

Jagiellonian University

Doctoral School of Exact and Life Sciences
Faculty of Mathematics and Computer Science



Ph.D. Thesis

Adaptiveness in Deep Learning Models

mgr Maciej Wołczyk

Under the supervision of prof. dr hab. Jacek Tabor

Kraków, 2023

Dziękuję mojej rodzinie i mojej żonie Ewie za ciągłe wsparcie.

*Dziękuję również prof. Jackowi Taborowi za bezcenne rady
oraz moim współautorom za przyjemną i owocną współpracę.*

Bez nich ta praca by nie powstała.

Contents

List of publications	7
1. Introduction and Motivation	8
2. Part I: Adaptiveness in Continual Learning	12
2.1. Research Scope	12
2.2. Continual World: A Robotic Benchmark For Continual Reinforcement Learning	13
2.3. Disentangling Transfer in Continual Reinforcement Learning	15
2.4. Continual Learning with Guarantees via Weight Interval Constraints	17
3. Part II: Adaptation to New Paradigms	21
3.1. Research Scope	21
3.2. Zero Time Waste: Recycling Predictions in Early Exit Neural Networks	22
3.3. PluGeN: Multi-Label Conditional Generation From Pre-Trained Models	24
4. Other works & achievements	28
5. Conclusion	33
Bibliography	34
A. Full text of the Publications	41

Streszczenie

Modele głębokiego uczenia w ostatnich latach okazały się niezwykle biegłe w wielu domenach, takich jak widzenie komputerowe, przetwarzanie języka naturalnego i chemioinformatyka. Jednocześnie dany model głębokiego uczenia jest zwykle ograniczony do jednego konkretnego zastosowania i nie może zostać łatwo zaadaptowany do nowych warunków. Z drugiej strony, w świecie rzeczywistym zwierzęta i ludzie muszą stale dostosowywać się do nowych warunków, rozwiązywać sekwencje heterogenicznych zadań i kalibrować czas reakcji w zależności od sytuacji. Nieustanne zmiany charakterystyczne dla świata rzeczywistego są jedną z głównych przeszkód na drodze do budowy bardziej uniwersalnych modeli sztucznej inteligencji, które mogą być stosowane do różnorodnych problemów. W tej rozprawie badamy różne sposoby adaptacji wcześniej wytrenowanych sieci do nowych zadań i ustawień.

Niniejsza rozprawa składa się z pięciu prac i jest podzielona na dwie części. Pierwsza część rozprawy zawiera Publikacje [I-III]¹, które skupiają się na problemie ciągłego uczenia. Ten obszar badawczy zajmuje się budowaniem algorytmów zdolnych do trenowania na zmieniającym się rozkładzie danych przy efektywnym wykorzystaniu wiedzy z przeszłości do uczenia się nowych zadań. Jest to istotne, jako że w sieciach neuronowych występuje zjawisko katastroficznego zapominania, co oznacza, że w obliczu przesunięcia rozkładu danych model traci zdolność rozwiązywania wcześniej napotkanych zadań. Publikacja [I] wprowadza środowisko oraz protokół ewaluacji dla ciągłego uczenia ze wzmocnieniem, oparte na sekwencji zadań manipulacji robotycznej wraz z zestawem kluczowych metryk. Publikacja [II] rozszerza poprzednią pracę, skupiając się na zagadnieniu transferu wiedzy, tj. jak wykorzystać wiedzę z przeszłości, aby wydajnie uczyć się na nowych zadaniach. Identyfikujemy kluczowe aspekty architektoniczne oraz algorytmiczne transferu wiedzy oraz proponujemy zestaw wytycznych, które prowadzą do bardziej efektywnych rozwiązań. W Publikacji [III] proponujemy metodę ciągłego uczenia z gwarancjami na zapominanie, pokazując, że spadek wydajności może być ograniczony z góry przy użyciu arytmetyki interwałowej.

Druga część tej rozprawy skupia się na metodach adaptacji istniejących modeli do innych paradygmatów. W Publikacji [IV] pokazujemy, jak efektywnie przekształcić model, który wykonuje obliczenia z wykorzystaniem statycznej ścieżki obliczeniowej, tak aby dynamicznie adaptował ścieżkę obliczeniową do każdego przykładu. Dokonujemy tego, wprowadzając klasyfikatory wczesnego wyjścia, które umożliwiają zmianę czasu przetwarzania w zależności od wejścia. W Publikacji [V] wykorzystujemy technikę normalizujących przepływów do wprowadzenia warunkowania do wcześniej wytrenowanych nienadzorowanych modeli generatywnych, co pozwala na lepszą kontrolę procesu generowania.

¹ Informacje bibliograficzne dotyczące Publikacji zostały zamieszczone w kolejnej sekcji.

Wszystkie powyżej wymienione Publikacje zostały zaprezentowane na topowych konferencjach rangi A* i we wszystkich jestem pierwszym autorem. Powstały w wyniku współpracy zarówno polskiej, jak i międzynarodowej. Dodatkowo opublikowałem szereg prac dotyczących innych zagadnień głębokiego uczenia, brałem udział w stażach naukowych w renomowanych instytucjach, byłem recenzentem dla czasopism i konferencji najwyższej rangi, a także organizowałem warsztaty i szkoły letnie z zakresu głębokiego uczenia.

Słowa kluczowe: głębokie uczenie, adaptacja, uczenie ciągłe, uczenie ze wzmocnieniem

Abstract

Deep learning models in recent years have proven to be extremely proficient in many different domains such as computer vision, language processing, and cheminformatics. However, a deep learning model is usually restricted to a specific, stationary setting and cannot be trivially adapted to new conditions. On the contrary, animals and humans acting in the real world have to constantly readjust to new conditions, solve sequences of heterogeneous tasks and calibrate response times depending on the situation. The constant change specific to the real world is one of the major obstacles to building more generally applicable artificial intelligence agents that can be used for a wide variety of problems. In this thesis, we look into different ways of adapting previously trained networks to new tasks and settings.

This thesis consists of five papers and is divided into two parts. The first part of this thesis contains Publications [I-III]² which focus on the problem of continual learning. This research area deals with building algorithms capable of training on a shifting data distribution while reusing past knowledge to efficiently learn new tasks. It is an important problem, since artificial neural networks experience catastrophic forgetting, i.e. after a data shift the model loses its performance on the previous tasks. Publication [I] proposes a continual reinforcement learning benchmark built using a sequence of robotic manipulation tasks, along with a set of crucial metrics. Publication [II] extends the previous paper by focusing on the issue of transfer, i.e. how to reuse the knowledge from the past in order to efficiently learn on the new data. We identify key architectural and algorithmic aspects of transfer and propose a set of guidelines that lead to more effective approaches. In Publication [III], we propose a continual learning method with guarantees on forgetting, showing that performance drop can be bounded from above using interval arithmetic.

The second part of this thesis focuses on methods for adapting existing models to different settings and paradigms. In Publication [IV], we show how to efficiently adapt a deep learning model with a static computation path, so that it dynamically chooses the computation path depending on the example. We do this by introducing early exit classifiers, which allow us to control the processing time of each example. In Publication [V] we use normalizing flows to introduce conditioning factors to pre-trained unsupervised generative models, allowing for a more controlled generation process.

All the Publications listed here were presented at top A* conferences and I am the first author of all of them. They were written as a result of national as well as international cooperation. Additionally, I published several works on different topics in deep learning, interned at established research institutions, served as a reviewer for top-ranking journals and conferences, and organized workshops and summer schools on machine learning.

Keywords: deep learning, adaptation, continual learning, reinforcement learning

² The bibliographical information on the Publications is included in the next section.

List of publications

- [I] Wołczyk, Maciej, Michał Zając, Razvan Pascanu, Łukasz Kuciński, and Piotr Miłoś. “Continual World: A robotic benchmark for continual reinforcement learning.” Advances in Neural Information Processing Systems (NeurIPS), pp. 28496-28510, 2021, CORE A*, 200 MEiN points.
- [II] Wołczyk, Maciej, Michał Zając, Razvan Pascanu, Łukasz Kuciński, and Piotr Miłoś. “Disentangling Transfer in Continual Reinforcement Learning.” Advances in Neural Information Processing Systems (NeurIPS), pp. 6304-6317, 2022, CORE A*, 200 MEiN points.
- [III] Wołczyk, Maciej, Karol J. Piczak, Bartosz Wójcik, Łukasz Pustelnik, Paweł Morawiecki, Jacek Tabor, Tomasz Trzciński, and Przemysław Spurek. “Continual Learning with Guarantees via Weight Interval Constraints.” International Conference on Machine Learning (ICML), pp. 23897-23911, 2022, CORE A*, 200 MEiN points.
- [IV] Wołczyk, Maciej, Bartosz Wójcik, Klaudia Bałazy, Igor T. Podolak, Jacek Tabor, Marek Śmieja, and Tomasz Trzciński. “Zero Time Waste: Recycling Predictions in Early Exit Neural Networks.” Advances in Neural Information Processing Systems (NeurIPS), pp. 2516-2528, 2021, CORE A*, 200 MEiN points.
- [V] Wołczyk, Maciej, Magdalena Proszewska, Łukasz Maziarka, Maciej Zięba, Patryk Wielopolski, Rafał Kurczab, and Marek Śmieja. “PluGeN: Multi-Label Conditional Generation From Pre-Trained Models.” Proceedings of the AAAI Conference on Artificial Intelligence (AAAI), pp. 8647-8656. 2022, CORE A*, 200 MEiN points.

