diversity of (systematic) distribution

shifts, architectures and pre-training methods we tested in this paper, we found that self-learning almost universally improved test-time performance. An important limitation of current self-learning methods is the observed instability over longer adaptation time frames. While we mitigate this issue through model selection (and show its robustness across synthetic and natural distribution shifts), this might not universally hold across distribution shifts encountered in practice. Concurrent work, e.g. Niu et al. (2022) tackle this problem through modifications of self-learning algorithms, and we think this direction will be important to continue to explore in future work. That being said, we hope that our results encourage both researchers and practitioners to *experiment with self-learning if their data distribution shifts.*

Reproducibility Statement We attempted to make our work as reproducible as possible: We mostly used pre-trained models which are publicly available and we denoted the URL addresses of all used checkpoints; for the checkpoints that were necessary to retrain, we report the Github directories with the source code and used an official or verified reference implementation when available. We report all used hyperparameters in the Appendix and released the code.

Software and Data Code for reproducing results of this paper is available at `https://github.com/bethgelab/robustness`.

Author Contributions StS and ER conceived the project with suggestions from MB and WB. ER ran initial experiments. StS performed large scale experiments on ImageNet-C,R,A with support from PG, ER and WB. ER performed domain adaptation and CIFAR experiments with suggestions from StS. ER implemented all baselines that have been included as comparisons in the manuscript. GP ran DINO adaptation, WILDS, and self-ensembling experiments with suggestions from ER & StS. ER developed the ImageNet-D dataset with suggestions from StS and WB, and performed the large scale experiments on ImageNet-D. LE and StS developed the theoretical model, ER contributed the CIFAR-C experiments. WB, ER and StS wrote the initial manuscript with help from all authors. ER wrote an extensive related work section. ER reviewed and edited the manuscript in the course of the author-reviewer discussion period.

Acknowledgements We thank Roland S. Zimmermann, Yi Zhu, Raghavan Manmatha, Matthias Kümmerer, Matthias Tangemann, Bernhard Schölkopf, Justin Gilmer, Shubham Krishna, Julian Bitterwolf, Berkay Kicanaoglu and Mohammadreza Zolfaghari for helpful discussions on pseudo labeling and feedback on our paper draft. We thank Yasser Jadidi and Alex Smola for support in the setup of our compute infrastructure. We thank the anonymous reviewers from TMLR and our Action Editor for their constructive feedback.

We thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting E.R. and St.S.; St.S. acknowledges his membership in the European Laboratory for Learning and Intelligent Systems (ELLIS) PhD program. This work was supported by the German Federal Ministry of Education and Research (BMBF) through the Tübingen AI Center (FKZ: 01IS18039A), by the Deutsche Forschungsgemeinschaft (DFG) in the priority program 1835 under grant BR2321/5-2 and in the SFB 1233, Robust Vision: Inference Principles and Neural Mechanisms (TP3), project number: 276693517. WB acknowledges financial support via an Emmy Noether Grant funded by the German Research Foundation (DFG) under grant no. BR 6382/1-1 and via the Open Philantropy Foundation funded by the Good Ventures Foundation. WB is a member of the Machine Learning Cluster of Excellence, EXC number 2064/1 – Project number 390727645.

References

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pp. 265–283, 2016. Cited on p. 48.

Fatemeh Azimi, Sebastian Palacio, Federico Raue, Jörn Hees, Luca Bertinetto, and Andreas Dengel. Self-supervised test-time adaptation on video data. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 3439–3448, 2022. Cited on p. 4.

Shaojie Bai, Vladlen Koltun, and J Zico Kolter. Multiscale deep equilibrium models. *Advances in Neural Information Processing Systems*, 33:5238–5250, 2020. Cited on p. 9.

Alexander Bartler, Andre Bühler, Felix Wiewel, Mario Döbler, and Bin Yang. Mt3: Meta test-time training for self-supervised test-time adaption. In *International Conference on Artificial Intelligence and Statistics*, pp. 3080–3090. PMLR, 2022. Cited on pp. 3, 4, 8, 10, 12, 36, and 37

Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemyslaw Debiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, et al. Dota 2 with large scale deep reinforcement learning. *ArXiv preprint*, abs/1912.06680, 2019. URL `https://arxiv.org/abs/1912.06680`. Cited on p. 2.

David Berthelot, Rebecca Roelofs, Kihyuk Sohn, Nicholas Carlini, and Alex Kurakin. Adamatch: A unified approach to semi-supervised learning and domain adaptation, 2021. Cited on p. 4.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL `https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html`. Cited on p. 2.

Tianle Cai, Ruiqi Gao, Jason D Lee, and Qi Lei. A theory of label propagation for subpopulation shift. *arXiv preprint arXiv:2102.11203*, 2021. Cited on p. 5.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. *ArXiv preprint*, abs/2104.14294, 2021. URL `https://arxiv.org/abs/2104.14294`. Cited on pp. 3, 5, 7, 11, 15, and 31

Dian Chen, Dequan Wang, Trevor Darrell, and Sayna Ebrahimi. Contrastive test-time adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 295–305, 2022. Cited on pp. 4 and 6

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PMLR, 2020a. Cited on p. 4.

Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey E. Hinton. Big self-supervised models are strong semi-supervised learners. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020b. URL `https://proceedings.neurips.cc/paper/2020/hash/fcbc95ccdd551da181207c0c1400c655-Abstract.html`. Cited on p. 13.

François Chollet. Xception: Deep learning with depthwise separable convolutions. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pp. 1800–1807. IEEE Computer Society, 2017. doi: 10.1109/CVPR.2017.195. URL `https://doi.org/10.1109/CVPR.2017.195`. Cited on pp. 7 and 30

Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, 2011. Cited on p. 6.

Francesco Croce, Maksym Andriushchenko, Vikash Sehwag, Edoardo Debenedetti, Nicolas Flammarion, Mung Chiang, Prateek Mittal, and Matthias Hein. Robustbench: a standardized adversarial robustness benchmark. *ArXiv preprint*, abs/2010.09670, 2020. URL `https://arxiv.org/abs/2010.09670`. Cited on p. 31.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pp. 248–255. IEEE Computer Society, 2009. doi: 10.1109/CVPR.2009.5206848. URL `https://doi.org/10.1109/CVPR.2009.5206848`. Cited on p. 2.

Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012. Cited on p. 6.

Samuel F. Dodge and Lina J. Karam. A study and comparison of human and deep learning recognition performance under visual distortions. In *International Conference on Computer Communications and Networks, ICCCN 2017*, 2017. Cited on p. 2.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *Proceedings of the 36th International Conference on Machine Learning*, 2021. Cited on pp. 3 and 7

Cian Eastwood, Ian Mason, Christopher KI Williams, and Bernhard Schölkopf. Source-free adaptation to measurement shift via bottom-up feature restoration. In *International Conference on Learning Representations (ICLR)*, 2022. Cited on pp. 4, 10, 11, and 13

Abolfazl Farahani, Sahar Voghoei, Khaled Rasheed, and Hamid R Arabnia. A brief review of domain adaptation. *Advances in data science and information engineering*, pp. 877–894, 2021. Cited on p. 2.

Geoffrey French, Michal Mackiewicz, and Mark H. Fisher. Self-ensembling for visual domain adaptation. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL `https://openreview.net/forum?id=rkpoTaxA-`. Cited on pp. 5, 46, and 48

Aram Galstyan and Paul R. Cohen. Empirical comparison of hard and soft label propagation for relational classification. In *17th international conference on Inductive logic programming*, 2007. Cited on p. 5.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The journal of machine learning research*, 17(1):2096–2030, 2016. Cited on pp. 2, 6, 7, 8, and 9

Robert Geirhos, Carlos R. Medina Temme, Jonas Rauber, Heiko H. Schütt, Matthias Bethge, and Felix A. Wichmann. Generalisation in humans and deep neural networks. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pp. 7549–7561, 2018. URL `https://proceedings.neurips.cc/paper/2018/hash/0937fb5864ed06ffb59ae5f9b5ed67a9-Abstract.html`. Cited on p. 2.

Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A. Wichmann, and Wieland Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL `https://openreview.net/forum?id=Bygh9j09KX`. Cited on pp. 2 and 40

Aritra Ghosh, Himanshu Kumar, and P. S. Sastry. Robust loss functions under label noise for deep neural networks. In Satinder P. Singh and Shaul Markovitch (eds.), *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pp. 1919–1925. AAAI Press, 2017. URL `http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14759`. Cited on p. 6.

Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. *arXiv preprint arXiv:1803.07728*, 2018. Cited on p. 3.

Yves Grandvalet and Yoshua Bengio. Semi-supervised learning by entropy minimization. In *Advances in Neural Information Processing Systems 17 [Neural Information Processing Systems, NIPS 2004, December 13-18, 2004, Vancouver, British Columbia, Canada]*, pp. 529–536, 2004. URL `https://proceedings.neurips.cc/paper/2004/hash/96f2b50b5d3613adf9c27049b2a888c7-Abstract.html`. Cited on p. 5.

Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. In *Proceedings of the 36th International Conference on Machine Learning*, 2021. Cited on pp. 5 and 17

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pp. 770–778. IEEE Computer Society, 2016a. doi: 10.1109/CVPR.2016.90. URL `https://doi.org/10.1109/CVPR.2016.90`. Cited on p. 2.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pp. 770–778. IEEE Computer Society, 2016b. doi: 10.1109/CVPR.2016.90. URL `https://doi.org/10.1109/CVPR.2016.90`. Cited on pp. 6, 7, and 31

Dan Hendrycks and Thomas G. Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL `https://openreview.net/forum?id=HJz6tiCqYm`. Cited on pp. 4, 6, 7, and 39

Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. *ArXiv preprint*, abs/1907.07174, 2019. URL `https://arxiv.org/abs/1907.07174`. Cited on p. 6.

Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. *ArXiv preprint*, abs/2006.16241, 2020a. URL `https://arxiv.org/abs/2006.16241`. Cited on pp. 2, 3, 6, 7, 8, 30, 31, and 40

Dan Hendrycks, Norman Mu, Ekin Dogus Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020b. URL `https://openreview.net/forum?id=S1gmrxHFvB`. Cited on pp. 7, 8, 9, 31, and 40

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. In *NIPS Deep Learning Workshop*, 2014. Cited on pp. 7 and 30

Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *International conference on machine learning*, pp. 1989–1998. Pmlr, 2018. Cited on p. 2.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*, 2019. Cited on p. 11.

Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pp. 2261–2269. IEEE Computer Society, 2017. doi: 10.1109/CVPR.2017.243. URL `https://doi.org/10.1109/CVPR.2017.243`. Cited on pp. 7, 9, 31, and 45

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Francis R. Bach and David M. Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop*

and Conference Proceedings, pp. 448–456. JMLR.org, 2015. URL `http://proceedings.mlr.press/v37/ioffe15.html`. Cited on pp. 3 and 6

Yusuke Iwasawa and Yutaka Matsuo. Test-time classifier adjustment module for model-agnostic domain generalization. *Advances in Neural Information Processing Systems*, 34:2427–2440, 2021. Cited on p. 4.

Youngeun Kim, Donghyeon Cho, Kyeongtak Han, Priyadarshini Panda, and Sungeun Hong. Domain adaptation without source data. *IEEE Transactions on Artificial Intelligence*, 2021. Cited on p. 2.

Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, Tony Lee, Etienne David, Ian Stavness, Wei Guo, Berton A. Earnshaw, Imran S. Haque, Sara Beery, Jure Leskovec, Anshul Kundaje, Emma Pierson, Sergey Levine, Chelsea Finn, and Percy Liang. WILDS: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning (ICML)*, 2021. Cited on pp. 13, 45, and 48

Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*, pp. 491–507. Springer, 2020. Cited on pp. 12 and 46

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. Cited on pp. 4 and 6

Ananya Kumar, Tengyu Ma, and Percy Liang. Understanding self-training for gradual domain adaptation. In *International Conference on Machine Learning*, pp. 5468–5479. PMLR, 2020. Cited on p. 4.

Jogendra Nath Kundu, Naveen Venkat, R Venkatesh Babu, et al. Universal source-free domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4544–4553, 2020. Cited on p. 2.

Dong-Hyun Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *ICML Workshop : Challenges in Representation Learning (WREPL)*, 2013. Cited on p. 5.

Rui Li, Qianfen Jiao, Wenming Cao, Hau-San Wong, and Si Wu. Model adaptation: Unsupervised domain adaptation without source data. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. Cited on p. 2.