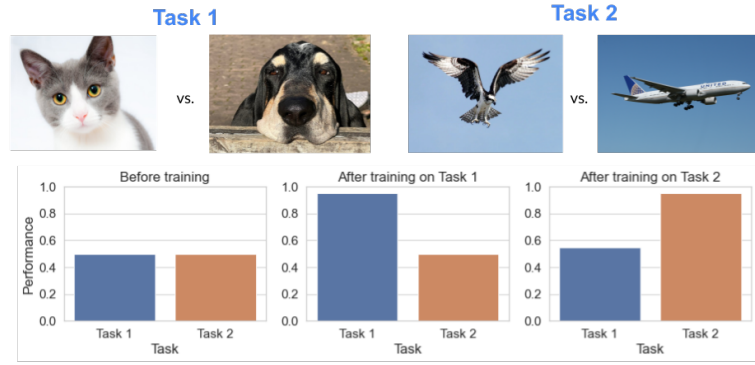
1. Introduction and Motivation

Recently, the world has seen remarkable achievements in the area of deep learning research. Over the last 15 years, the field has grown rapidly due to multiple factors such as the widespread availability of Graphics Processing Units (GPUs) capable of efficient parallelization [62, 75], finding well-scalable neural network architectures [34, 89], optimization schemes [46, 65], inductive biases for training large models [14, 41], and improvements in terms of data availability [13]. Today, deep learning models are capable of performing a plethora of tasks in natural language processing (NLP) [9, 67], computer vision (CV) [18, 34] and reinforcement learning (RL) [3, 61, 78], which were impossible two decades ago. In many of these areas, deep learning methods achieve superhuman performance.

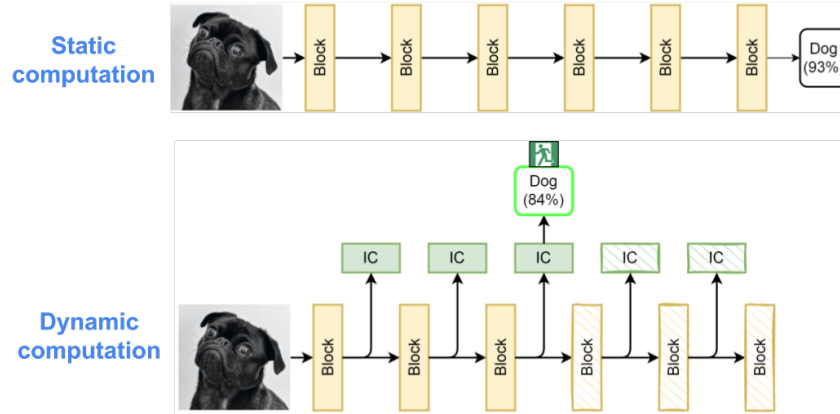
However, at the same time, these models do not reach the capabilities of human intelligence in many other areas, including generalization to out-of-distribution data [21, 36], robustness to simple transformations and adversarial attacks [2, 35, 70], and energy efficiency [59]. The focus of this thesis is the analysis of one such problem, where deep learning models are still significantly worse than humans, namely adaptation. Standard training regimes for deep learning algorithms are usually rigid in terms of assumptions about the data and the training process. Effectively, the deep learning paradigm requires that the datasets or environments used for learning must remain static throughout the training process, fulfilling the assumption that data are independent and identically distributed (i.i.d.). The overall setting used at inference must closely follow the conditions observed during the training [6, 32].

In practice, this means that most deep learning models are trained to do only a single specific task and work in only one particular setting, and adapting an existing model to new datasets, situations or goals is highly non-trivial. If a researcher or practitioner encounters a new scenario, a common solution is to start training completely from scratch, discarding previous models trained with different data and in different settings. This approach, although conceptually simple, is not efficient as it fails to take advantage of computational power and data used for the training of prior models. This is especially problematic as the cost of training has risen exponentially in recent years [5, 80].

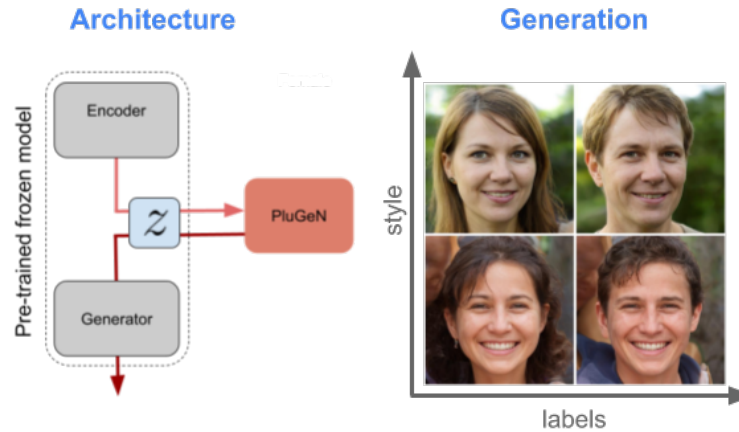
Recently, fine-tuning became a highly popular strategy for reusing large deep learning models [66, 83]. Fine-tuning is the practice of taking the parameters of a pre-trained model, usually one that was trained on a wide variety of data, and using them as an initialization for training on a specific downstream task that was not included in the original training set. Since



(a) Adaptation to data shift in continual learning.



(b) Adaptation to using dynamical computation paths.



(c) Adaptation to controlled generation.

Figure 1.1: Different types of adaptation discussed in this thesis. (a) Adaptation to a changing data stream in continual learning, here in the case of a two-class classification problem. Model, which achieved perfect performance on Task 1, will catastrophically forget this knowledge after training on Task 2. (b) Adaptation to a dynamic computational path. We add early exits to a model with a static computation path and train them. The adapted model can change the number of layers used for processing, depending on the input. (c) Adaptation to controlled generation. By disentangling the latent space of a pre-trained generative model, we are able to control features such as the gender of the person in the picture, which was not possible in the original model.

fine-tuning usually requires smaller datasets and fewer iterations than full training, it is orders of magnitude cheaper computationally. At the same time, it leads to great practical results [17] and the pre-trained model can be used for multiple downstream tasks, each training procedure starting from the same set of pre-trained parameters. Since parameters for many pre-trained models are publicly available, it allows researchers without access to large compute and data to build sophisticated deep learning models, thus pushing forward the democratization of this field. Recently, the term *foundation model* emerged, which describes especially large models that were trained on broad data and can be adapted to a variety of tasks [7].

However, the adaptiveness of this paradigm is still limited. The weights of the pre-trained model are assumed to be frozen throughout its lifetime and only fine-tuned to specific downstream tasks. The knowledge gained through training on downstream tasks is not distilled back to the pre-trained model as we use a separate copy of the network to initialize training of a downstream task. Additionally, one might want to update the pre-trained model as time goes by in order to train on newly acquired data. Finally, it might be useful to change certain characteristics of the problem such as the computational paradigm or the conditioning factors. Although fine-tuning can be used to adapt an existing model to different data distributions or tasks, the objectives mentioned above are currently very difficult to achieve due to the underlying limitations of deep learning.

Contributions The main hypothesis of this dissertation is that adaptation properties of deep learning models can be significantly improved beyond simple fine-tuning, leading to better performance in the light of a data distribution shift and other types of changes. We conduct studies in the areas such as reinforcement learning, continual learning, generative modeling, and computer vision, with the aim of finding better ways to adapt neural networks. Since adaptation is a highly complex problem with many facets and possible applications, we put emphasis on specific areas which were previously underexplored in the literature. Central issues discussed in this paper are shown in Figure 1.1. The thesis consists of five papers dedicated to different aspects of adaptivity, presented in two chapters. All of the presented papers were published at top machine learning venues, i.e. conferences with A* CORE ratings, and I am the first author in all of them. Additionally, all these conferences are within the top 45 English scientific venues in terms of h5-index, according to Google Scholar [27].

In Chapter 2, we focus on the problem of continual learning, i.e. learning from a changing stream of data [32]. As we outline in Section 2.1, an adaptive agent should be able to adjust to constant changes in input distribution, goals, and scenarios, and efficiently accumulate past knowledge throughout its lifetime. In practice, unconstrained deep learning models suffer from a phenomenon called *catastrophic forgetting* [22, 26], where the performance of the network on previously encountered tasks falls considerably. In Section 2.2, we highlight the importance of this problem in reinforcement learning and in particular in continuous control,

an area that incorporates robotic tasks such as object manipulation and locomotion. Here, we describe Publication [I], in which we created a comprehensive benchmark for testing and analyzing existing continual learning approaches in this context. In Section 2.3, we describe Publication [II], in which we proposed solutions for efficient knowledge accumulation and behavior transfer which significantly outperform standard continual learning methods.

In Section 2.4, we investigate another underexplored aspect of continual learning, namely algorithms with guarantees on forgetting. Since existing continual learning methods often minimize forgetting through soft regularization, they cannot guarantee that performance on past tasks will not fall below any arbitrary level. That makes them a poor choice for safety-critical problems. We describe Publication [III], which introduces the first method for general continual learning that offers guarantees, which we implement through interval arithmetic. In summary, the chapter serves as a study of adaptiveness in continual learning with a focus on reinforcement learning and guarantee constraints.

In Chapter 3, we discuss two other non-trivial cases where fine-tuning solutions are not sufficient, and advanced adaptation techniques are necessary. We start with describing the research scope of the problem in Section 3.1. In Section 3.2, we describe Publication [IV] which considers the problem of adapting the computational paradigm of a deep learning model. We start with a model with a static computational path, where each input sample is processed using the same sequence of operations and we adapt it to the conditional computation paradigm where the computational cost varies between examples. We implement this idea by introducing early exit classifiers and we focus on efficiently sharing knowledge between them. Next, in Section 3.3, we describe Publication [V] which considers the problem of changing the conditioning of existing generative models. Namely, starting from an unsupervised generative model we want to move on to multi-label conditional generation. We achieve this by disentangling the latent space of the original model using a normalizing flow.

In Chapter 4, I briefly discuss publications I worked on during my Ph.D. that do not have direct connections to adaptiveness and are not a part of the thesis, as well as my other contributions. Finally, Chapter 5 contains a short summary of the contributions presented in this thesis as well as directions for future work.

2. Part I: Adaptiveness in Continual Learning

2.1. Research Scope

In this chapter, we discuss adapting deep learning models to changing streams of data, i.e. the *continual learning* problem [55, 71, 86]. Standard training of deep learning models often relies on the i.i.d. assumption which effectively requires the distribution governing the dataset to be static and perfectly mixed throughout the training [32]. In particular, training an existing model on any data from a different distribution than one originally used for learning may prove problematic. As such, a common, although inefficient solution when encountering an adaptation task, is to disregard previously acquired knowledge and restart training from scratch. Continual learning aims to find better solutions, which is essential to the development of deep learning methods in current times when full retraining becomes prohibitively expensive.

In practice, the changes in data might come in many different forms, such as shifts in the input distributions, the appearance of new classes in classification problems, or modification of the reward function in reinforcement learning. Although the space of possible changes might be continuous, for simplicity one usually assumes a discrete (even though possibly infinite) number of varying parts of the data stream, often called tasks or batches [12, 64].

One of the key issues in continual learning is *catastrophic forgetting* [22], a rapid deterioration of performance on the old data once new data is introduced. Since the model once knew how to solve previous tasks and then loses that ability, the *forgetting* term is used. Although forgetting might be easily alleviated by limiting the capability of the network to adapt (e.g. by freezing all the parameters in the network), doing so will reduce or even completely remove the ability of the network to solve new problems. Essentially, we want to be able to reuse the knowledge we already have in order to learn new tasks. The tension between alleviating catastrophic forgetting and increasing forward transfer, often described as the *plasticity-stability dilemma* [60, 64], is at the heart of continual learning.

Continual learning deals with different kinds of changes happening in the real world and as such, it is of critical importance from the perspective of adaptiveness of deep learning models. This thesis focuses on two specific subproblems of continual learning: (1) continual reinforcement learning and (2) providing guarantees for forgetting in continual learning. These will be discussed in the following sections.

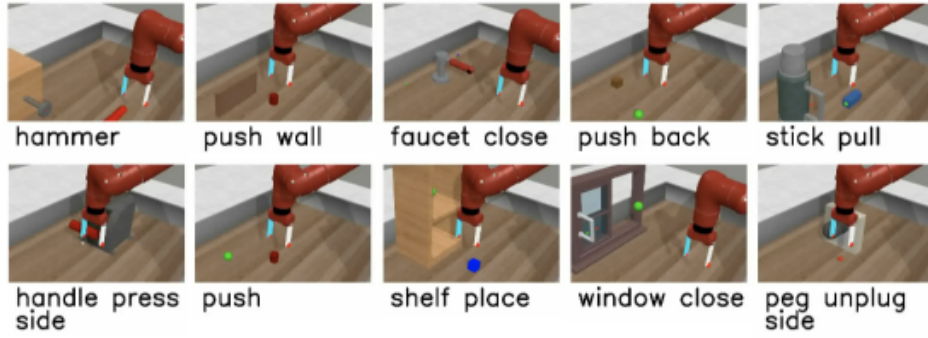


Figure 2.1: The CW10 sequence from the Continual World benchmark. The goal is to efficiently learn how to perform each task in the proposed sequence without forgetting the previous ones. While CW10 consists of ten robotic tasks, we also use another sequence, CW20, which is CW10 repeated twice. This repetition allows us to check the ability of the agent to revisit previously seen tasks.

2.2. Continual World: A Robotic Benchmark For Continual Reinforcement Learning

This section covers results from Publication [I] which investigates the problem of continual reinforcement learning, especially from the perspective of efficient knowledge transfer.

Motivation In literature, continual learning is usually discussed from the perspective of supervised learning, in particular tasks such as image classification, regression, and semantic segmentation [19, 54, 86]. There are relatively fewer works investigating the continual learning problem in the context of reinforcement learning [45], a paradigm dedicated to learning how to interact with an environment to maximize rewards [81]. In works discussed here, we argue that continual learning is crucial from the perspective of reinforcement learning problems and especially robotics. Although for non-embodied artificial intelligence agents, some changes in the world can be abstracted away, robots with physical presence have to deal with a plethora of problems such as wear and tear, evolving weather conditions, and changing locations. Going forward, it will be necessary to develop adaptive agents in order to deal with these data shifts. The continual learning setup serves as a tool to study the non-stationarity inherent to reinforcement learning in a controlled manner. The following works form a crucial investigation into this emerging issue.

Contents Publication [I] lays the groundwork for the analysis and development of continual reinforcement learning by introducing a robotic benchmark for this setting. The paper fills in an important existing gap in the continual reinforcement learning literature by proposing a benchmark that is computationally cheap, easily interpretable, but at the same time complex and extensible. To this end, we proposed a set of robotic tasks for continual learning by carefully

curating a sequence of environments with interesting inter-task dynamics. Each of these tasks has a clear connection to a real-world robotic manipulation problem and an easily interpretable success condition that allowed us to build useful and practical metrics. Figure 2.1 introduces the CW10 sequence we created to capture interesting phenomena in continual reinforcement learning.

The metrics we defined capture various important desiderata occurring in continual learning. Besides analyzing the average success rate and catastrophic forgetting, we highlighted the role of forward transfer, i.e. how well we can reuse the knowledge from previous tasks to efficiently train on new tasks. Subsequently, we implemented and ran several standard continual learning methods on our benchmark, including Elastic Weight Consolidation (EWC) [48], PackNet [56] and experience replay [11,69]. This required careful consideration, as some of these approaches were not created with reinforcement learning in mind.

With the testing ground established, we ran over 45,000 experiments checking the performance of implemented methods on various sequences of tasks, conditions, and hyperparameter configurations. Through this extensive empirical analysis, we discovered numerous critical issues in the continual reinforcement learning scenario that were not present in supervised learning. Strikingly, we noticed that experience replay methods which rely on rehearsing past data and perform very well on supervised benchmarks [12, 86] achieved poor results in our setting, even without imposing any constraints on memory. This unexpected result clearly shows that continual reinforcement learning necessarily has to take into consideration the inherent instability of reinforcement learning [81]. Quite surprisingly, we also found that the PackNet method is able to outperform the multi-task solution which is usually considered to be the upper bound in terms of performance on other benchmarks [12], further highlighting the gap between reinforcement learning and supervised learning. As such, the Continual World benchmark captures important phenomena that are not observed in well-explored tasks such as classification or semantic segmentation.

Additionally, we analyzed the problem of forward transfer. While the majority of continual learning benchmarks in the past have focused on catastrophic forgetting, we showed that in Continual World forgetting is mostly mitigated by several tested methods. On the other hand, the problem of knowledge reuse is highly non-trivial, as forward transfer might fail to appear in some seemingly easy cases such as revisiting already-seen tasks. Finally, to ensure that our conclusions are sound and well-supported empirically, we conduct numerous ablation studies, perform careful hyperparameter optimization, and analyze the statistical significance.

Summary Publication [I] introduces a long-needed benchmark dedicated to adaptivity in continual reinforcement learning and offers one of the most detailed empirical studies of this problem currently available in the literature. We proposed a complex environment, that at the same time is accessible in terms of computational costs. We implemented and tested numerous

methods from continual learning literature, identified their major strengths and weaknesses, and analyzed in a controlled setting what are the major drawbacks that have to be fixed. As a result, we identified major unexpected research problems relating to knowledge transfer, exploration, and regularization in continual reinforcement learning.

Contributions & Impact I am the first author of this paper and the primary contributor along with Michał Zając. I implemented, ran experiments, and analyzed the performance of regularization and experience replay-based continual learning methods. I proposed the forward transfer metric and used it as a tool for measuring adaptiveness in the paper. I performed studies on short sequences of tasks. Moreover, I participated in the conceptual development of the benchmark, and its implementation, as well as preparing the manuscript. I estimate my contribution to be 40%.

This work was presented at an A* conference NeurIPS 2021 and was well-received by the community. As of May 2023, it was cited 32 times. Several researchers tested their proposed methods on Continual World [4, 10, 24, 76], showcasing the usefulness of the proposed benchmark. Additionally, some papers repurposed the metrics we proposed in this work. For example, the forward transfer metric which measures the relative efficiency of learning was used in a continual supervised learning benchmark NEVIS’22 [8].

2.3. Disentangling Transfer in Continual Reinforcement Learning

This section covers Publication [II], in which we investigated further and proposed solutions to some of the problems we identified in Publication [I].