Jian Liang, Dapeng Hu, and Jiashi Feng. Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In *International Conference on Machine Learning*, 2020. Cited on pp. 2 and 6

Yuejiang Liu, Parth Kothari, Bastien Germain van Delft, Baptiste Bellot-Gurlet, Taylor Mordan, and Alexandre Alahi. Ttt++: When does self-supervised test-time training fail or thrive? In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021. Cited on pp. 4 and 10

Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens van der Maaten. Exploring the limits of weakly supervised pretraining. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. Cited on pp. 2, 7, 9, 30, and 31

Sébastien Marcel and Yann Rodriguez. Torchvision the machine-vision package of torch. In *ACM International Conference on Multimedia*, 2010. Cited on pp. 31 and 48

Dirk Merkel. Docker: Lightweight linux containers for consistent development and deployment. *Linux J.*, 2014(239), 2014. ISSN 1075-3583. Cited on p. 48.

John P Miller, Rohan Taori, Aditi Raghunathan, Shiori Sagawa, Pang Wei Koh, Vaishaal Shankar, Percy Liang, Yair Carmon, and Ludwig Schmidt. Accuracy on the line: on the strong correlation between out-of-distribution and in-distribution generalization. In *International Conference on Machine Learning*, pp. 7721–7735. PMLR, 2021. Cited on p. 9.

Chaithanya Kumar Mummadi, Robin Hutmacher, Kilian Rambach, Evgeny Levinkov, Thomas Brox, and Jan Hendrik Metzen. Test-time adaptation to distribution shift by confidence maximization and input transformation. *arXiv preprint arXiv:2106.14999*, 2021. Cited on pp. 3, 4, 6, 9, 10, 12, and 13

Zachary Nado, Shreyas Padhy, D Sculley, Alexander D'Amour, Balaji Lakshminarayanan, and Jasper Snoek. Evaluating prediction-time batch normalization for robustness under covariate shift. *ArXiv preprint*, abs/2006.10963, 2020. URL `https://arxiv.org/abs/2006.10963`. Cited on p. 3.

Mahdi Pakdaman Naeini, Gregory Cooper, and Milos Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015. Cited on p. 13.

Shuaicheng Niu, Jiaxiang Wu, Yifan Zhang, Yaofo Chen, Shijian Zheng, Peilin Zhao, and Mingkui Tan. Efficient test-time model adaptation without forgetting. In *Proceedings of the 39th International Conference on Machine Learning*, Proceedings of Machine Learning Research. PMLR, 2022. Cited on pp. 4, 6, 10, 12, 13, and 18

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in PyTorch. In *NIPS Autodiff Workshop*, 2017. Cited on p. 48.

Viraj Prabhu, Shivam Khare, Deeksha Kartik, and Judy Hoffman. Sentry: Selective entropy optimization via committee consistency for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8558–8567, 2021. Cited on p. 4.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pp. 8748–8763. PMLR, 2021. Cited on p. 39.

Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? In *International Conference on Machine Learning*, pp. 5389–5400. PMLR, 2019. Cited on p. 39.

Evgenia Rusak, Lukas Schott, Roland Zimmermann, Julian Bitterwolf, Oliver Bringmann, Matthias Bethge, and Wieland Brendel. Increasing the robustness of dnns against image corruptions by playing the game of noise. *ArXiv preprint*, abs/2001.06057, 2020. URL `https://arxiv.org/abs/2001.06057`. Cited on pp. 2, 17, and 40

Kate Saenko, Xingchao Peng, Ben Usman, Kuniaki Saito, and Ping Hu. *Visual Domain Adaptation Challenge (VisDA-2019)*, 2019. URL `http://ai.bu.edu/visda-2019/`. Cited on p. 15.

Mark Sandler, Andrew G. Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pp. 4510–4520. IEEE Computer Society, 2018. doi: 10.1109/CVPR.2018.00474. URL `http://openaccess.thecvf.com/content_cvpr_2018/html/Sandler_MobileNetV2_Inverted_Residuals_CVPR_2018_paper.html`. Cited on p. 9.

Steffen Schneider, Evgenia Rusak, Luisa Eck, Oliver Bringmann, Wieland Brendel, and Matthias Bethge. Improving robustness against common corruptions by covariate shift adaptation. In *Advances in neural information processing systems*, 2020. Cited on pp. 3, 4, 7, 9, 10, 12, 13, 30, 34, and 36

Jun Shu, Qian Zhao, Keyu Chen, Zongben Xu, and Deyu Meng. Learning adaptive loss for robust learning with noisy labels. *ArXiv preprint*, abs/2002.06482, 2020. URL `https://arxiv.org/abs/2002.06482`. Cited on p. 6.

Rui Shu, Hung H. Bui, Hirokazu Narui, and Stefano Ermon. A DIRT-T approach to unsupervised domain adaptation. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL `https://openreview.net/forum?id=H1q-TM-AW`. Cited on pp. 4, 5, and 47

Kihyuk Sohn, David Berthelot, Chun-Liang Li, Zizhao Zhang, Nicholas Carlini, Ekin D Cubuk, Alex Kurakin, Han Zhang, and Colin Raffel. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In *NeurIPS*, 2020. Cited on p. 4.

Hwanjun Song, Minseok Kim, Dongmin Park, and Jae-Gil Lee. Learning from noisy labels with deep neural networks: A survey. *ArXiv preprint*, abs/2007.08199, 2020. URL `https://arxiv.org/abs/2007.08199`. Cited on p. 6.

Yu Sun, Eric Tzeng, Trevor Darrell, and Alexei A Efros. Unsupervised domain adaptation through self-supervision. *ArXiv preprint*, abs/1909.11825, 2019. URL `https://arxiv.org/abs/1909.11825`. Cited on pp. 7, 8, and 9

Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei Efros, and Moritz Hardt. Test-time training with self-supervision for generalization under distribution shifts. In *International conference on machine learning*, pp. 9229–9248. PMLR, 2020. Cited on pp. 3, 4, 10, 35, and 36

Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pp. 6105–6114. PMLR, 2019. URL `http://proceedings.mlr.press/v97/tan19a.html`. Cited on pp. 3, 7, and 31

O. Tange. Gnu parallel - the command-line power tool. *;login: The USENIX Magazine*, 36(1):42–47, 2011. URL `http://www.gnu.org/s/parallel`. Cited on p. 48.

Rohan Taori, Achal Dave, Vaishaal Shankar, Nicholas Carlini, Benjamin Recht, and Ludwig Schmidt. Measuring robustness to natural distribution shifts in image classification. *Advances in Neural Information Processing Systems*, 33:18583–18599, 2020. Cited on p. 39.

Ya Tsypkin. Self-learning–what is it? *IEEE Transactions on Automatic Control*, 13(6):608–612, 1968. doi: 10.1109/TAC.1968.1099015. Cited on pp. 3 and 5

Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7167–7176, 2017. Cited on p. 2.

Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, CJ Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake Vand erPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1. 0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi: https://doi.org/10.1038/s41592-019-0686-2. Cited on p. 48.

Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. Fully test-time adaptation by entropy minimization. In *International Conference on Learning Representations (ICLR)*, 2021. Cited on pp. 3, 4, 5, 6, 9, 11, 12, and 13

Haohan Wang, Songwei Ge, Zachary Lipton, and Eric P Xing. Learning robust global representations by penalizing local predictive power. In *Advances in Neural Information Processing Systems*, pp. 10506–10518, 2019. Cited on p. 16.

Colin Wei, Kendrick Shen, Yining Chen, and Tengyu Ma. Theoretical analysis of self-training with deep networks on unlabeled data. In *ICLR*, 2020. Cited on p. 5.

Ross Wightman. Pytorch image models. `https://github.com/rwightman/pytorch-image-models`, 2019. Cited on pp. 46 and 48

Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. Cited on pp. 10 and 12

Qizhe Xie, Minh-Thang Luong, Eduard H. Hovy, and Quoc V. Le. Self-training with noisy student improves imagenet classification. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pp. 10684–10695. IEEE, 2020a. doi: 10.1109/CVPR42600.2020.01070. URL https://doi.org/10.1109/CVPR42600.2020.01070. Cited on pp. 2, 3, 5, 7, 8, 30, 31, and 34

Saining Xie, Ross B. Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pp. 5987–5995. IEEE Computer Society, 2017. doi: 10.1109/CVPR.2017.634. URL https://doi.org/10.1109/CVPR.2017.634. Cited on pp. 7, 9, and 31

Sang Michael Xie, Ananya Kumar, Robbie Jones, Fereshte Khani, Tengyu Ma, and Percy Liang. In-n-out: Pre-training and self-training using auxiliary information for out-of-distribution robustness. *arXiv preprint arXiv:2012.04550*, 2020b. Cited on p. 4.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In Richard C. Wilson, Edwin R. Hancock, and William A. P. Smith (eds.), *Proceedings of the British Machine Vision Conference 2016, BMVC 2016, York, UK, September 19-22, 2016*. BMVA Press, 2016. URL http://www.bmva.org/bmvc/2016/papers/paper087/index.html. Cited on pp. 7, 8, and 12

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=Sy8gdB9xx. Cited on p. 6.

Marvin Zhang, Sergey Levine, and Chelsea Finn. Memo: Test time robustness via adaptation and augmentation. *arXiv preprint arXiv:2110.09506*, 2021. Cited on pp. 4, 12, 13, and 38

Zhilu Zhang and Mert R. Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pp. 8792–8802, 2018. URL https://proceedings.neurips.cc/paper/2018/hash/f2925f97bc13ad2852a7a551802feea0-Abstract.html. Cited on p. 6.

Hengshuang Zhao, Jiaya Jia, and Vladlen Koltun. Exploring self-attention for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10076–10085, 2020. Cited on p. 9.

Yang Zou, Zhiding Yu, Xiaofeng Liu, BVK Kumar, and Jinsong Wang. Confidence regularized self-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5982–5991, 2019. Cited on p. 4.

A A two-point model of self-learning

A.1 Proof of Proposition 7.1

Learning dynamics with stop gradient. Computing the stop gradient evolution defined in equation 7 explicitly yields

$$\begin{aligned}\dot{\mathbf{w}}^s &= -\nabla_{\mathbf{w}^s}\mathcal{L} = \frac{1}{\tau_s}\sum_{i=1}^{N}\left(\sigma_t(\mathbf{x}_i^\top\mathbf{w}^t)\sigma_s(-\mathbf{x}_i^\top\mathbf{w}^s) - \sigma_t(-\mathbf{x}_i^\top\mathbf{w}^t)\sigma_s(\mathbf{x}_i^\top\mathbf{w}^s)\right)\mathbf{x}_i \\ \dot{\mathbf{w}}^t &= \alpha(\mathbf{w}^s - \mathbf{w}^t)\end{aligned} \tag{10}$$

The second equality uses the well-known derivative of the sigmoid function, $\partial_z\sigma(z) = \sigma(z)\sigma(-z)$.

The equation system of $2d$ nonlinear, coupled ODEs for $\mathbf{w}^s \in \mathbb{R}^d$ and $\mathbf{w}^t \in \mathbb{R}^d$ in equation 10 is analytically difficult to analyze. Instead of studying the ODEs directly, we act on them with the data points $\mathbf{x}_k^\top$, $k = 1, \ldots, N$, and investigate the dynamics of the components $\mathbf{x}_k^\top\mathbf{w}^{s,t} \equiv y_k^{s,t}$:

$$\begin{aligned}\dot{y}_k^s &= \frac{1}{\tau_s}\sum_{i=1}^{N}\left(\mathbf{x}_i^\top\mathbf{x}_k\right)\left(\sigma_t(y_i^t)\sigma_s(-y_i^s) - \sigma_t(-y_i^t)\sigma_s(y_i^s)\right) \\ \dot{y}_k^t &= \alpha(y_k^s - y_k^t).\end{aligned} \tag{11}$$

The learning rate of each mode y_k^s is scaled by $(\mathbf{x}_k^\top\mathbf{x}_i)$ which is much larger for $i = k$ than for $i \neq k$ in high-dimensional spaces. In the two-point approximation, we consider only the two (in absolute value) largest terms $i = k, l$ for a given k in the sum in equation 11. Any changes that $y_k^{s,t}(t)$ and $y_l^{s,t}(t)$ might induce in other modes $y_i^{s,t}(t)$ are neglected, and so we are left with only four ODEs:

$$\begin{aligned}\dot{y}_k^s =& \frac{1}{\tau_s}\|\mathbf{x}_k\|^2\left(\sigma_t(y_k^t)\sigma_s(-y_k^s) - \sigma_t(-y_k^t)\sigma_s(y_k^s)\right) \\ &+ \frac{1}{\tau_s}(\mathbf{x}_k^\top\mathbf{x}_l)\left(\sigma_t(y_l^t)\sigma_s(-y_l^s) - \sigma_t(-y_l^t)\sigma_s(y_l^s)\right), \\ \dot{y}_l^s =& \frac{1}{\tau_s}\|\mathbf{x}_l\|^2\left(\sigma_t(y_l^t)\sigma_s(-y_l^s) - \sigma_t(-y_l^t)\sigma_s(y_l^s)\right) \\ &+ \frac{1}{\tau_s}(\mathbf{x}_k^\top\mathbf{x}_l)\left(\sigma_t(y_k^t)\sigma_s(-y_k^s) - \sigma_t(-y_k^t)\sigma_s(y_k^s)\right) \\ \dot{y}_k^t =& \alpha(y_k^s - y_k^t),\ \dot{y}_l^t = \alpha(y_l^s - y_l^t).\end{aligned} \tag{12}$$