Motivation In this work, we focused on the issue of transfer, i.e. how to efficiently reuse past knowledge to quickly learn new tasks in the continual reinforcement learning setting. This problem is crucial, since training new models becomes more expensive, especially in fields that interact with the real world, such as robotics. In principle, we can significantly reduce these costs by efficiently reusing what we already learned. However, as we showed in Publication [I], existing continual learning methods are not efficient in terms of knowledge transfer. As such, we examine what rules govern the transfer in reinforcement learning with the goal of proposing a better approach.

Contents In order to better understand the properties of transfer, we examined how different components of the Soft Actor-Critic (SAC) algorithm [30], a popular reinforcement learning method [23, 31, 52, 68], contribute to the overall knowledge transfer. We set up an extensive empirical testbed by using all possible combinations of pairs of tasks from Continual World as well as longer sequences such as CW10 and CW20. We isolated the effect of transferring

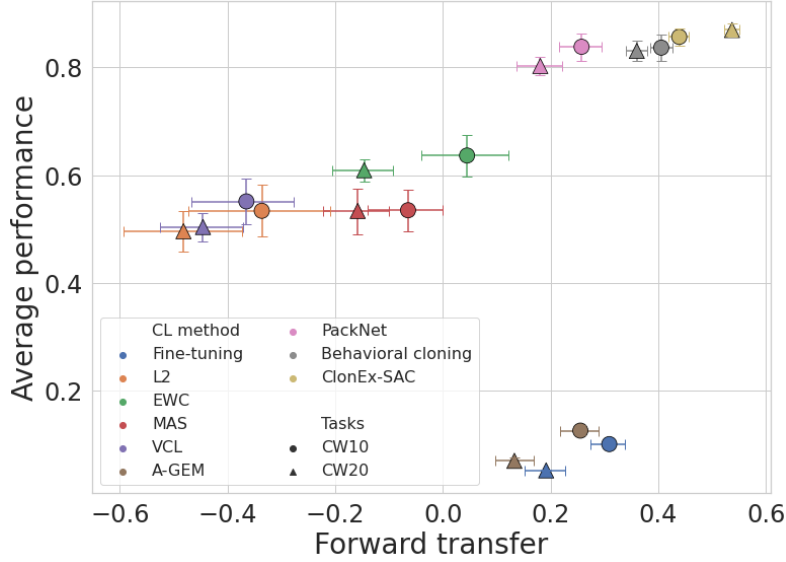


Figure 2.2: Comparison of the ClonEx-SAC approach proposed in Publication [II] to algorithms tested in Continual World (Publication [I]), evaluated on the CW10 and CW20 test sequences. The proposed method outperforms prior approaches, especially with respect to the forward transfer metric.

different SAC components such as the parameters of the actor, parameters of the critic, exploration policy, and the replay buffer.

We find that the critic is the most important contributor in terms of transfer since having good representations for estimating the value of each state simplifies the training process. On the other hand, the impact of the other components is also significantly positive and additive. That is, the impact of the actor and the exploration policy is still visible even when we transfer the critic, suggesting that these components accelerate learning in different ways. This hints at interesting underlying training dynamics and opportunities to leverage this synergy through efficient knowledge transfer in different components.

Following this idea, we tried to leverage past data to improve the transfer properties. We proposed and tested different strategies for incorporating past data in the replay buffer. Surprisingly, we found that behavioral cloning is a much less disruptive technique of transferring data than the more common experience replay. Subsequently, we examined the impact of different exploration strategies to understand how to pick the initial policy for gathering data in the current task. Finally, we analyzed the impact of regularization schemes and layer freezing on transfer.

These studies increased our understanding of the transfer of knowledge in reinforcement learning and in turn enabled us to compile a set of recommended design choices for setting up the Soft Actor-Critic algorithm in continual learning environments. We combined the proposed improvements and introduced an approach dubbed ClonEx-SAC which outperforms all methods previously tested in Publication [I]. The performance of ClonEx-SAC on the

Continual World benchmark is summarized in Figure 2.2, showing that by carefully studying the underlying problem, we can find a simple approach that performs better than more sophisticated approaches.

Summary In this work, we carefully investigated the properties of forward transfer, both in a simple two-task setting as well as in a continual learning scenario with longer sequences and long-term dependencies between tasks. We analyzed how different components of the actor-critic architecture affect the knowledge transfer, and looked through the lens of transfer at design choices related to exploration, populating the replay buffer, and regularizing the model. Based on these insights, we built ClonEx-SAC, an approach that performs better than out-of-the-box solutions from continual supervised learning. As such, this work contains contributions in the areas of continual learning, transfer learning, and reinforcement learning.

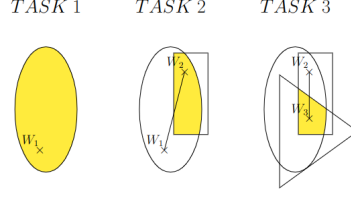
Contributions & Impact I am the first author of this paper and the primary contributor along with Michał Zając. I stated the research problem of forward transfer analysis and proposed the transfer matrices as the main tool of the study of adaptiveness. I conducted experiments in the two-task setting, compiled and analyzed results, and drew conclusions about the impact of algorithmic components. I proposed and implemented the behavioral cloning solution for efficient data reuse. I participated in preparing the manuscript. I estimate my contribution to be 40%. The paper was presented at an A* conference NeurIPS 2022 in December 2022. As of May 2023, it has 4 citations. In future work, we aim to further investigate the question of transfer, trying to pinpoint its sources and ways to amplify it.

2.4. Continual Learning with Guarantees via Weight Interval Constraints

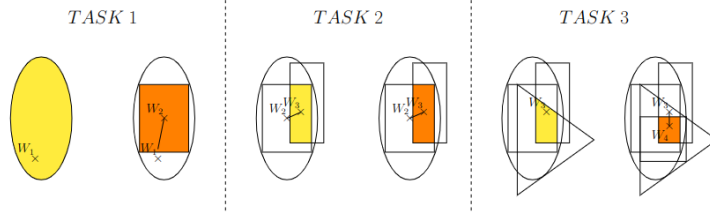
In this section, we describe Publication [III] which investigates the problem of providing guarantees in continual learning.

Motivation Although eliminating catastrophic forgetting is one of the most discussed aspects of continual learning, the majority of proposed solutions only apply soft measures without putting hard constraints on forgetting [12, 48, 91]. This approach is troublesome in real-life safety-critical situations where we want to be certain that the performance on previous tasks will stay above a given level. Modularization-based continual learning methods such as PackNet [56] or Progressive Networks [73] can guarantee zero forgetting through completely freezing parts of the network, but they require the task identity at inference to work properly, vastly limiting their usefulness in real life scenarios. As such, a more general solution is needed.

Contents In Publication [III], we take a step towards solving this problem as we propose a method for continual learning which guarantees that performance on the previous tasks will not



(a) General continual learning setting.



(b) InterContiNet continual learning setting.

Figure 2.3: Different perspectives on the continual learning problem, as presented in Publication [III]. (a) In general, continual learning can be thought of as finding the solution for the new task in a region of parameter space that contains parameters performing well on all of the previous tasks. (b) In Publication [III], we implement this idea by representing regions of the parameter space as hyperrectangles. Namely, we try to find the hyperrectangle with the highest volume that still fits within the region of well-performing parameters for the current task. In subsequent tasks, we constrain the optimization process to stay within this region.

fall below a given threshold (i.e. we set hard constraints on forgetting) and can be applied to settings where task identity is not available at inference.

We use a previously proposed formulation of continual learning which rephrases it as a problem of finding intersecting regions of the parameter space that perform well for particular tasks [50]. Effectively, continual learning requires the learner to find a solution for the current task within the region of viable parameters for the previous tasks. At the same time, this problem was proved to be NP-hard since the regions of the parameter space might be arbitrarily complex [50]. In Publication [III], we propose a more feasible instantiation of this problem, where we represent viable regions of the parameter space as hyperrectangles, reducing the complexity of the problem. This idea is sketched out in Figure 2.3.

We implement this idea by introducing an interval formulation of a neural network [28] and use it to build a nested structure of tasks. Rather than considering the value of each parameter to be a point in space, we instead conceptualize it as an interval. As such the parameters of the network form a Cartesian product of intervals, i.e. a hyperrectangle in the parameter space. With this formulation, we found that for each input we can easily compute the set of predictions that can be obtained by choosing arbitrary parameters from this hyperrectangle. While the naive solution would require sweeping over a continuous region of space and as such is infeasible,

we can instead use interval arithmetic [37] to compute it. With this approach, we optimize the model to find a region of parameters that perform well on a given task.

We apply this idea to the continual learning setting. When training on a sequence of tasks, we enforce that the hyperrectangle representing good solutions for the current task must be a subset of the parameter region representing the solutions for the previous tasks. We use interval arithmetic to compute the worst possible performance over the hyperrectangle. Since the parameters will not leave the hyperrectangle for the rest of the training, this value will be a lower bound on the worst-case performance, as we show in the paper. Being able to guarantee that the performance of the network will not fall below this level is an important property in settings where mistakes due to forgetting are especially harmful, such as autonomous driving and medical applications. To the best of our knowledge, the proposed framework is the first approach to achieve such guarantees in a general continual learning setting.

We implemented this model, dubbed InterContiNet, in practice on neural networks consisting of linear and convolutional layers and ran experiments on standard supervised continual learning benchmarks such as Split-MNIST and Split-CIFAR. We discovered that our method achieves similar performance as standard regularization methods such as Elastic Weight Consolidation [48] and Synaptic Intelligence [91] which do not put hard constraints on forgetting. As such, InterContiNet matched the performance of previous methods while at the same time being applicable to safety-critical settings. Finally, we conducted ablation studies to understand the optimization dynamics and, in particular, how the volume of the hyperrectangles changes throughout the training. We find that by tweaking hyperparameters one can reliably balance the plasticity and stability of InterContiNet.