The fixed points of equation 12 satisfy

$$\dot{y}_k^s = \dot{y}_l^s = \dot{y}_k^t = \dot{y}_l^t = 0. \tag{13}$$

For $\alpha > 0$, requiring $\dot{y}_k^t = \dot{y}_l^t = 0$ implies that $y_k^s = y_k^t$ and $y_l^s = y_l^t$. For $\tau_s = \tau_t$, the two remaining equations $\dot{y}_k^s = \dot{y}_l^s = 0$ vanish automatically so that there are no non-trivial two-point learning dynamics. For $\tau_s \neq \tau_t$, there is a fixed point at $y_k^{s,t} = y_l^{s,t} = 0$ since at this point, each bracket in equation 12 vanishes individually:

$$\sigma_t(y_{k,l})\sigma_s(-y_{k,l}) - \sigma_s(-y_{k,l})\sigma_t(y_{k,l})\Big|_{y_{k,l}=0} = \frac{1}{4} - \frac{1}{4} = 0. \tag{14}$$

At the fixed point $y_k^{s,t} = y_l^{s,t} = 0$, $\mathbf{w}^s$ and $\mathbf{w}^t$ are orthogonal to both $\mathbf{x}_k$ and $\mathbf{x}_l$ and hence classification fails. If this fixed point is stable, $\mathbf{w}^s$ and $\mathbf{w}^t$ will stay at the fixed point once they have reached it, i.e. the model collapses. The fixed point is stable when all eigenvalues of the Jacobian J of the ODE system equation 12 evaluated at $y_k^{s,t} = y_l^{s,t} = 0$ are negative. Two eigenvalues $\lambda_\pm$ are always negative, whereas the two other

eigenvalues $\tilde{\lambda}_\pm$ are positive if $\tau_s > \tau_t$:

$$
\begin{aligned}
J\Big|_{y_k^{s,t}=y_l^{s,t}=0} &= \begin{pmatrix} \frac{-\|\mathbf{x}_k\|^2}{4\tau_s^2} & \frac{-(\mathbf{x}_k^\top \mathbf{x}_l)}{4\tau_s^2} & \frac{\|\mathbf{x}_k\|^2}{4\tau_s\tau_t} & \frac{(\mathbf{x}_k^\top \mathbf{x}_l)}{4\tau_s\tau_t} \\ \frac{-(\mathbf{x}_k^\top \mathbf{x}_l)}{4\tau_s^2} & \frac{-\|\mathbf{x}_l\|^2}{4\tau_s^2} & \frac{(\mathbf{x}_k^\top \mathbf{x}_l)}{4\tau_s\tau_t} & \frac{\|\mathbf{x}_l\|^2}{4\tau_s\tau_t} \\ \alpha & 0 & -\alpha & 0 \\ 0 & \alpha & 0 & -\alpha \end{pmatrix}, \\
\lambda_\pm &= -\frac{1}{16\tau_s^2}\left(A_\pm + \sqrt{A_\pm^2 + 32\alpha\tau_s^2 B(\tau_s/\tau_t - 1)}\right) < 0, \\
\tilde{\lambda}_\pm &= \frac{1}{16\tau_s^2}\left(-A_\pm + \sqrt{A_\pm^2 + 32\alpha\tau_s^2 B(\tau_s/\tau_t - 1)}\right) > 0 \text{ if } \tau_s > \tau_t, \text{ where} \\
A_\pm &= 8\alpha\tau_s^2 \pm B, \quad B = \|\mathbf{x}_k\| + \|\mathbf{x}_l\| + \sqrt{(\|\mathbf{x}_k\| - \|\mathbf{x}_l\|)^2 + 4(\mathbf{x}_k^\top \mathbf{x}_l)^2}
\end{aligned}
\tag{15}
$$

To sum up, training with stop gradient and $\tau_s > \tau_t$ avoids a collapse of the two-point model to the trivial representation $y_k^{s,t} = y_l^{s,t} = 0$ since the fixed point is not stable in this parameter regime.

Learning dynamics without stop gradient Without stop gradient, we set $\mathbf{w}^t = \mathbf{w}^s \equiv \mathbf{w}$ which leads to an additional term in the gradient:

$$
\begin{aligned}
\dot{\mathbf{w}} = -\nabla_{\mathbf{w}}\mathcal{L} =& \frac{1}{\tau_s}\sum_{i=1}^{N}\left(\sigma_t(\mathbf{x}_i^\top \mathbf{w})\sigma_s(-\mathbf{x}_i^\top \mathbf{w}) - \sigma_t(-\mathbf{x}_i^\top \mathbf{w})\sigma_s(\mathbf{x}_i^\top \mathbf{w})\right)\mathbf{x}_i \\
&+ \frac{1}{\tau_t}\sum_{i=1}^{N}\sigma_t(\mathbf{x}_i^\top \mathbf{w})\sigma_t(-\mathbf{x}_i^\top \mathbf{w})\underbrace{\left(\log\sigma_s(\mathbf{x}_i^\top \mathbf{w}) - \log\sigma_s(-\mathbf{x}_i^\top \mathbf{w})\right)}_{=\log\left((1+e^{y_i/\tau_s})/(1+e^{-y_i/\tau_s})\right)=y_i/\tau_s}\mathbf{x}_i.
\end{aligned}
\tag{16}
$$

As before, we focus on the evolution of the two components $y_k = \mathbf{w}^\top \mathbf{x}_k$ and $y_l = \mathbf{w}^\top \mathbf{x}_l$.

$$
\begin{aligned}
\dot{y}_k =& \|\mathbf{x}_k\|^2\left(\frac{1}{\tau_s}\left(\sigma_t(y_k)\sigma_s(-y_k) - \sigma_t(-y_k)\sigma_s(y_k)\right) + \frac{1}{\tau_t}\sigma_t(y_k)\sigma_t(-y_k)y_k\right) \\
&+ (\mathbf{x}_k^\top \mathbf{x}_l)\left(\frac{1}{\tau_s}\left(\sigma_t(y_l)\sigma_s(-y_l) - \sigma_t(-y_l)\sigma_s(y_l)\right) + \frac{1}{\tau_s\tau_t}\sigma_t(y_l)\sigma_t(-y_l)y_l\right) \\
\dot{y}_l =& \|\mathbf{x}_l\|^2\left(\frac{1}{\tau_s}\left(\sigma_t(y_l)\sigma_s(-y_l) - \sigma_t(-y_l)\sigma_s(y_l)\right) + \frac{1}{\tau_t}\sigma_t(y_l)\sigma_t(-y_l)y_l\right) \\
&+ (\mathbf{x}_k^\top \mathbf{x}_l)\left(\frac{1}{\tau_s}\left(\sigma_t(y_k)\sigma_s(-y_k) - \sigma_t(-y_k)\sigma_s(y_k)\right) + \frac{1}{\tau_s\tau_t}\sigma_t(y_k)\sigma_t(-y_k)y_k\right)
\end{aligned}
\tag{17}
$$

There is a fixed point at $y_k = y_l = 0$ where each bracket in equation 17 vanishes individually,

$$
\frac{1}{\tau_s}\left(\sigma_t(y_{k,l})\sigma_s(-y_{k,l}) - \sigma_t(-y_{k,l})\sigma_s(y_{k,l})\right) + \frac{1}{\tau_s\tau_t}\sigma_t(y_{k,l})\sigma_t(-y_{k,l})y_{k,l}\Big|_{y_{k,l}} = 0. \tag{18}
$$

The Jacobian of the ODE system in equation 17 and its eigenvalues evaluated at the fixed point are given by

$$
\begin{aligned}
J\Big|_{y_k=y_l=0} &= \begin{pmatrix} \frac{\|\mathbf{x}_k\|^2}{4\tau_s}\left(\frac{2}{\tau_t} - \frac{1}{\tau_s}\right) & \frac{(\mathbf{x}_k^\top \mathbf{x}_l)}{4\tau_s}\left(\frac{2}{\tau_t} - \frac{1}{\tau_s}\right) \\ \frac{(\mathbf{x}_k^\top \mathbf{x}_l)}{4\tau_s}\left(\frac{2}{\tau_t} - \frac{1}{\tau_s}\right) & \frac{\|\mathbf{x}_l\|^2}{4\tau_s}\left(\frac{2}{\tau_t} - \frac{1}{\tau_s}\right) \end{pmatrix} \\
\lambda_{1,2} &= \frac{1}{8\tau_s}\left(\frac{2}{\tau_t} - \frac{1}{\tau_s}\right)\underbrace{\left(\pm\underbrace{\sqrt{\|\mathbf{x}_k\|^4 + \|\mathbf{x}_l\|^4 - 2\|\mathbf{x}_k\|^2\|\mathbf{x}_l\|^2 + 4(\mathbf{x}_k^\top \mathbf{x}_l)^2}}_{\leq \|\mathbf{x}_k\|^2 + \|\mathbf{x}_l\|^2} + \|\mathbf{x}_k\|^2 + \|\mathbf{x}_l\|^2\right)}_{\geq 0 \text{ with equality if } \mathbf{x}_k = \pm\mathbf{x}_l}.
\end{aligned}
\tag{19}
$$

Hence the fixed point is unstable when $\tau_s > \tau_t/2$ and thus the model without stop gradient does not collapse onto $y_k = y_l = 0$ in this regime, concluding the proof.

A.2 Simulation of the two-point model

For visualization purposes in the main paper, we set $\mathbf{w}^s = \mathbf{w}^t = [0.5, 0.5]^\top$ and train the model using instant gradient updates on the dataset with points $\mathbf{x}_1 = [1, 0]$ and $\mathbf{x}_2 = [0, -1]$ using SGD with learning rate 0.1 and momentum 0.9. We varied student and teacher temperatures on a log-scale with 250 points from 10^{-3} to 10. Qualitatively similar results can be obtained without momentum training, at higher learning rates (most likely due to the implicit learning rate scaling introduced by the momentum term).

Note that the temperature scales for observing the collapse effect depend on the learning rate, and the exact training strategy—lower learning rates can empirically prevent the model from collapsing and shift the convergence region. The result in Figure 2 will hence depend on the exact choice of learning rate (which is currently not considered in our continuous time evolution theory), while the predicted region without collapse is robust to details of the optimization.

To visualize the impact of different hyperparameters, we show variants of the two point model with different learning rates using gradient descent with (Figure 3) and without momentum (Figure 4), and with different start conditions (Figure 5), which all influence the regions where the model degrades, but not the stable regions predicted by our theory.

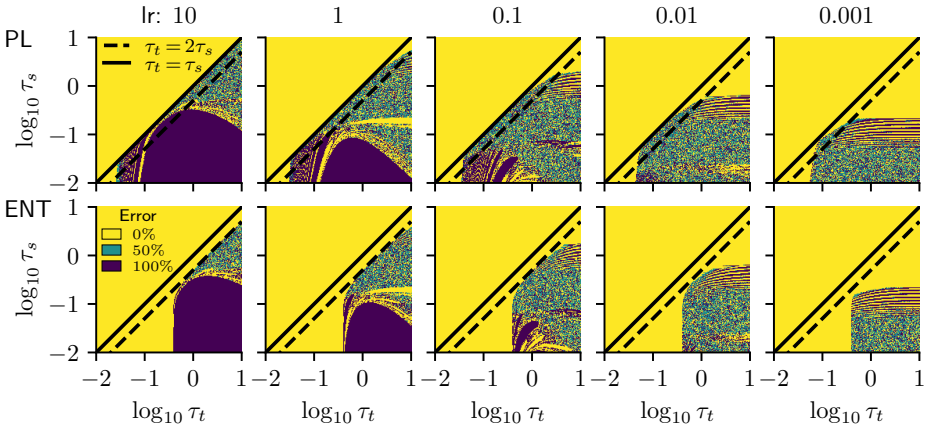


Figure 3: Entropy minimization (top) Training two point model with momentum 0.9 and different learning rates with initialization $\mathbf{w}^s = \mathbf{w}^t = [0.5, 0.5]^\top$.

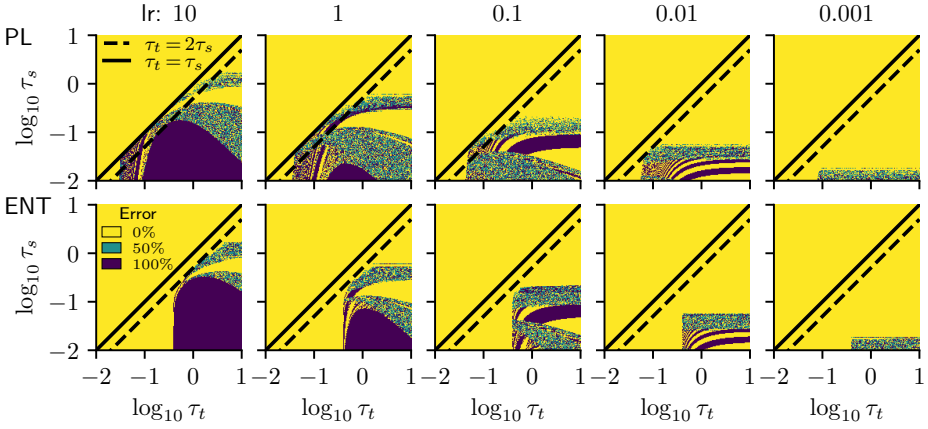


Figure 4: Training a two point model without momentum and different learning rates with initialization $\mathbf{w}^s = \mathbf{w}^t = [0.5, 0.5]^\top$. Note that especially for lower learning rates, longer training would increase the size of the collapsed region.

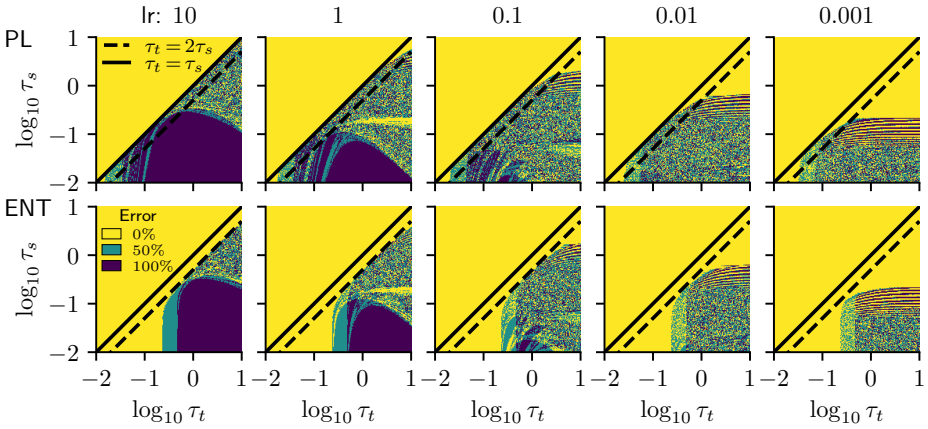


Figure 5: Training a two point model with momentum 0.9 and different learning rates with initialization $\mathbf{w}^s = \mathbf{w}^t = [0.6, 0.3]^\top$.

B Additional information on used models

B.1 Details on all hyperparameters we tested for different models

For all models except EfficientNet-L2, we adapt the batch norm statistics to the test domains following (Schneider et al., 2020). We do not expect significant gains for combining EfficientNet-L2 with batch norm adaptation: as demonstrated in (Schneider et al., 2020), models trained with large amounts of weakly labeled data do not seem to benefit from batch norm adaptation.

ResNet50 models (IN-C) We use a vanilla ResNet50 model and compare soft- and hard-labeling against entropy minimization and robust pseudo-labeling. To find optimal hyperparameters for all methods, we perform an extensive evaluation and test (i) three different adaptation mechanisms (ii) several learning rates 1.0×10^{-4}, 1.0×10^{-3}, 1.0×10^{-2} and 5.0×10^{-2}, (iii) the number of training epochs and (iv) updating the teacher after each epoch or each iteration. For all experiments, we use a batch size of 128. The hyperparameter search is performed on IN-C dev. We then use the optimal hyperparameters to evaluate the methods on the IN-C test set.

ResNeXt101 models The ResNeXt101 model is considerably larger than the ResNet50 model and we therefore limit the number of ablation studies we perform for this architecture. Besides a baseline, we include a state-of-the-art robust version trained with DeepAugment+Augmix (DAug+AM, Hendrycks et al., 2020a) and a version that was trained on 3.5 billion weakly labeled images (IG-3.5B, Mahajan et al., 2018). We only test the two leading methods on the ResNeXt101 models (ENT and RPL). We vary the learning rate in same interval as for the ResNet50 model but scale it down linearly to account for the smaller batch size of 32. We only train the affine batch normalization parameters because adapting only these parameters leads to the best results on ResNet50 and is much more resource efficient than adapting all model parameters. Again, the hyperparameter search is performed only on the development corruptions of IN-C. We then use the optimal hyperparameters to evaluate the methods on the IN-C test set.

EfficientNet-L2 models The current state of the art on IN, IN-C, IN-R and IN-A is an EfficientNet-L2 trained on 300 million images from JFT-300M (Chollet, 2017; Hinton et al., 2014) using a noisy student-teacher protocol (Xie et al., 2020a). We adapt this model for only one epoch due to resource constraints. During the hyperparameter search, we only evaluate three corruptions on the IN-C development set[4] and test the learning rates 4.6×10^{-2}, 4.6×10^{-3}, 4.6×10^{-4} and 4.6×10^{-5}. We use the optimal hyperparameters to evaluate ENT and RPL on the full IN-C test set (with all severity levels).

UDA-SS models We trained the models using the scripts from the official code base at github.com/yueatsprograms/uda_release. We used the provided scripts for the cases: (a) source: CIFAR10, target: STL10 and (b) source: MNIST, target: MNIST-M. For the case (c) source: CIFAR10, target: CIFAR10-C, we used the hyperparameters from case (a) since this case seemed to be the closest match to the new setting. We think that the baseline performance of the UDA-SS models can be further improved with hyperparameter tuning.

DANN models To train models with the DANN-method, we used the PyTorch implementation of this paper at https://github.com/fungtion/DANN_py3. The code base only provides scripts and hyperparameters for the case (b) source: MNIST, target: MNIST-M. For the cases (a) and (c), we used the same optimizer and trained the model for 100 epochs. We think that the baseline performance of the DANN models can be further improved with hyperparameter tuning.

Preprocessing For IN, IN-R, IN-A and IN-D, we resize all images to 256×256 px and take the center 224×224 px crop. The IN-C images are already rescaled and cropped. We center and re-scale the color values with $\mu_{RGB} = [0.485, 0.456, 0.406]$ and $\sigma_{RGB} = [0.229, 0.224, 0.225]$. For the EfficientNet-L2, we follow the

[4]We compare the results of computing the dev set on the 1, 3 and 5 severities versus the 1, 2, 3, 4 and 5 severities on our ResNeXt101 model in the Supplementary material.

procedure in Xie et al. (2020a) and rescale all inputs to a resolution of 507×507 px and then center-crop them to 475×475 px.

B.2 Full list of used models

ImageNet scale models ImageNet trained models (ResNet50, DenseNet161, ResNeXt) are taken directly from torchvision (Marcel & Rodriguez, 2010). The model variants trained with DeepAugment and AugMix augmentations (Hendrycks et al., 2020b;a) are taken from https://github.com/hendrycks/imagenet-r. The weakly-supervised ResNeXt101 model is taken from the PyTorch Hub. For EfficientNet (Tan & Le, 2019), we use the PyTorch re-implementation available at https://github.com/rwightman/gen-efficientnet-pytorch. This is a verified re-implementation of the original work by Xie et al. (2020a). We verify the performance on ImageNet, yielding a 88.23% top-1 accuracy and 98.546% top-5 accuracy which is within 0.2% points of the originally reported result (Xie et al., 2020a). On ImageNet-C, our reproduced baseline achieves 28.9% mCE vs. 28.3% mCE originally reported by Xie et al. (2020a). As noted in the re-implementation, this offset is possible due to minor differences in the pre-processing. It is possible that our adaptation results would improve further when applied on the original codebase by Xie *et al.*.

Small scale models We train the UDA-SS models using the original code base at github.com/yueatsprograms/uda_release, with the hyperparameters given in the provided bash scripts. For our DANN experiments, we use the PyTorch implementation at github.com/fungtion/DANN_py3. We use the hyperparameters in the provided bash scripts.

The following Table 16 contains all models we evaluated on various datasets with references and links to the corresponding source code.

Table 16: Model checkpoints used for our experiments.

Model	Source
WideResNet(28,10) (Croce et al., 2020)	https://github.com/RobustBench/robustbench/tree/master/robustbench
WideResNet(40,2)+AugMix (Croce et al., 2020)	https://github.com/RobustBench/robustbench/tree/master/robustbench
ResNet50 (He et al., 2016b)	https://github.com/pytorch/vision/tree/master/torchvision/models
ResNeXt101, 32×8d (He et al., 2016b)	https://github.com/pytorch/vision/tree/master/torchvision/models
DenseNet (Huang et al., 2017)	https://github.com/pytorch/vision/tree/master/torchvision/models
ResNeXt101, 32×8d (Xie et al., 2017)	https://pytorch.org/hub/facebookresearch_WSL-Images_resnext/
ResNet50+DeepAugment+AugMix (Hendrycks et al., 2020a)	https://github.com/hendrycks/imagenet-r
ResNext101 (Hendrycks et al., 2020a)	https://github.com/hendrycks/imagenet-r
ResNext101 32×8d IG-3.5B (Mahajan et al., 2018)	https://github.com/facebookresearch/WSL-Images/blob/master/hubconf.py
Noisy Student EfficientNet-L2 (Xie et al., 2020a)	https://github.com/rwightman/gen-efficientnet-pytorch
ViT-S/16 (Caron et al., 2021)	https://github.com/facebookresearch/dino

C Detailed and additional Results on IN-C

C.1 Definition of the mean Corruption Error (mCE)

The established performance metric on IN-C is the mean Corruption Error (mCE), which is obtained by normalizing the model's top-1 errors with the top-1 errors of AlexNet across the C=15 test corruptions and S=5 severities:

$$\text{mCE(model)} = \frac{1}{C}\sum_{c=1}^{C}\frac{\sum_{s=1}^{S}\text{err}_{c,s}^{\text{model}}}{\sum_{s=1}^{S}\text{err}_{c,s}^{\text{AlexNet}}}. \tag{20}$$

The AlexNet errors used for normalization are shown in Table 17.

Category	Corruption	top1 error
Noise	Gaussian Noise	0.886428
	Shot Noise	0.894468
	Impulse Noise	0.922640
Blur	Defocus Blur	0.819880
	Glass Blur	0.826268
	Motion Blur	0.785948
	Zoom Blur	0.798360
Weather	Snow	0.866816
	Frost	0.826572
	Fog	0.819324
	Brightness	0.564592
	Contrast	0.853204
Digital	Elastic Transform	0.646056
	Pixelate	0.717840
	JPEG Compression	0.606500
Hold-out Noise	Speckle Noise	0.845388
Hold-out Digital	Saturate	0.658248
Hold-out Blur	Gaussian Blur	0.787108
Hold-out Weather	Spatter	0.717512

Table 17: AlexNet top1 errors on ImageNet-C

C.2 Detailed results for tuning epochs and learning rates

We tune the learning rate for all models and the number of training epochs for all models except the EfficientNet-L2. In this section, we present detailed results for tuning these hyperparameters for all considered models. The best hyperparameters that we found in this analysis, are summarized in Table 22.

Table 18: mCE in % on the IN-C dev set for ENT and RPL for different numbers of training epochs when adapting the affine batch norm parameters of a ResNet50 model.

criterion		ENT			RPL	
lr	10^{-4}	10^{-3}	10^{-2}	10^{-4}	10^{-3}	10^{-2}
epoch						
0	60.2	60.2	60.2	60.2	60.2	60.2
1	54.3	**50.0**	72.5	57.4	51.1	52.5
2	52.4	50.9	96.5	55.8	49.6	57.4
3	51.5	51.0	112.9	54.6	49.2	64.2
4	51.0	52.4	124.1	53.7	49.0	71.0
5	50.7	53.5	131.2	52.9	**48.9**	76.3

Table 19: mCE (↘) in % on the IN-C dev set for different learning rates for EfficientNet-L2. We favor $q = 0.8$ over $q = 0.7$ due to slightly improved robustness to changes in the learning rate in the worst case error setting.

lr (4.6 ×)	base	10^{-3}	10^{-4}	10^{-5}	10^{-6}
ENT	25.5	87.8	25.3	**22.2**	24.1
$\text{RPL}_{q=0.7}$	25.5	60.3	**21.3**	23.3	n/a
$\text{RPL}_{q=0.8}$	25.5	58.2	**21.4**	23.4	n/a

Table 22: The best hyperparameters for all models that we found on IN-C. For all models, we fine-tune only the affine batch normalization parameters and use $q = 0.8$ for RPL. The small batchsize for the EfficientNet model is due to hardware limitations.