Summary In this work, we proposed the first method for general continual learning that puts hard constraints on forgetting. We parametrize the weights of a neural network as a hyperrectangle, and we use interval arithmetic to find the worst-case accuracy on the current task over the whole region. By staying in that region, we can assure that the accuracy will not get lower than the computed threshold as we continue training on subsequent tasks. We showed empirically that this approach performs similarly or better than previous approaches in this family, while at the same time giving us guarantees on forgetting.

Contributions & Impact I am the first author of this paper and the top contributor along with Karol Piczak. I stated the research problem and performed a review of related literature. I proposed the theoretical concept of hyperrectangle training and performed a theoretical analysis. I took part in implementing the practical method and analyzing the results. I conducted ablation studies and analysis of the proposed method. I took a major part in preparing the manuscript. I estimate my contribution to be 30%. The work was presented at the ICML 2022 conference in June 2022. Currently, as of May 2023, there is 1 citation in a survey on continual learning [90].

In future work, we would like to explore different types of parameterization of the neural network, giving the model greater expressivity. Additionally, we want to investigate how to combine this approach with methods for meta-learning such as MAML [20].

3. Part II: Adaptation to New Paradigms

3.1. Research Scope

The adaptiveness problem in deep learning models is not restricted to adapting to shifts in data, as in continual learning. It also extends to using models outside of the paradigms they were created for. Although in some cases fine-tuning approaches can be used with great success, there are numerous situations where more sophisticated methods are required. In this chapter, we will focus on two particularly difficult types of adaptation: (1) adaptation to a different computational paradigm, and (2) adaptation of an unconditioned generative model to multi-label conditional generation.

In Section 3.2, we investigate the problem of adapting existing models to a different computational paradigm. Deep learning models usually work with a static computational graph. That is, independent of the input, the same series of arithmetic operations are performed. This approach is simple and easy to implement, especially when using hardware accelerators such as GPUs or TPUs. However, at the same time, it is inherently inefficient. In particular, different training examples might require vastly different amounts of resources to process them. Samples that are very difficult should use all the available computational resources, while easy samples should be processed quickly. In conditional computation approaches the model adapts the computational path to particular inputs, allowing in certain cases for per-example adjustment of the computational costs. Although such approaches, which include mixture-of-experts [72, 77, 92] and early exits [44, 84], are getting more common, most of these works assume that the network is trained from scratch in a conditional computation manner. In contrast, Section 3.2 focuses on a setting where we start with a pre-trained model with a static computation graph and turn it into a model with a dynamic computation graph. We discuss Publication [IV] which proposes such a method.

In Section 3.3, we describe the problem of adapting unsupervised generative models for conditional generation. Generative models are powerful tools for modeling the target distribution and sampling new data points from it. In particular, in recent years, multiple works showed how to generate images [43, 47, 87], audio tracks [63, 88] or even videos [39, 85]. However, in most of these cases, it is difficult to control the properties of the samples one wants to obtain. For example, ProgGAN [42] can be used to generate an arbitrary face, but cannot be

controlled to generate a face with any specific attributes or modify existing images in a reliable way. Even models that allow for better control are limited, as the type of conditioning must be decided upon during the training. For example, if we modified the training process of ProgGAN to condition the model to generate faces of people of specific gender and age, but we did not condition it on the face expression, we will not be able to control this property at inference. In Publication [V], we propose a method that aims to solve this problem by introducing a simple plugin model for adapting pre-trained unconditioned models to the task of conditional generation, thus giving the user control over the generative process.

As such, in this chapter, we aim to explore the problem of adapting existing pre-trained models to settings and situations where standard adaptation approaches are not viable. In particular, we study (1) adaptation to conditional computation and (2) adaptation to multi-label conditional generation. In both cases, we do so by introducing additional modules without affecting the underlying model.

3.2. Zero Time Waste: Recycling Predictions in Early Exit Neural Networks

In this section, we describe Publication [IV], which proposes a method for adapting a pre-trained model for conditional computation.

Motivation Early-exit approaches are a popular family of methods for conditional computation, i.e. they change the computational path depending on the example [33]. In classification tasks, standard neural networks usually only have a single classifier placed at the end of the network and it is activated after sequentially processing the input through all layers. However, early exit methods introduce additional classifiers, so-called *internal classifiers* (ICs) after selected hidden layers of the neural network [44]. This way, when processing an example, we might stop at each internal classifier and decide whether to answer based on its predictions. If we decide to answer early, we save some computational costs since the example is not processed by the remaining layers in the network.

Contents In Publication [IV], we focus on the problem of wastefulness in early exit methods in the context of adapting existing pre-trained models for this task, where smart reuse of available resources is especially important. In this scenario, one has to carefully use features obtained by the network, as it was not specifically trained with this application in mind. In particular, we investigate the question of reusing information between internal classifiers. We start by pointing out an important drawback of the early exit approach. If a given internal classifier does not return a prediction that is confident enough to be used as the final answer, its predictions will be discarded and the computation required to run this branch will be effectively wasted.

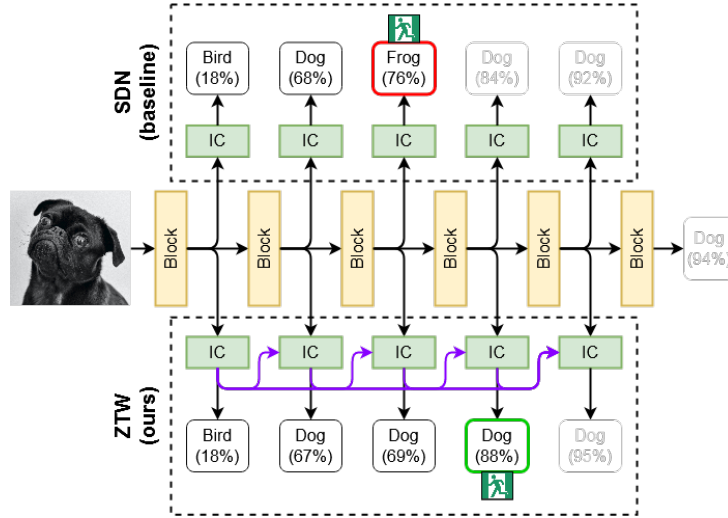


Figure 3.1: The overview of the Zero Time Waste approach from Publication [IV]. Standard early exit methods, shown on the top, do not transfer knowledge between internal classifiers. As such, they might disregard correct but unconfident predictions of previous classifiers. In Publication [IV], we propose the Zero Time Waste method shown at the bottom, which solves this issue by introducing mechanisms to reuse information from the past.

To understand the severity of this problem, we introduced the Hindsight Improvability metric which captures how many mistakes we could avoid if we efficiently reused predictions of the previous internal classifiers. We discovered that the potential gains of knowledge reuse are substantial and with a theoretical perfect agent, we would be able to eliminate as much as 70% of the total errors of the final classifier.

In order to reduce knowledge waste and take advantage of potential improvements, we introduced two techniques to reuse the computation from prior ICs. First, we proposed cascading, i.e. we add residual connections between subsequent internal classifiers which allows for direct information reuse. Second, we introduce ensembling, i.e. the predictions of all the previous ICs are aggregated through a weighted geometric average. When combined, these two approaches work together very well, covering different aspects of the information loss problem. They form a method we call Zero Time Waste (ZTW), as it minimizes the computational waste in early exit approaches. The overview of this approach is shown in Figure 3.1.

In Publication [IV], we show that this approach works very well on standard early exit benchmarks when compared to prior works. We investigate several standard computer vision datasets and architectures and show that the improvements offered by ZTW are significant in all of them. Additionally, we run experiments in a transfer learning paradigm and we perform thorough ablation studies, also showing great improvements. Finally, we show that ZTW can be applied to reinforcement learning, which to the best of our knowledge was the first application of early exit methods in that setting.

Summary In this work, we show that early exit models adapted from architectures with a static computational path are wasteful. Namely, the shallow early exit classifiers often predict correctly but without enough confidence, and as such their contribution is discarded, as we show using the Hindsight Improvability metric. We propose the Zero Time Waste method which combines cascading connections and ensembling of early exits to combat this inefficiency. The proposed approach outperforms prior work in this area, and it can be used with success in reinforcement learning scenarios. As such, this is a significant achievement in the problem of adaptivity in conditional computation.

Contributions & Impact I am the first author of this paper and a top contributor alongside Bartosz Wójcik. I stated the research problem, reviewed the relevant literature, and identified and examined the inefficiency problem. I proposed the ensembling component of the Zero Time Waste method, implemented it, ran experiments, and analyzed the design choices. I conceptualized and implemented the Hindsight Improvability metric and performed quantitative analysis, failure case analysis, and ablation studies of the proposed method. Additionally, I took a major part in preparing the manuscript. I estimate my contribution to be 40%. The paper was presented at an A* conference NeurIPS 2021 conference in December 2021. It was cited 12 times as of May 2023, and in many of these cases its influence was significant. For example, Laufer-Goldshtein et al. improved their approach using a cascading-like mechanism inspired by ZTW [53], Hou et al. [40] use it as a baseline in their experiments, and Matsubara et al. [57] states in their survey that ZTW "improves tradeoff between accuracy and inference time comparing to other early exit methods". We wrote an extension of this work and submitted it to IEEE Transactions on Pattern Analysis and Machine Intelligence journal. In future work, we aim to investigate adaptiveness with other conditional computation methods such as mixture-of-experts [72].