Model	Method	Learning rate	batch size	number of epochs
vanilla ResNet50	ENT	1×10^{-3}	128	1
vanilla ResNet50	RPL	1×10^{-3}	128	5
vanilla ResNeXt101	ENT	2.5×10^{-4}	128	1
vanilla ResNeXt101	RPL	2.5×10^{-4}	128	4
IG-3.5B ResNeXt101	ENT	2.5×10^{-4}	128	4
IG-3.5B ResNeXt101	RPL	2.5×10^{-3}	128	2
DAug+AM ResNeXt101	ENT	2.5×10^{-4}	128	1
DAug+AM ResNeXt101	RPL	2.5×10^{-4}	128	4
EfficientNet-L2	ENT	4.6×10^{-5}	8	1
EfficientNet-L2	RPL	4.6×10^{-4}	8	1

Table 20: mCE in % on IN-C dev for entropy minimization for different learning rates and training epochs for ResNeXt101. (div.=diverged)

ENT	Baseline			IG-3.5B			DAug+AM		
lr 2.5 × epoch	1e-4	1e-3	5e-3	1e-4	1e-3	5e-3	1e-4	1e-3	5e-3
BASE	53.6	53.6	53.6	47.4	47.4	47.4	37.4	37.4	37.4
1	**43.0**	92.2	div.	40.9	40.4	58.6	**35.4**	46.4	div.
2	44.8	118.4	div.	39.8	41.5	69.5	35.5	90.8	div.
3	45.4	131.9	div.	39.3	42.6	76.1	35.5	122.5	div.
4	46.7	div.	div.	**39.1**	44.2	84.3	35.6	133.8	div.

Table 21: mCE in % on IN-C dev for robust pseudo-labeling for different learning rates and training epochs for ResNeXt101. (div.=diverged)

RPL	Baseline			IG-3.5B			DAug+AM		
lr 2.5× epoch	1e-4	1e-3	5e-3	1e-4	1e-3	5e-3	1e-4	1e-3	5e-3
BASE	53.6	53.6	53.6	47.4	47.4	47.4	37.4	37.4	37.4
1	43.4	51.3	div.	45.0	39.9	43.6	35.3	35.1	79.1
2	42.3	63.2	div.	43.4	**39.3**	48.2	34.9	35.6	121.2
3	42.0	72.6	div.	42.4	39.4	52.9	34.7	40.1	133.5
4	**42.0**	72.6	div.	42.4	39.4	52.9	**34.7**	40.1	133.5

C.3 Detailed results for all IN-C corruptions

We outline detailed results for all corruptions and models in Table 23. Performance across the severities in the dataset is depicted in Figure 6. All detailed results presented here are obtained by following the model selection protocol outlined in the main text.

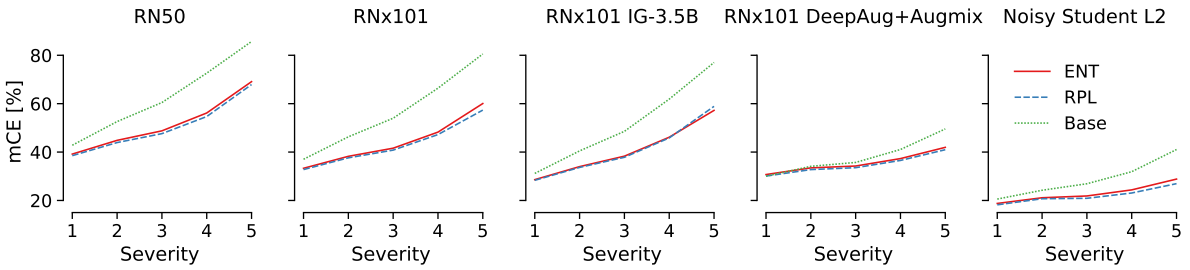


Figure 6: Severity-wise mean corruption error (normalized using the *average* AlexNet baseline error for each corruption) for ResNet50 (RN50), ResNext101 (RNx101) variants and the Noisy Student L2 model. Especially for more robust models (DeepAugment+Augmix and Noisy Student L2), most gains are obtained across higher severities 4 and 5. For weaker models, the baseline variant (Base) is additionally substantially improved for smaller corruptions.

Table 23: Detailed results for each corruption along with mean corruption error (mCE) as reported in Table 2 in the main paper. We show (unnormalized) top-1 error rate averaged across 15 test corruptions along with the mean corruption error (mCE: which is normalized). Hyperparameter selection for both ENT and RPL was carried out on the dev corruptions as outlined in the main text. Mismatch in baseline mCE for EfficientNet-L2 can be most likely attributed to pre-processing differences between the original tensorflow implementation Xie et al. (2020a) and the PyTorch reimplementation we employ. We start with slightly weaker baselines for ResNet50 and ResNext101 than Schneider et al. (2020): ResNet50 and ResNext101 results are slightly worse than previously reported results (typically 0.1% points) due to the smaller batch size of 128 and 32. Smaller batch sizes impact the quality of re-estimated batch norm statistics when computation is performed on the fly Schneider et al. (2020), which is of no concern here due to the large gains obtained by pseudo-labeling.

	gauss	shot	impulse	defocus	glass	motion	zoom	snow	frost	fog	bright	contrast	elastic	pixelate	jpeg	**mCE**
ResNet50																
Baseline (Schneider et al., 2020)																62.2
Baseline (ours)	57.2	59.5	60.0	61.4	62.3	51.3	49.5	54.6	54.1	39.3	29.1	46.7	41.4	38.2	41.8	62.8
ENT	45.5	45.5	46.8	48.4	48.7	40.0	40.3	42.0	46.6	33.2	28.1	42.4	35.2	32.2	35.1	51.6
RPL	44.2	44.4	45.5	47.0	47.4	38.8	39.2	40.7	46.2	32.5	27.7	42.7	34.6	31.6	34.4	50.5
ResNeXt101 Baseline																
Baseline (Schneider et al., 2020)																56.7
Baseline (ours)	52.8	54.1	54.0	55.4	56.8	46.7	46.6	48.5	49.4	36.6	25.4	42.8	37.8	32.5	36.7	56.8
ENT	40.5	39.5	41.4	41.6	43.0	34.1	34.5	35.0	39.4	28.5	24.0	33.8	30.3	27.2	30.5	44.3
RPL	39.4	38.9	39.8	40.3	41.0	33.4	33.8	34.6	38.7	28.0	23.7	31.4	29.8	26.8	30.0	43.2
ResNeXt101 IG-3.5B																
Baseline (Schneider et al., 2020)																51.6
Baseline (ours)	50.7	51.5	53.1	54.2	55.5	45.5	44.7	41.7	42.0	28.1	20.1	33.8	35.4	27.8	33.9	51.8
ENT	38.6	38.3	40.4	41.4	41.5	33.8	33.6	32.2	34.6	24.1	19.7	26.3	27.6	24.2	27.9	40.8
RPL	39.1	39.2	40.8	42.1	42.4	33.7	33.5	31.8	34.7	23.9	19.6	26.1	27.5	23.8	27.5	40.9
ResNeXt101 DeepAug+Augmix																
Baseline (Schneider et al., 2020)																38.0
Baseline (ours)	30.0	30.0	30.2	32.9	35.5	28.9	31.9	33.3	32.8	29.5	22.6	28.4	31.2	23.0	26.5	38.1
ENT	28.7	28.5	29.0	29.8	30.9	26.9	28.0	29.3	30.5	26.2	23.2	26.3	28.5	23.7	26.0	35.5
RPL	28.1	27.8	28.3	29.1	30.1	26.3	27.4	28.8	29.8	25.9	22.7	25.6	27.9	23.2	25.4	34.8
Noisy Student L2																
Baseline (Xie et al., 2020a)																28.3
Baseline (ours)	21.6	22.0	20.5	23.9	40.5	19.8	23.2	22.8	26.9	21.0	15.2	21.2	24.8	17.9	18.6	28.9
ENT	18.5	18.7	17.4	18.8	23.4	16.9	18.8	17.1	19.6	16.8	14.1	16.6	19.6	15.8	16.5	23.0
RPL	17.8	18.0	17.0	18.1	21.4	16.4	17.9	16.4	18.7	15.7	13.6	15.6	19.2	15.0	15.6	22.0

C.4 Detailed results for the CIFAR10-C and UDA adaptation

Table 24: Detailed results for each corruption along with mean error on CIFAR10-C as reported in Table 2 in the main paper.

	gauss	shot	impulse	defocus	glass	motion	zoom	snow	frost	fog	bright	contrast	elastic	pixelate	jpeg	**avg**
WRN-28-10 vanilla																
Baseline	53.0	41.2	44.7	18.5	49.0	22.3	24.4	18.1	25.0	11.2	6.7	17.4	16.2	28.0	22.4	26.5
BN adapt	20.8	17.6	22.7	8.1	28.4	10.9	9.2	14.2	13.0	8.7	6.8	8.5	13.5	12.1	21.0	14.4
ENT	18.5	15.9	20.6	7.8	25.5	10.6	8.5	13.1	12.3	8.3	6.9	8.0	12.6	11.1	18.9	13.3
RPL	19.1	21.4	16.3	8.1	26.4	8.9	10.9	13.7	12.9	6.9	13.1	19.7	8.2	11.4	8.7	13.7
WRN-40-2 AM																
Baseline	19.1	14.0	13.3	6.3	17.1	7.9	7.0	10.4	10.6	8.5	5.9	9.7	9.2	16.8	11.9	11.2
BN adapt	14.1	11.9	13.9	7.2	17.6	8.7	7.9	10.8	10.6	9.0	6.8	9.0	10.9	10.1	14.0	10.8
ENT	10.8	9.1	10.9	6.0	13.4	7.2	6.3	8.4	7.8	7.1	5.7	7.1	9.2	7.4	11.2	8.5
RPL	11.5	11.6	9.6	6.2	14.2	6.5	7.4	8.8	8.2	6.0	9.5	11.9	7.9	8.0	7.6	9.0
WRN-26-16 UDA-SS																
Baseline	26.0	24.7	19.3	22.4	56.2	32.4	32.1	31.7	31.2	26.6	15.8	20.4	26.3	21.5	28.9	27.7
BN adapt	20.5	19.0	15.6	13.5	43.1	19.4	18.3	23.1	21.2	16.2	12.8	14.1	20.9	16.7	23.4	19.9
ENT	16.9	16.7	12.3	11.3	37.6	15.6	14.8	18.3	18.2	13.4	10.8	11.9	17.9	14.4	20.9	16.7
RPL	18.1	17.1	13.2	11.9	41.5	17.3	16.1	20.4	19.1	14.5	11.8	12.7	18.8	18.1	22.6	18.2
WRN-26-16 vanilla																
Baseline	50.8	46.9	39.3	15.9	44.2	20.8	18.8	14.9	17.8	4.8	13.6	20.4	19.0	25.0	10.0	24.2
BN adapt	18.6	20.4	15.6	6.1	24.9	7.4	8.3	11.6	10.8	5.3	11.4	18.9	6.8	9.9	6.8	12.2
ENT	16.7	18.4	13.9	5.7	23.0	6.8	7.9	10.8	9.9	5.0	10.8	17.3	6.4	9.1	6.4	11.2
RPL	16.1	18.0	13.6	6.6	23.2	8.7	9.3	11.9	11.1	6.4	11.7	17.0	7.4	9.0	7.2	11.8

Table 25: Detailed results for the UDA methods reported in Table 2 of the main paper.

	Baseline	BN adapt	RPL	ENT
UDA CIFAR10→STL10, top1 error on target [%](↘)				
WRN-26-16 UDA-SS	28.7	24.6	22.9	21.8
WRN-26-16 DANN	25.0	25.0	24.0	23.9
UDA MNIST→MNIST-M, top1 error on target [%](↘)				
WRN-26-16 UDA-SS	4.8	3.9	2.4	2.0
WRN-26-2 DANN	11.4	6.2	5.2	5.1

C.5 Ablation over the hyperparameter q for RPL

For RPL, we must choose the hyperparameter q. We performed an ablation study over q and show results in Table 26, demonstrating that RPL is robust to the choice of q, with slight preference to higher values. Note: In the initial parameter sweep for this paper, we only compared $q = 0.7$ and $q = 0.8$. Given the result in Table 26, it could be interesting to re-run the models in Table 1 of the main paper with $q = 0.9$, which could yield another (small) improvement in mCE.

Table 26: ImageNet-C dev set mCE in %, vanilla ResNet50, batch size 96. We report the best score across a maximum of six adaptation epochs.

q	0.5	0.6	0.7	0.8	0.9
mCE (dev)	49.5	49.3	49.2	49.2	49.1

C.6 Self-learning outperforms Test-Time Training (Sun et al., 2020)

Sun et al. (2020) use a ResNet18 for their experiments on ImageNet and only evaluate their method on severity 5 of IN-C. To enable a fair comparison, we trained a ResNet18 with both hard labeling and RPL and compare the efficacy of both methods to Test-Time Training in Table 27. For both hard labeling and RPL,

we use the hyperparameters we found for the vanilla ResNet50 model and thus, we expect even better results for hyperparameters tuned on the vanilla ResNet18 model and following our general hyperparameter search protocol.

While all methods (self-learning and TTT) improve the performance over a simple vanilla ResNet18, we note that even the very simple baseline using hard labeling already outperfoms Test-Time Training; further gains are possible with RPL. The result highlights the importance of simple baselines (like self-learning) when proposing new domain adaptation schemes. It is likely that many established DA techniques more complex than the basic self-learning techniques considered in this work will even further improve over TTT and other adaptation approaches developed exclusively in robustness settings.

We report better accuracy numbers achieved with self-learning compared to online TTT, but note that they adapt to a single test sample while we make use of batches of data. Due to the single-image approach, they utilize GN instead of BN, and this may explain the performance gap to a certain degree as we show that BN is more effective for test-time adaptation compared to GN, even though the un-adapted BN models are less robust compared to the un-adapted GN models, as also noted by Sun et al. (2020). Further, Sun et al. (2020) optimize all model parameters while we only optimize the affine BN parameters which works better.

Table 27: Comparison of hard-pseudo labeling and robust pseudo-labeling to Test-Time Training Sun et al. (2020): Top-1 error for a ResNet18 and severity 5 for all corruptions. Simple hard pseudo-labeling already outperforms TTT, robust pseudo labeling over multiple epochs yields additional gains.

	gauss	*shot*	*impulse*	*defocus*	*glass*	*motion*	*zoom*	*snow*	*frost*	*fog*	*bright*	*contrast*	*elastic*	*pixelate*	*jpeg*	**Avg**
vanilla ResNet18	98.8	98.2	99.0	88.6	91.3	88.8	82.4	89.1	83.5	85.7	48.7	96.6	83.2	76.9	70.4	85.4
Test-Time Training	73.7	71.4	73.1	76.3	93.4	71.3	66.6	64.4	81.3	52.4	41.7	**64.7**	55.7	52.2	55.7	66.3
hard PL, (1 epoch)	73.2	70.8	73.6	76.5	75.6	63.9	56.1	59.0	**65.9**	48.4	39.7	85.2	50.4	47.0	51.5	62.5
ENT(1 epoch)	72.8	69.8	73.2	77.2	75.7	63.1	55.5	58.0	68.1	48.0	39.8	92.7	49.6	46.4	51.3	62.8
RPL (4 epochs)	**71.3**	**68.3**	**71.7**	**76.2**	**75.6**	**61.5**	**54.4**	**56.9**	67.1	**47.3**	**39.3**	93.2	**48.9**	**45.7**	**50.4**	**61.9**

C.7 Comparison to Meta Test-Time Training (Bartler et al., 2022)

Bartler et al. (2022) report an error of 24.4% as their top result on CIFAR10-C which is far worse than our top results of 8.5% with an AugMix trained model, or 13.3% with a vanilla trained model, but a different architecture: they used a WRN-26-1 while we report results with a WRN-40-2. Thus, to make the comparison more fair, we tested our approach on their model architecture.

A direct comparison is not straight-forward since Bartler et al. (2022) trained their models using Keras while our code-base is in PyTorch. Therefore, we first trained a baseline model in PyTorch on clean CIFAR10 using their architecture with the standard and widely used CIFAR10 training code available at https://github.com/kuangliu/pytorch-cifar (1.9k forks, 4.8k stars); the baseline test accuracy on clean CIFAR10 using the architecture of Bartler et al. (2022) is at 94.3%. As a second step, we adapted the baseline model with ENT and RPL on CIFAR10-C. Please see Table 28 for our results. BN adaptation is not possible for the model used by Bartler et al. (2022) since they use GroupNorm instead of BatchNorm. Due to the usage of GroupNorm, it is not evident whether the affine GroupNorm parameters should be adapted, or all model parameters. Thus, we tested both, and report the results for both adaptation mechanisms.

We find that adaptation of affine GN layers works better than full model adaptation, consistent with our results for the adaptation of BN layers. As the gains due to ENT and RPL seem lower than for our WRN-40-2 architecture, we hypothesize that the issue lies in the GN layers as the presence of GN instead of BN layers is the main difference between our WRN-40-2 and the new WRN-26-1 model. Therefore, we trained another WRN-26-1 model with BN layers instead of GN, and adapted this model using BN, ENT and RPL. The baseline accuracy on clean CIFAR10 of WRN-26-1-BN is 95.04%. Indeed, we find that adapting the model with BN instead of GN layers leads to much larger gains, which is consistent with findings by Schneider et al. (2020) who found that BN adaptation outperforms non-adapted models trained with GN. Thus, we conclude

that self-learning techniques work better with models which have BN layers and less well with models with GN layers. For both model types (with GN or with BN layers), ENT works better than RPL which is consistent with our other results on small-scale datasets.

Overall, our best result for the model architecture used by Bartler et al. (2022) (after replacing GN layers with BN layers) et al. is 13.1% which is much lower than their best result of 24.4%. Even when using GN layers, our best top-1 error is 18.0% which is significantly lower than the best result of Bartler et al. (2022).

Table 28: Detailed results for our comparison to MT3 (Bartler et al., 2022)

	gauss	shot	impulse	defocus	glass	motion	zoom	snow	frost	fog	bright	contrast	elastic	pixelate	jpeg	**avg**
WRN-26-1-GN vanilla																
Baseline (Bartler et al., 2022)	50.5	47.2	56.1	23.7	51.7	24.3	26.3	25.6	34.4	28.1	13.5	25.0	27.4	55.8	29.8	34.6
adapted (Bartler et al., 2022)	30.1	29.5	41.8	15.6	33.7	22.8	18.7	20.2	18.8	24.1	13.8	22.4	23.7	27.6	22.7	24.4
WRN-26-1-GN vanilla																
Baseline [ours]	39.1	32.0	30.2	10.9	31.9	13.5	13.3	14.1	16.7	10.6	7.3	9.4	14.1	16.1	20.6	18.6
ENT [GN layers, ours]	41.6	30.7	32.8	9.7	30.9	11.8	11.4	13.5	14.9	10.0	7.2	8.9	13.2	13.3	19.8	18.0
RPL [GN layers, ours]	39.3	31.6	30.6	10.6	31.6	13.0	12.6	14.0	16.1	10.4	7.3	9.2	13.9	15.4	20.4	18.4
ENT [full model, ours]	61.2	46.0	37.6	9.1	30.2	11.1	10.4	13.1	14.4	10.0	7.2	8.7	13.1	11.9	19.7	20.3
RPL [full model, ours]	54.1	37.3	33.0	10.2	31.5	11.9	12.4	13.6	15.3	7.3	13.6	20.4	9.2	13.6	10.2	19.6
WRN-26-1-BN vanilla																
Baseline [ours]	55.7	44.8	43.1	15.5	44.2	20.5	21.1	17.2	20.7	6.6	15.6	21.3	23.6	24.7	11.8	25.8
BN adapt [ours]	28.0	28.9	24.2	10.5	31.5	12.7	14.0	18.4	18.1	9.4	17.1	26.9	13.2	15.3	12.7	18.7
ENT [BN layers, ours]	17.9	19.6	14.9	8.1	23.2	9.1	10.4	13.1	13.2	7.2	13.2	18.0	9.9	10.5	8.9	**13.1**
RPL [BN layers, ours]	21.3	22.0	17.9	9.3	26.6	10.5	11.8	15.1	15.0	7.8	14.7	20.8	12.1	11.8	10.4	15.1

C.8 Effect of batch size and linear learning rate scaling

How is self-learning performance affected by batch size constraints? We compare the effect of different batch sizes and linear learning rate scaling. In general, we found that affine adaptation experiments on ResNet50 scale can be run with batch size 128 on a Nvidia V100 GPU (16GB), while only batch size 96 experiments are possible on RTX 2080 GPUs.

The results in Table 29 show that for a ResNet50 model, higher batch size yields a generally better performance.

Table 29: ImageNet-C dev set mCE for various batch sizes with linear learning rate scaling. All results are computed for a vanilla ResNet50 model using RPL with $q = 0.8$, reporting the best score across a maximium of six adaptation epochs.

batch size	16	32	64	80	96	128
learning rate ($\times 10^{-3}$)	0.125	0.250	0.500	0.625	0.750	1
dev mCE	53.8	51.0	49.7	49.3	49.2	48.9

C.9 Performance over different seeds in a ResNet50 on ImageNet-C

To limit the amount of compute, we ran RPL and ENT for our vanilla ResNet50 model three times with the optimal hyperparameters. The averaged results, displayed as "mean (unbiased std)" are:

Table 30: ImageNet-C performance for three seeds on a ResNet50 for ENT and RPL.

ResNet50 + self-learning	mCE on IN-C dev [%]	mCE on IN-C test [%]
ENT	50.0 (0.04)	51.6 (0.04)
RPL	48.9 (0.02)	50.5 (0.03)

C.10 Self-learning as continuous test-test adaptation

We test our method on continuous test-time adaptation where the model adapts to a continuous stream of data from the same domain. In Fig. 7, we display the error of the Noisy Student L2 model while it is being adapted to ImageNet-C and ImageNet-R. The model performance improves as the model sees more data from the new domain. We differentiate continuous test-time adaptation from the online test-time adaptation setting (Zhang et al., 2021) where the model is adapted to each test sample individually, and reset after each test sample.

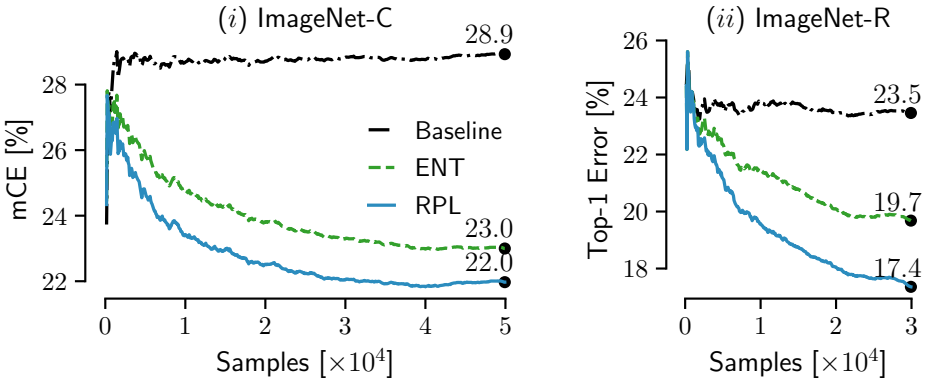


Figure 7: Evolution of error during online adaptation for EfficientNet-L2.

Table 31: Statistics of one-to-many mappings from IN-D to ImageNet.

Number of IN classes one IN-D class is mapped to	1	2	3	4	5	6	7	8	13	28	39	132
Frequency of these mappings	102	32	13	3	5	1	1	2	1	2	1	1

D Detailed and additional Results on IN-D

D.1 Detailed protocol for label mapping from DomainNet to ImageNet

The mapping was first done by comparing the class labels in DomainNet and the synset labels on ImageNet. Afterwards, the resulting label maps were cleaned manually, because simply comparing class label strings resulted in imperfect matches. For example, images of the class "hot dog" in DomainNet were mapped to the class "dog" in IN. Another issue is that IN synset labels of different animal species do not contain the animal name in the text label, e.g., the class "orangutan, orang, orangutang, Pongo pygmaeus" does not contain the word "monkey" and we had to add this class to the hierarchical class "monkey" manually. We verified the mappings by investigating the class-confusion matrix of the true DomainNet class and the predicted ImageNet classes remapped to DomainNet on the "Real" domain, and checked that the predictions lay on the main diagonal, indicating that ImageNet classes have not been forgotten. The statistics for the mappings are shown in Table 31. Most IN-D classes (102) are mapped to one single ImageNet class. A few IN-D classes are mapped to more than 20 ImageNet classes: the IN-D classes "monkey" and "snake" are mapped to 28 ImageNet monkey and snake species classes, the IN-D class "bird" is mapped to 39 ImageNet bird species classes, and the IN-D class "dog" is mapped to 132 ImageNet dog breed classes.

We have considered max-pooling predictions across all sub-classes according to the ImageNet class hierarchy following the approach of Taori et al. (2020) for Youtube-BB and ImageNet-Vid (Recht et al., 2019). However, Radford et al. (2021) note that the resulting mappings are sometimes "much less than perfect", thus, we decided to clean the mappings ourselves to increase the mappings' quality. The full dictionary of the mappings will be released alongside the code.

D.2 Evaluation protocol on IN-D

The domains in IN-D differ in terms of their difficulty for the studied models. Therefore, to calculate an aggregate score, we propose normalizing the error rates by the error achieved by AlexNet on the respective domains to calculate the mean error, following the approach in Hendrycks & Dietterich (2019) for IN-C. This way, we obtain the aggregate score mean Domain Error (mDE) by calculating the mean over different domains,

$$\mathrm{DE}_d^f = \frac{E_d^f}{E_d^{\mathrm{AlexNet}}}, \qquad \mathrm{mDE} = \frac{1}{D}\sum_{d=1}^{D} E_d^f, \tag{21}$$

where E_d^f is the top-1 error of a classifier f on domain d.