3.3. PluGeN: Multi-Label Conditional Generation From Pre-Trained Models

In this section, we describe Publication [V] which investigates the problem of controlling the generation process in unsupervised generative models.

Motivation Deep generative models are another great recent achievement in the field of deep learning. They are capable of modeling complex, high-dimensional distributions in domains such as computer vision [43, 87], natural language processing [9, 67] or 3D point clouds [29]. However, the capability of controlling the generative process is usually fairly limited. For example, most Generative Adversarial Networks [25] in vision are capable of generating an arbitrary image, but not an image with any specific features or attributes. Similarly, generative

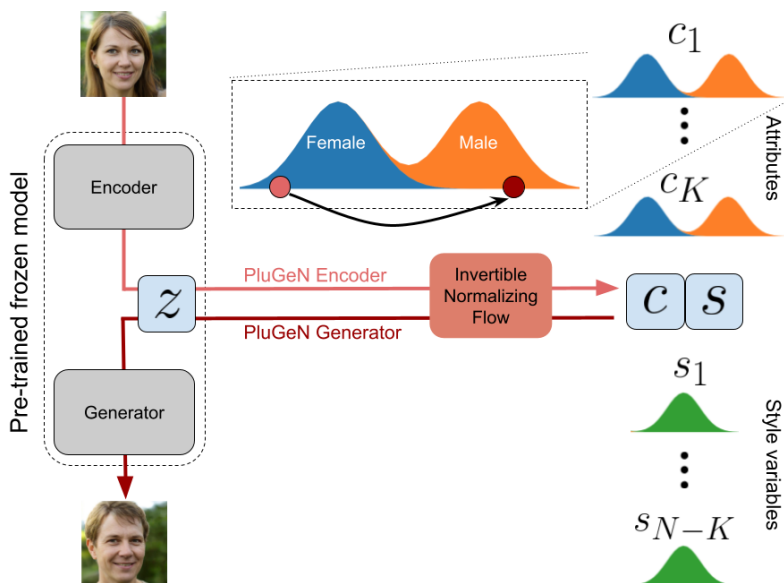


Figure 3.2: The scheme of the PluGeN method from Publication [V]. The left side of the figure represents the backbone pre-trained generative model. We transform the z latent space of this model using an invertible normalizing flow to get into the disentangled (c, s) -space. In this space, each attribute has an independent dimension that models its possible values with a Gaussian mixture. By changing values at these dimensions and then going through the inverse of the flow model, one can modify existing examples or generate new ones with desired features.

models for molecules can only generate an arbitrary molecule and it might not be possible to directly condition the model on a specific desirable trait. This is problematic in certain settings, as one might be interested in generating examples of a specific kind, e.g. molecules with given solubility. The naive solution would be to retrain the model from scratch while introducing these conditioning factors. However, this approach might be complex conceptually and expensive in terms of the required computation. Here, we search for more efficient alternatives.

Contents In Publication [V], we investigate the problem of controlling the generative process. For that purpose, we introduce PluGeN (Plugin Generative Network), a method for enabling conditional generation in unsupervised generative models. We operate on the latent space of such models which is usually by design unstructured with respect to the attributes, i.e. the training process does not force the latent directions to correspond to examples with given attributes. Although some form of disentanglement might occasionally happen and meaningful directions might sometimes emerge [1, 38], it is difficult to reliably use this property. As such, the tools for controlling the generative process are limited, making them difficult to use in many real-life scenarios.

To overcome this issue, we propose an invertible operation that allows us to go from the unstructured latent space of the original model to a specially constructed disentangled latent space. In the disentangled latent space, each attribute corresponds to a particular dimension, and as such the modification and sampling of specific attributes is very easy. To achieve

disentanglement, we use a normalizing flow [15, 16], a generative model that is capable of directly modeling the data through an invertible operation with a Jacobian matrix whose determinant is easy to compute. Thanks to these properties, it can be efficiently trained with a simple objective that maximizes the log-likelihood of the data. The simplicity of the optimization process allows us to apply it to existing models quickly and efficiently.

To achieve better control of the latent space, we use a novel parameterization of the latent distribution. We model each attribute as a mixture of Gaussians, with each Gaussian component corresponding to a different value of the attribute. For example, in the most common case of binary attributes, we use a mixture with two components. During the training, the pre-trained model is not modified in any way and we only train parameters of the normalizing flow in PluGeN, which allows for cheap and fast optimization. Since the normalizing flow is invertible, it is possible to freely move between these two latent spaces. We can sample or modify examples in the structured space and then go back to the unsupervised latent space to process them with the backbone generative model. This allows us to freely control the generative process which was previously heavily constrained and adapt it to new objectives or data types with a quick and cheap training process. The overview of the model is shown in Figure 3.2.

We implement PluGeN with three different backbone models, showcasing its usefulness in computer vision and cheminformatics, including the challenging task of molecule generation. We run experiments that demonstrate the high capabilities of our approach in tasks such as conditional generation of new samples, modification of existing examples, and interpolation between different values of attributes. Additionally, we show that the proposed formulation of the structure of the latent space leads to great performance in difficult generalization tasks such as extrapolation outside of attribute range and modeling rarely seen combinations of attributes.

Summary This work proposes a way to efficiently adapt and control the generative process of existing models. Most of the literature in generative modeling focuses on unsupervised cases, where one cannot control the properties of the generated sample. We use normalizing flows to perform a supervised disentanglement of the latent space of a pre-trained generative model. Then, by changing the latent codes in the disentangled space, we can modify generated samples and generate new ones with desired properties. We show that our method, PluGeN, performs very well at these tasks, as well as on extrapolation and generation of samples with uncommon attribute combinations. As such it is a significant contribution to the field of generative modelling.

Contributions & Impact I am the first author of this paper and a top contributor alongside Magdalena Proszewska. I stated the research problem and performed a literature review. I participated in the conceptualization and implementation of the Plugen model. I ran experiments with the VAE backbone and analyzed the results. I ran ablation studies and the

analysis of the proposed algorithm. Moreover, I participated in preparing the manuscript. I estimate my contribution to be 30%. This work was presented at an A* conference AAAI 2022 with a 15% acceptance rate in February 2022. Since then, we wrote an extension of this article, covering applications in problems such as 3D modeling and cases with a limited number of labels. We submitted the extension to the renowned IEEE Transactions on Pattern Analysis and Machine Intelligence journal. Future directions include applying PluGeN to world models in reinforcement learning, where it might be used to precisely control the training process.

4. Other works & achievements

This chapter includes short descriptions of other papers co-authored by me during my Ph.D. studies, followed by a full list of publications and other achievements. Although they are not directly connected with the adaptiveness problem and as such are not included as a main part of the thesis, they contain notable contributions in the area of deep learning.

Description of selected other publications My studies on Publication [V] that introduced PluGeN were preceded by other investigations of generative models. In particular, in "Segma: Semi-supervised gaussian mixture autoencoder" [79], we introduced a model for the semi-supervised classification and image generation tasks, based on the Cramer-World autoencoder architecture [51]. By using a Gaussian mixture as the prior distribution in the latent space of the model, we obtained a natural clustering of the data for multi-class problems. My contributions included implementing the theoretical model, running experiments, and compiling the results. The work was published in IEEE Transactions on Neural Networks and Learning Systems and was cited 11 times.

In another work, we studied a different perspective on generative modelling, by proposing a hypernetwork approach to image generation and the superresolution problem. In "Hypernetwork functional image representation" [49], we represent images using neural networks that map coordinates of pixel positions in the image to RGB values. The parameters of these neural networks are generated using another network (called the hypernetwork). The functional representation of the image allows us to easily control its resolution. My contributions to this work include empirical analysis of the properties of hypernetwork-based representations. This work was presented at ICANN 2019 and was cited 57 times.

Additionally, I worked on the problem of disentanglement, i.e. finding the ground-truth factors explaining the data, and its relation to multi-task settings. In "On the relationship between disentanglement and multi-task learning" [58], we studied this problem by investigating synthetic regression tasks in a teacher-student setting. We found that networks trained in the multi-task setting lead to more disentangled representations than networks trained on a single task. My contributions included the conceptualization and preparation of the synthetic dataset and preparing the manuscript. The paper was published at ECML-PKDD 2022 and was cited once.

I conducted a 6-month research internship at Lyft Level-5, an industry research division dedicated to solving the autonomous driving problem. During the internship, I worked on

building planning modules for self-driving vehicles that map semantical representations of the surroundings to low-level steering actions. As a result of my work, I co-authored two papers. In "Urban Driver: Learning to Drive from Real-world Demonstrations Using Policy Gradients" [74], we introduced the idea of using closed-loop training for planning, thus bringing together the behavioral cloning approach rooted in supervised learning with policy gradients from the reinforcement learning literature. I worked on implementing the proposed solution in a production environment and helped with preparing the manuscript. The work was presented at CoRL 2021 and was cited 23 times. Subsequently, in "SafetyNet: Safe Planning for Real-World Self-Driving Vehicles Using Machine-Learned Policies", we showed how to combine easily-scalable deep learning solutions with a reliable rule-based safety layer. This pairing allowed us to build a dependable solution that worked well in real-world tests on the roads of San Francisco. I worked on evaluating the model in simulation and compiling the results, as well as preparing the manuscript. The paper was presented at ICRA 2022 and was cited 21 times.