Leave-one-out-cross-validation For all IN-D results we report in this paper, we chose the hyperparameters on the IN-C dev set. We tried a different model selection scheme on IN-D as a control experiment with "Leave one out cross-validation" (L1outCV): with a round-robin procedure, we choose the hyperparameters for the test domain on all other domains. We select the same hyperparameters as when tuning on the "dev" set: For the ResNet50 model, we select over the number of training epochs (with a maximum of 7 training epochs) and search for the optimal learning rate in the set [0.01, 0.001, 0.0001]. For the EfficientNet-L2 model, we train only for one epoch as before and select the optimal learning rate in the set [4.6×10^{-3}, 4.6×10^{-4}, 4.6×10^{-5}, 4.6×10^{-6}]. This model selection leads to worse results both for the ResNet50 and the EfficientNet-L2 models, highlighting the robustness of our model selection process, see Table 32.

Table 32: mDE in % on IN-D for different model selection strategies.

model	model selection	
	L1outCV	IN-C dev
ResNet50 RPL$_{q=0.8}$	81.3	76.1
ResNet50 ENT	82.4	77.3
EfficientNet-L2 ENT	69.2	66.8
EfficientNet-L2 RPL$_{q=0.8}$	69.1	67.2

D.3 Detailed results for robust ResNet50 models on IN-D

We show detailed results for all models on IN-D for vanilla evaluation (Table 33) BN adaptation (Table 34), RPL$_{q=0.8}$ (Table 35) and ENT(Table 36). For RPL$_{q=0.8}$ and ENT, we use the same hyperparameters that we chose on our IN-C 'dev' set. This means we train the models for 5 epochs with RPL$_{q=0.8}$ and for one epoch with ENT.

We evaluate the pre-trained and public checkpoints of SIN (Geirhos et al., 2019), ANT (Rusak et al., 2020), ANT+SIN (Rusak et al., 2020), AugMix (Hendrycks et al., 2020b), DeepAugment (Hendrycks et al., 2020a) and DeepAug+Augmix (Hendrycks et al., 2020a) in the following tables.

Table 33: Top-1 error on IN-D in % as obtained by robust ResNet50 models. For reference, we also show the mCE on IN-C and the top-1 error on IN-R. See main test for model references.

Model	Clipart	Infograph	Painting	Quickdraw	Real	Sketch	mDE	IN-C	IN-R
vanilla	76.0	89.6	65.1	99.2	40.1	82.0	88.2	76.7	63.9
SIN	71.3	88.6	62.6	97.5	40.6	77.0	85.6	69.3	58.5
ANT	73.4	88.9	63.3	99.2	39.9	80.8	86.9	62.4	61.0
ANT+SIN	68.4	88.6	60.6	95.5	40.8	70.3	83.1	60.7	53.7
AugMix	70.8	88.6	62.1	99.1	39.0	78.5	85.4	65.3	58.9
DeepAugment	72.0	88.8	61.4	98.9	39.4	78.5	85.6	60.4	57.8
DeepAug+Augmix	68.4	88.1	58.7	98.2	39.2	75.2	83.4	53.6	53.2

Table 34: Top1 error on IN-D in % as obtained by state-of-the-art robust ResNet50 models and batch norm adaptation, with a batch size of 128. See main text for model references.

Model	Clipart	Infograph	Painting	Quickdraw	Real	Sketch	mDE
vanilla	70.2	88.2	63.5	97.8	41.1	78.3	80.2
SIN	67.3	89.7	62.2	97.2	44.0	75.2	79.6
ANT	69.2	89.4	63.0	97.5	42.9	79.5	80.7
ANT+SIN	64.9	88.2	60.0	96.8	42.6	73.0	77.8
AugMix	66.9	88.1	61.2	97.1	40.4	75.0	78.4
DeepAugment	66.6	89.7	60.0	97.2	42.5	75.1	78.8
DeepAug+Augmix	61.9	85.7	57.5	95.3	40.2	69.2	74.9

Table 35: Top-1 error on IN-D in % as obtained by state-of-the-art robust ResNet50 models and RPL$_{q=0.8}$. See main text for model references.

Model	Clipart	Infograph	Painting	Quickdraw	Real	Sketch	mDE
vanilla	63.6	85.1	57.8	99.8	37.3	73.0	76.1
SIN	60.8	86.4	56.0	99.0	37.8	67.0	76.8
ANT	63.4	86.3	57.7	99.2	37.7	71.0	78.1
ANT+SIN	61.5	86.4	56.8	97.0	39.0	67.1	76.1
AugMix	59.7	83.4	54.1	98.2	35.6	70.1	74.6
DeepAugment	58.1	84.6	53.3	99.0	36.2	64.2	74.8
DeepAug+Augmix	57.0	83.2	53.4	99.1	36.5	61.3	72.6

Table 36: Top-1 error on IN-D in % as obtained by state-of-the-art robust ResNet50 models and ENT. See main text for references to the used models.

Model	Clipart	Infograph	Painting	Quickdraw	Real	Sketch	mDE
vanilla	65.1	85.8	59.2	98.5	38.4	75.8	77.3
SIN	62.1	87.0	57.3	99.1	39.0	68.6	75.5
ANT	64.2	86.9	58.7	97.1	38.8	72.8	76.5
ANT+SIN	62.2	86.8	57.7	95.8	40.1	68.7	75.2
AugMix	60.2	84.6	55.8	97.6	36.8	72.0	74.4
DeepAugment	59.5	85.7	54.4	98.0	37.1	66.4	73.3
DeepAug+Augmix	58.4	84.3	54.7	98.5	38.1	63.6	72.7

Table 37: mDE on IN-D in % as obtained by robust ResNet50 models with a baseline evaluation, batch norm adaptation, $\text{RPL}_{q=0.8}$ and ENT. See main text for model references.

	mDE on IN-D (↘)			
Model	Baseline	BN adapt	$\text{RPL}_{q=0.8}$	ENT
vanilla	88.2	80.2	76.1	77.3
SIN	85.6	79.6	76.8	75.5
ANT	86.9	80.7	78.1	76.5
ANT+SIN	**83.1**	77.8	76.1	75.2
AugMix	85.4	78.4	74.6	74.4
DeepAugment	85.6	78.8	74.8	73.3
DeepAugment+Augmix	83.4	**74.9**	**72.6**	**72.7**

The summary results for all models are shown in Table 37.

We show the top-1 error for the different IN-D domains versus training epochs for a vanilla ResNet50 in Fig. 8. We indicate the epochs 1 and 5 at which we extract the errors with dashed black lines.

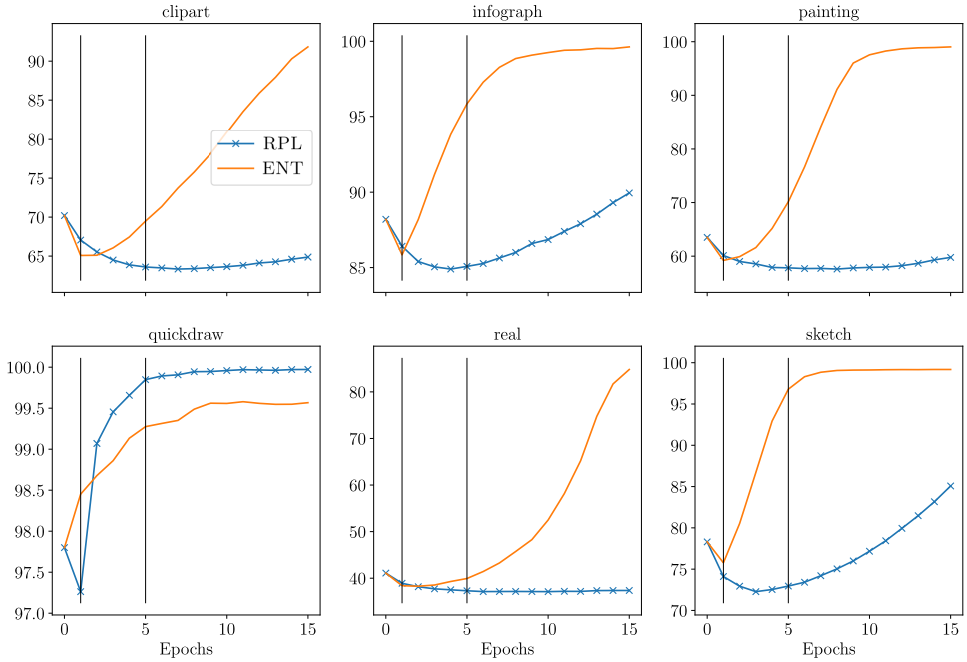


Figure 8: Top-1 error for the different IN-D domains for a ResNet50 and training with $\text{RPL}_{q=0.8}$ and ENT. We indicate the epochs at which we extract the test errors by the dashed black lines (epoch 1 for ENT and epoch 5 for $\text{RPL}_{q=0.8}$).

D.4 Detailed results for the EfficientNet-L2 Noisy Student model on IN-D

We show the detailed results for the EfficientNet-L2 Noisy Student model on IN-D in Table 38.

Table 38: Top-1 error (↘) on IN-D in % for EfficientNet-L2

Domain	Baseline	ENT	RPL
Clipart	45.0	39.8	**37.9**
Infograph	**77.9**	91.3	94.3
Painting	42.7	41.7	**40.9**
Quickdraw	**98.4**	99.4	99.4
Real	29.2	28.7	**27.9**
Sketch	56.4	**48.0**	51.5
mDE	67.2	**66.8**	67.2

D.5 Detailed results on the error analysis on IN-D

Analysing frequently predicted classes We analyze the most frequently predicted classes on IN-D by a vanilla ResNet50 and show the results in Fig. 9. The colors of the bars indicate whether the predicted class is part of the IN-D dataset: "blue" indicates that the class appear in the IN-D dataset, while "orange" means that the class is not present in IN-D.

We make several interesting observations: First, we find most errors interpretable: it makes sense that a ResNet50 assigns the label "comic book" to images from the "Clipart" or "Painting" domains, or "website" to images from the "Infograph" domain, or "envelope" to images from the "Sketch" domain. Second, on the hard domain "Quickdraw", the ResNet50 mostly predicts non-sensical classes that are not in IN-D, mirroring its almost chance performance on this domain. Third, we find no systematic errors on the "Real" domain which is expected since this domain should be similar to IN.

Analyzing the correlation between the performance on IN-C/IN-R and IN-D We show the Spearman's rank correlation coefficients for errors on ImageNet-D correlated to errors on ImageNet-R and ImageNet-C for robust ResNet50 models in Fig. 9. For this correlation analysis, we take the error numbers from Table 33. We find the correlation to be high between most domains in IN-D and IN-R which is expected since the distribution shift between IN-R and IN is similar to the distribution shift between IN-D and ImageNet. The only domain where the Spearman's rank correlation coefficient is higher for IN-C is the "Real" domain which can be explained with IN-C being closer to real-world data than IN-R. Thus, we find that the Spearman's rank correlation coefficient reflects the similarity between different datasets.

Filtering predictions on IN-D that cannot be mapped to ImageNet We perform a second analysis: We filter the predicted labels according to whether they can be mapped to IN-D and report the filtered top-1 errors as well as the percentage of filtered out inputs in Table 39. We note that for the domains "infograph" and "quickdraw", the ResNet50 predicts labels that cannot be mapped to IN-D in over 70% of all cases, highlighting the hardness of these two domains.

Filtering labels and predictions on IN that cannot be mapped to ImageNet-D To test for possible class-bias effects, we test the performance of a ResNet50 model on ImageNet classes that can be mapped to IN-D and report the results in Table 39.

First, we map IN labels to IN-D to make the setting as similar as possible to our experiments on IN-D and report the top-1 error (12.1%). This error is significantly lower compared to the top-1 error a ResNet50 obtains following the standard evaluation protocol (23.9%). This can be explained by the simplification of the task: While in IN there are 39 bird classes, these are all mapped to the same hierarchical class in IN-D. Therefore, the classes in IN-D are more dissimilar from each other than in IN. Additionally, there are only 164 IN-D classes compared to the 1000 IN classes, raising the chance level prediction.

Table 39: top-1 error on IN and different IN-D domains for different settings: left column: predicted labels that cannot be mapped to IN-D are filtered out, right column: percentage of filtered out labels.