Finally, I co-authored a textbook on deep learning called "Głębokie uczenie. Wprowadzenie" (*Deep Learning. Introduction*), published by the Helion publishing company [82]. This work is a comprehensive introductory text on deep learning for undergraduate and graduate students. It covers topics ranging from classical machine learning approaches such as linear regression to complex deep learning-based architectures such as transformers. I wrote three out of thirteen chapters, explaining the core concepts behind multilayer perceptrons, recurrent neural networks, and the attention mechanism. Additionally, I compiled results and prepared visualization for some of the other chapters.

Chronological list of publications to which I contributed but they are not part of this dissertation.

- **Wołczyk, Maciej**. "Deep learning-based initialization for object packing." In *Schedae Informaticae* 27, 2018, 20 MEiN points.
- Klocek, Sylwester, Łukasz Maziarka, **Maciej Wołczyk**, Jacek Tabor, Jakub Nowak, and Marek Śmieja. "Hypernetwork functional image representation." In *International Conference on Artificial Neural Networks*, pp. 496-510, 2019, CORE B, 70 MEiN points.
- **Wołczyk, Maciej**, Jacek Tabor, Marek Śmieja, and Szymon Maszke. "Biologically-Inspired Spatial Neural Networks." In *Real Neurons & Hidden Units workshop at NeurIPS*, 2019, unrated by CORE, unrated by MEiN.
- Śmieja, Marek, **Maciej Wołczyk**, Jacek Tabor, and Bernhard C. Geiger. "Segma: Semi-supervised gaussian mixture autoencoder." In *IEEE Transactions on Neural Networks and Learning Systems* 32, pp. 3930-3941, 2020, 200 MEiN points.
- Wójcik, Bartosz, **Maciej Wołczyk**, Klaudia Bałazy, and Jacek Tabor. "Finding the optimal network depth in classification tasks." In *Machine Learning and Knowledge Discovery in*

Databases: European Conference (ECML PKDD), pp. 263-278, 2020, CORE A, 140 MEiN points.

- **Wołczyk, Maciej**, and Andrii Krutsylo. "Remember more by recalling less: Investigating the role of batch size in continual learning with experience replay (student abstract)." In Proceedings of the AAAI Conference on Artificial Intelligence, (AAAI), pp. 15923-15924, 2021, unrated by CORE, unrated by MEiN.
- Tabor, Jacek, Marek Śmieja, Łukasz Struski, Przemysław Spurek, and **Maciej Wołczyk**. "Głębokie Uczenie: Wprowadzenie", 2022.
- Scheel, Oliver, Luca Bergamini, **Maciej Wołczyk**, Błażej Osiński, and Peter Ondruska. "Urban driver: Learning to drive from real-world demonstrations using policy gradients." In Conference on Robot Learning (CoRL), pp. 718-728, 2022, unrated by CORE, unrated by MEiN.
- Vitelli, Matt, Yan Chang, Yawei Ye, Ana Ferreira, **Maciej Wołczyk**, Błażej Osiński, Moritz Niendorf, Hugo Grimmert, Qiangui Huang, Ashesh Jain, and Peter Ondruska. "Safetynet: Safe planning for real-world self-driving vehicles using machine-learned policies." In International Conference on Robotics and Automation (ICRA), pp. 897-904, 2022, CORE B, 70 MEiN points.
- Maziarka, Łukasz, Aleksandra Nowak, **Maciej Wołczyk**, and Andrzej Bedychaj. "On the relationship between disentanglement and multi-task learning." In Machine Learning and Knowledge Discovery in Databases: European Conference (ECML PKDD), pp. 625-641, 2022, CORE A, 140 MEiN points.
- Morawiecki, Paweł, Andrii Krutsylo, **Maciej Wołczyk**, and Marek Śmieja "Hebbian Continual Representation Learning". In Hawaii International Conference on System Sciences (HICSS), pp. 1259–1268, 2023, CORE A, 140 MEiN points.

List of grants to which I contributed (as a co-investigator):

- NCN Opus: Hypernetworks methods in Meta-Learning
National Science Center, Poland
principal investigator: prof. dr hab. Przemysław Spurek
2022 - 2023
- POB DigiWorld
Disentangled representation in generative models applicable in cheminformatics
principal investigator: prof. dr hab. Marek Śmieja
Jagiellonian University, Poland
2021 - 2022
- Team-Net: Bio-inspired artificial neural networks
Foundation for Polish Science, Poland

principal investigator: prof. dr hab. Jacek Tabor
2020 - 2023

- NCN Opus: Efficient unsupervised learning with applications in deep learning
National Science Center, Poland
principal investigator: prof. dr hab. Jacek Tabor
2018 - 2020

Established international cooperation

- prof. João Sacramento at ETH Zurich. Work in progress on two papers.
- prof. Razvan Pascanu at Google Deepmind. Co-written two papers on continual reinforcement learning, presented at NeurIPS 2021 and NeurIPS 2022.
- prof. Bernhard Geiger at Know-Center GmbH. Co-written a paper on a semi-supervised generative model, published in IEEE Transactions on Neural Networks and Learning Systems.
- Lyft/Woven Planet Level 5 autonomous driving team. Co-written two papers on planning in autonomous vehicles, presented at CoRL 2021 and ICRA 2022.

Established national cooperation

- prof. Piotr Miłoś and dr Łukasz Kuciński at IMPAN. Co-written two papers on continual reinforcement learning, presented at NeurIPS 2021 and NeurIPS 2022.
- dr Karol Piczak and prof. Tomasz Trzciński at Warsaw University of Technology. Co-written two papers, one on continual learning with guarantees, presented at ICML 2022, and another on early exits, presented at NeurIPS 2021.
- prof. Paweł Morawiecki and Andrii Krutsylo at IPIAN, co-written three papers on continual learning, presented at AAAI 2021 (as a student abstract), ICML 2022, and HICCS 2023.
- prof. Maciej Zięba and Patryk Wielopolski, at Wrocław University of Science and Technology, co-written a paper on adapting generative models, presented at AAAI 2022.

International research internships:

- ETH Zurich
PhD Intern with prof. João Sacramento
2022
- Woven Planet/Lyft Level-5
PhD Research intern in autonomous vehicles research team,
2021

Served as a reviewer for:

- International Conference on Machine Learning (ICML), in 2021 and 2022, received a *highlighted reviewer* reward both times, CORE rank A*, 200 MEiN points.
- Conference on Neural Information Processing Systems (NeurIPS) in 2021, 2022, and 2023, CORE rank A*, 200 MEiN points.
- International Conference on Learning Representations (ICLR) in 2022, received a *highlighted reviewer* reward, CORE rank A*, 200 MEiN points.
- Conference on Lifelong Learning Agents (CoLLAs) in 2022 and 2023, unrated.
- Conference on Robot Learning (CORL) in 2023, unrated.
- IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 200 MEiN points.
- Neural Networks journal, 200 MEiN points.
- Transactions on Machine Learning Research (TMLR), unrated.

Co-organized scientific events:

- Eastern European Machine Learning 2020 Summer School, Virtual
Co-organizer
Over 200 attendants, lectures from world-class experts on machine learning (e.g. Alex Graves, Kyunghyun Cho, Tomas Mikolov)
- Machine Learning Meets Neuroscience Summer School 2022, Kraków
Co-organizer
Over 150 attendants, lectures from world-class experts on machine learning (e.g. Andrew Saxe, Vincent Lepetit, Andrea Tagliasacchi, Razvan Pascanu)
- Workshop on Dynamic Neural Networks at ICML 2022
Co-organizer

5. Conclusion

This work summarizes five papers studying different aspects of the problem of adaptivity in deep learning. Publications [I, II, III] focus on the problem of continual learning, that is adapting to changes in the data stream. Here, we focus on reinforcement learning and learning with guarantees. Publications [IV, V] focus on adapting existing models to widely different settings and scenarios, specifically changing the computational paradigm and modifying the conditioning scheme in generative processes. Finally, the thesis lists other contributions of the author, including established cooperations, reviewing duties, and organized scientific events.

As the cost of training constantly grows, it is crucial to reuse existing models as often as possible and avoid unnecessary retraining from scratch. We believe that the presented works contribute to a significant advance in this direction and cover some of the areas that were previously underexplored. In future work, we would like to further extend the investigation of the continual learning scenario to understand how large neural networks, especially reinforcement learning models, perform at adapting to data shifts in large-scale training processes. Other future directions include examining different conditional computation methods (e.g. mixture-of-experts, advanced routing) for pre-trained neural networks and extending capabilities of PluGeN to world models in model-based reinforcement learning. Adaptation of neural networks remains a difficult and complicated question, yet a very important one.