Dataset	top-1 error on filtered labels in %	percentage of rejected inputs
IN val	13.4	52.7
IN-D real	17.2	27.6
IN-D clipart	59.0	59.0
IN-D infograph	59.3	74.6
IN-D painting	39.5	42.4
IN-D quickdraw	96.7	76.1
IN-D sketch	65.6	47.9

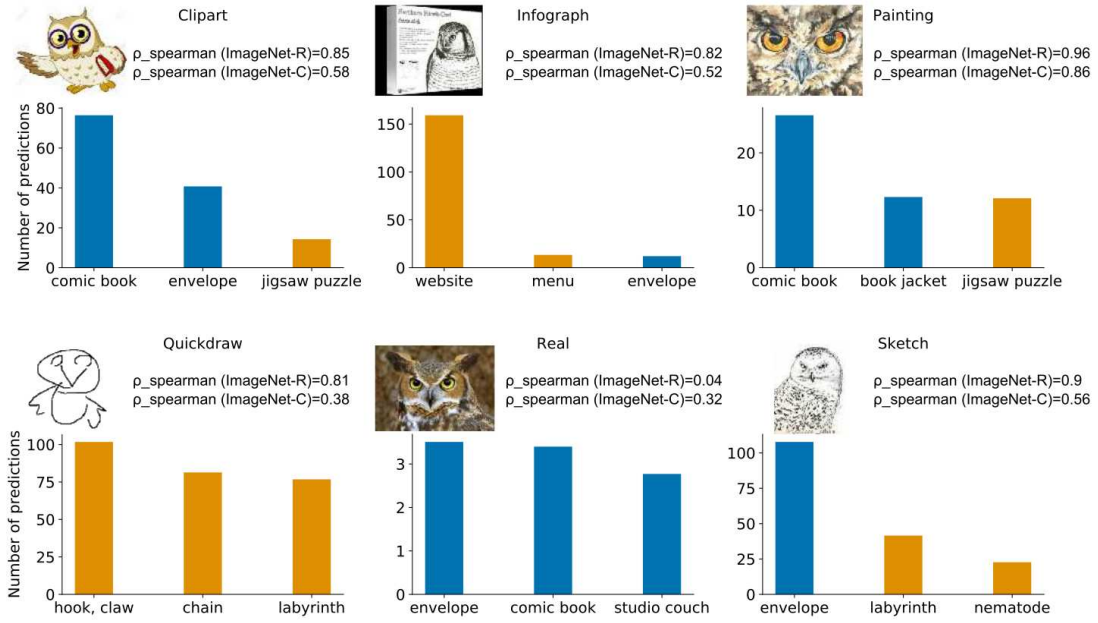


Figure 9: Systematic predictions of a vanilla ResNet50 on IN-D for different domains. The colors of the bars indicate whether the predicted class is part of the IN-D dataset: "blue" indicates that the class appear in the IN-D dataset, while "orange" means that the class is not present in IN-D. The errors on IN-R are strongly correlated to errors on IN-D for most domains, except for the "Real" domain.

If we further only accept predictions that can be mapped to IN-D, the top-1 error is slightly increased to 13.4%. In total, about 52.7% of all images in the IN validation set cannot be mapped to IN-D.

D.6 Top-1 error on IN-D for AlexNet

We report the top-1 error numbers on different IN-D as achieved by AlexNet in Table 40. We used these numbers for normalization when calculating mDE.

Table 40: top-1 error on IN-D by AlexNet which was used for normalization.

Dataset	top-1 error in %
IN-D real	54.887
IN-D clipart	84.010
IN-D infograph	95.072
IN-D painting	79.080
IN-D quickdraw	99.745
IN-D sketch	91.189

E Additional experiments

E.1 Beyond ImageNet classes: Self-learning on WILDS

The WILDS benchmark (Koh et al., 2021) is comprised of ten tasks to test domain generalization, subpopulation shift, and combinations thereof. In contrast to the setting considered here, many of the datasets in WILDS mix several 10s or 100s domains during test time.

The Camelyon17 dataset in WILDS contains histopathological images, with the labels being binary indicators of whether the central 32×32 region contains any tumor tissue; the domain identifies the hospital that the patch was taken from. Camelyon17 contains three different test splits with different domains and varying difficulty levels. For evaluation, we took the pretrained checkpoint from worksheets.codalab.org/worksheets/0x00d14c55993548a1823a710642f6d608 (camelyon17_erm_densenet121_seed0) for a DenseNet121 model (Huang et al., 2017) and verified the reported baseline performance numbers. We adapt the models using ENT or RPL for a maximum of 10 epochs using learning rates $\{3 \times 10^{-5}, 3 \times 10^{-4}, \dots 3 \times 10^{-1}\}$. The best hyperparameter is selected according to OOD Validation accuracy.

The RxRx1 dataset in WILDS contains RGB images of cells obtained by fluorescent microscopy, with the labels indicating which of the 1,139 genetic treatments (including no treatment) the cells received; the domain identifies the batch in which the imaging experiment was run. The RxRx1 dataset contains three test splits, however, unlike Camelyon17, in all of the splits the domains are mixed. For evaluation, we took the pretrained checkpoint from worksheets.codalab.org/bundles/0x7d33860545b64acca5047396d42c0ea0 for a ResNet50 model and verified the reported baseline performance numbers. We adapt the models using ENT or RPL for a maximum of 10 epochs using base learning rates $\{6.25 \times 10^{-6}, 6.25 \times 10^{-5}, \dots 6.25 \times 10^{-2}\}$, which are scaled to the admissible batch size for single GPU adaptation using linear scaling. The best hyperparameter is selected according to OOD Validation accuracy.

Table 41: Self-learning can improve performance on WILDS if a systematic shift is present—on Camelyon17, the ood validation and test sets are different hospitals, for example. On datasets like RxRx1 and FMoW, we do not see an improvement, most likely because the ood domains are shuffled, and a limited amount of images exist for each test domain.

	Top-1 accuracy [%]		
	Validation (ID)	Validation (OOD)	Test (OOD)
Camelyon17			
Baseline	81.4	88.7	63.1
BN adapt	97.8 (+16.4)	90.9 (+2.2)	88.0 (+24.9)
ENT	97.6 (+16.2)	92.7 (+4.0)	91.6 (+28.5)
RPL	97.6 (+16.2)	93.0 (+4.3)	91.0 (+27.9)
RxRx1			
Baseline	35.9	19.1	29.7
BN adapt	35.0 (-0.9)	19.1 (0.0)	29.4 (-0.3)
ENT	34.8 (-1.1)	19.2 (+0.1)	29.4 (-0.3)
RPL	34.8 (-1.1)	19.2 (+0.1)	29.4 (-0.3)
FMoW			
Baseline	60.5	59.2	52.9
BN adapt	59.9 (-0.6)	57.6 (-1.6)	51.8 (-1.1)
ENT	59.9 (-0.6)	58.5 (-0.7)	52.2 (-0.7)
RPL	59.8 (-0.7)	58.6 (-0.6)	52.1 (-0.8)

The FMoW dataset in WILDS contains RGB satellite images, with the labels being one of 62 building or land use categories; the domain specifies the year in which the image was taken and its geographical region (Africa, the Americas, Oceania, Asia, or Europe). The FMoW dataset contains four test splits for different time periods, for which all regions are mixed together. For evaluation, we took the pretrained checkpoint from //worksheets.codalab.org/bundles/0x20182ee424504e4a916fe88c91afd5a2 for a DenseNet121 model and verified the reported baseline performance numbers. We adapt the models using ENT or RPL for a maximum

of 10 epochs using learning rates $\{5.0 \times 10^{-6}, 5.0 \times 10^{-5}, \dots 5.0 \times 10^{-2}\}$. The best hyperparameter is selected according to OOD Validation accuracy.

While we see improvements on Camelyon17, neither BN adaptation nor self-learning can improve performance on RxRx1 or FMoW. Initial experiments on PovertyMap and iWildsCam also do not show improvements with self-learning. We hypothesize that the reason lies in the mixing of the domains: Both BN adaptation and our self-learning methods work best on systematic domain shifts. These results support our claim that self-learning is effective, while showing the important limitation when applied to more diverse shifts.

E.2 Small improvements on BigTransfer models with Group normalization layers

We evaluated BigTransfer models (Kolesnikov et al., 2020) provided by the timm library (Wightman, 2019). A difference to the ResNet50, ResNeXt101 and EfficientNet models is the use of group normalization layers, which might influence the optimal method for adaptation—for this evaluation, we followed our typical protocol as performed on ResNet50 models, and used affine adaptation.

For affine adaptation, a distilled BigTransfer ResNet50 model improves from 49.6 % to 48.4 % mCE on the ImageNet-C development set, and from 55.0 % to 54.4 % mCE on the ImageNet-C test set when using RPL ($q = 0.8$) for adaptation, at learning rate 7.5×10^{-4} at batch size 96 after a single adaptation epoch. Entropy minimization did not further improve results on the ImageNet-C test set. An ablation over learning rates and epochs on the dev set is shown in Table 42, the final results are summarized in Table 43.

Table 42: mCE in % on the IN-C dev set for ENT and RPL for different numbers of training epochs when adapting the affine batch norm parameters of a BigTransfer ResNet50 model.

criterion	ENT			RPL		
lr, $7.5 \times$ epoch	10^{-5}	10^{-4}	10^{-3}	10^{-5}	10^{-4}	10^{-3}
0	49.63	49.63	49.63	49.63	49.63	49.63
1	49.44	50.42	52.59	49.54	48.89	48.95
2	49.26	50.27	56.47	49.47	**48.35**	50.77
3	49.08	52.18	60.06	49.39	48.93	51.45
4	48.91	52.03	60.50	49.31	50.01	51.53
5	**48.80**	51.97	62.91	49.24	49.96	51.34
6	48.83	52.10	62.96	49.16	49.71	51.19
7	48.83	52.10	62.96	49.16	49.71	51.19

Table 43: mCE in % on the IN-C dev set for ENT and RPL for different numbers of training epochs when adapting the affine batch norm parameters of a BigTransfer ResNet50 model.

	dev mCE	test mCE
Baseline	49.63	55.03
ENT	48.80	56.36
RPL	**48.35**	**54.41**

E.3 Can Self-Learning improve over Self-Learning based UDA?

An interesting question is whether test-time adaptation with self-learning can improve upon self-learning based UDA methods. To investigate this question, we build upon French et al. (2018) and their released code base at github.com/Britefury/self-ensemble-visual-domain-adapt. We trained the Baseline models from scratch using the provided shell scripts with the default hyperparameters and verified the reported performance. For adaptation, we tested BN adaptation, ENT, RPL, as well as continuing to train in exactly the setup of French et al. (2018), but without the supervised loss. For the different losses, we adapt the models for a maximum of 10 epochs using learning rates $\{1 \times 10^{-5}, 1 \times 10^{-4}, \dots, 1 \times 10^{-1}\}$.

Note that for this experiment, in contrast to any other result in this paper, **we purposefully do not perform proper hyperparameter selection based on a validation dataset**—instead we report the best accuracy across all tested epochs and learning rates to give an upper bound on the achievable performance for test-time adaptation.

As highlighted in Table 44, none of the four tested variants is able to meaningfully improve over the baseline, corroborating our initial hypothesis that self-learning within a full UDA setting is the optimal strategy, if dataset size and compute permits. On the other hand, results like the teacher refinement step in DIRT-T

(Shu et al., 2018) show that with additional modifications in the loss function, it might be possible to improve over standard UDA with additional adaptation at test time.

Table 44: Test-time adaptation marginally improves over self-ensembling.

	Baseline	BN adapt	ENT	RPL	Self-ensembling loss
MNIST→SVHN					
MT+TF	33.88	34.44	34.87	35.09	33.27
MT+CT*	32.62	34.11	34.25	34.21	33.36
MT+CT+TF	41.59	41.93	41.95	41.95	42.70
MT+CT+TFA	30.55	32.53	32.54	32.55	30.84
SVHN-specific aug.	97.05	96.82	96.91	96.87	97.12
MNIST→USPS					
MT+TF	98.01	97.91	97.96	97.91	98.16
MT+CT*	88.34	88.39	88.54	88.39	88.44
MT+CT+TF	98.36	98.41	98.41	98.41	98.50
MT+CT+TFA	98.45	98.45	98.45	98.45	98.61
SVHN→MNIST					
MT+TF	98.49	98.47	98.49	98.47	99.40
MT+CT*	88.34	88.36	88.36	88.36	89.36
MT+CT+TF	99.51	99.49	99.5	99.49	99.57
MT+CT+TFA	99.56	99.57	99.57	99.57	99.58
SVHN-specific aug.	99.52	99.49	99.5	99.49	99.65
USPS→MNIST					
MT+TF	92.79	92.62	92.62	92.66	93.08
MT+CT*	99.11	99.13	99.14	99.13	99.21
MT+CT+TF	99.41	99.42	99.45	99.42	99.52
MT+CT+TFA	99.48	99.54	99.57	99.54	99.54

F Software stack

We use different open source software packages for our experiments, most notably Docker (Merkel, 2014), scipy and numpy (Virtanen et al., 2020), GNU parallel (Tange, 2011), Tensorflow (Abadi et al., 2016), PyTorch (Paszke et al., 2017), timm (Wightman, 2019), Self-ensembling for visual domain adaptation (French et al., 2018), the WILDS benchmark (Koh et al., 2021), and torchvision (Marcel & Rodriguez, 2010).