Bibliography

- [1] Alessandro Achille and Stefano Soatto. Emergence of invariance and disentanglement in deep representations. *The Journal of Machine Learning Research*, 19(1):1947–1980, 2018.
- [2] Naveed Akhtar and Ajmal Mian. Threat of adversarial attacks on deep learning in computer vision: A survey. *Ieee Access*, 6:14410–14430, 2018.
- [3] Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, et al. Solving rubik’s cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019.
- [4] Eseoghene Ben-Iwhiwhu, Saptarshi Nath, Praveen K Pilly, Soheil Kolouri, and Andrea Soltoggio. Lifelong reinforcement learning with modulating masks. *arXiv preprint arXiv:2212.11110*, 2022.
- [5] Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*, pages 610–623, 2021.
- [6] Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*, volume 4. Springer, 2006.
- [7] 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.
- [8] Jörg Bornschein, Alexandre Galashov, Ross Hemsley, Amal Rannen-Triki, Yutian Chen, Arslan Chaudhry, Xuming He, Arthur Douillard, Massimo Caccia, Qixuang Feng, Jiajun Shen, Sylvestre-Alvise Rebuffi, Kitty Stacpoole, Diego de Las Casas, Will Hawkins, Angeliki Lazaridou, Yee Whye Teh, Andrei A. Rusu, Razvan Pascanu, and Marc’Aurelio Ranzato. Nevis’22: A stream of 100 tasks sampled from 30 years of computer vision research. *ArXiv*, abs/2211.11747, 2022.
- [9] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.
- [10] Massimo Caccia, Jonas Mueller, Taesup Kim, Laurent Charlin, and Rasool Fakoore. Task-agnostic continual reinforcement learning: In praise of a simple baseline. *arXiv preprint arXiv:2205.14495*, 2022.
- [11] Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip HS Torr, and Marc’Aurelio Ranzato. On tiny episodic memories in continual learning. *arXiv preprint arXiv:1902.10486*, 2019.

- [12] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. Continual learning: A comparative study on how to defy forgetting in classification tasks. *arXiv preprint arXiv:1909.08383*, 2(6):2, 2019.
- [13] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.
- [14] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- [15] Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.
- [16] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.
- [17] Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah Smith. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. *arXiv preprint arXiv:2002.06305*, 2020.
- [18] 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. *arXiv preprint arXiv:2010.11929*, 2020.
- [19] Arthur Douillard, Yifu Chen, Arnaud Dapogny, and Matthieu Cord. Plop: Learning without forgetting for continual semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4040–4050, 2021.
- [20] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR, 2017.
- [21] Stanislav Fort, Jie Ren, and Balaji Lakshminarayanan. Exploring the limits of out-of-distribution detection. *Advances in Neural Information Processing Systems*, 34:7068–7081, 2021.
- [22] Robert M French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135, 1999.
- [23] Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, pages 1587–1596. PMLR, 2018.
- [24] Jean-Baptiste Gaya, Thang Doan, Lucas Caccia, Laure Soulier, Ludovic Denoyer, and Roberta Raileanu. Building a subspace of policies for scalable continual learning. *arXiv preprint arXiv:2211.10445*, 2022.
- [25] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

- [26] Ian J Goodfellow, Mehdi Mirza, Da Xiao, Aaron Courville, and Yoshua Bengio. An empirical investigation of catastrophic forgetting in gradient-based neural networks. *arXiv preprint arXiv:1312.6211*, 2013.
- [27] Google. English - google scholar metrics. https://scholar.google.com/citations?view_op=top_venues. Accessed: 2023-04-03.
- [28] Sven Gowal, Krishnamurthy Dvijotham, Robert Stanforth, Rudy Bunel, Chongli Qin, Jonathan Uesato, Relja Arandjelovic, Timothy Mann, and Pushmeet Kohli. On the effectiveness of interval bound propagation for training verifiably robust models. *arXiv preprint arXiv:1810.12715*, 2018.
- [29] Yulan Guo, Hanyun Wang, Qingyong Hu, Hao Liu, Li Liu, and Mohammed Bennamoun. Deep learning for 3d point clouds: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 43(12):4338–4364, 2020.
- [30] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. PMLR, 2018.
- [31] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018.
- [32] Raia Hadsell, Dushyant Rao, Andrei A Rusu, and Razvan Pascanu. Embracing change: Continual learning in deep neural networks. *Trends in cognitive sciences*, 24(12):1028–1040, 2020.
- [33] Yizeng Han, Gao Huang, Shiji Song, Le Yang, Honghui Wang, and Yulin Wang. Dynamic neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11):7436–7456, 2021.
- [34] 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.
- [35] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.
- [36] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136*, 2016.
- [37] Timothy Hickey, Qun Ju, and Maarten H Van Emden. Interval arithmetic: From principles to implementation. *Journal of the ACM (JACM)*, 48(5):1038–1068, 2001.
- [38] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *International conference on learning representations*, 2017.
- [39] Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. *arXiv preprint arXiv:2204.03458*, 2022.
- [40] Xueyu Hou, Yongjie Guan, and Tao Han. Neulens: spatial-based dynamic acceleration of convolutional neural networks on edge. In *Proceedings of the 28th Annual International Conference on Mobile Computing And Networking*, pages 186–199, 2022.

- [41] 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.
- [42] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.
- [43] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8110–8119, 2020.
- [44] Yigitcan Kaya, Sanghyun Hong, and Tudor Dumitras. Shallow-deep networks: Understanding and mitigating network overthinking. In *International conference on machine learning*, pages 3301–3310. PMLR, 2019.
- [45] Khimya Khetarpal, Matthew Riemer, Irina Rish, and Doina Precup. Towards continual reinforcement learning: A review and perspectives. *Journal of Artificial Intelligence Research*, 75:1401–1476, 2022.
- [46] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [47] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.
- [48] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.
- [49] Sylwester Kloczek, Łukasz Maziarka, Maciej Wołczyk, Jacek Tabor, Jakub Nowak, and Marek Śmieja. Hypernetwork functional image representation. In *Artificial Neural Networks and Machine Learning–ICANN 2019: Workshop and Special Sessions: 28th International Conference on Artificial Neural Networks, Munich, Germany, September 17–19, 2019, Proceedings 28*, pages 496–510. Springer, 2019.
- [50] Jeremias Knoblauch, Hisham Husain, and Tom Diethe. Optimal continual learning has perfect memory and is np-hard. In *International Conference on Machine Learning*, pages 5327–5337. PMLR, 2020.
- [51] Szymon Knop, Przemysław Spurek, Jacek Tabor, Igor Podolak, Marcin Mazur, and Stanisław Jastrzebski. Cramer-wold auto-encoder. *The Journal of Machine Learning Research*, 21(1):6594–6621, 2020.
- [52] Misha Laskin, Kimin Lee, Adam Stooke, Lerrel Pinto, Pieter Abbeel, and Aravind Srinivas. Reinforcement learning with augmented data. *Advances in neural information processing systems*, 33:19884–19895, 2020.
- [53] Bracha Laufer-Goldshtein, Adam Fisch, Regina Barzilay, and Tommi Jaakkola. Efficiently controlling multiple risks with pareto testing. *arXiv preprint arXiv:2210.07913*, 2022.
- [54] Vincenzo Lomonaco and Davide Maltoni. Core50: a new dataset and benchmark for continuous object recognition. In *Conference on Robot Learning*, pages 17–26. PMLR, 2017.

- [55] David Lopez-Paz and Marc’Aurelio Ranzato. Gradient episodic memory for continual learning. *Advances in neural information processing systems*, 30, 2017.
- [56] Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 7765–7773, 2018.
- [57] Yoshitomo Matsubara, Marco Levorato, and Francesco Restuccia. Split computing and early exiting for deep learning applications: Survey and research challenges. *ACM Computing Surveys*, 55(5):1–30, 2022.
- [58] Łukasz Maziarka, Aleksandra Nowak, Maciej Wołczyk, and Andrzej Bedychaj. On the relationship between disentanglement and multi-task learning. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2022, Grenoble, France, September 19–23, 2022, Proceedings, Part I*, pages 625–641. Springer, 2023.
- [59] Gaurav Menghani. Efficient deep learning: A survey on making deep learning models smaller, faster, and better. *ACM Computing Surveys*, 2021.
- [60] Martial Mermillod, Aurélia Bugaïska, and Patrick Bonin. The stability-plasticity dilemma: Investigating the continuum from catastrophic forgetting to age-limited learning effects, 2013.
- [61] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.
- [62] Kyoung-Su Oh and Keechul Jung. Gpu implementation of neural networks. *Pattern Recognition*, 37(6):1311–1314, 2004.
- [63] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.
- [64] German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural networks*, 113:54–71, 2019.
- [65] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pages 1310–1318. Pmlr, 2013.
- [66] Filip Radenović, Giorgos Tolias, and Ondřej Chum. Fine-tuning cnn image retrieval with no human annotation. *IEEE transactions on pattern analysis and machine intelligence*, 41(7):1655–1668, 2018.
- [67] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.
- [68] Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *The Journal of Machine Learning Research*, 22(1):12348–12355, 2021.
- [69] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010, 2017.

- [70] Kui Ren, Tianhang Zheng, Zhan Qin, and Xue Liu. Adversarial attacks and defenses in deep learning. *Engineering*, 6(3):346–360, 2020.
- [71] Mark B Ring. Child: A first step towards continual learning. *Machine Learning*, 28(1):77–104, 1997.
- [72] Carlos Riquelme, Joan Puigcerver, Basil Mustafa, Maxim Neumann, Rodolphe Jenatton, André Susano Pinto, Daniel Keysers, and Neil Houlsby. Scaling vision with sparse mixture of experts. *Advances in Neural Information Processing Systems*, 34:8583–8595, 2021.
- [73] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.
- [74] Oliver Scheel, Luca Bergamini, Maciej Wolczyk, Błażej Osipiński, and Peter Ondruska. Urban driver: Learning to drive from real-world demonstrations using policy gradients. In *Conference on Robot Learning*, pages 718–728. PMLR, 2022.
- [75] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.
- [76] Jonathan Schwarz, Siddhant Jayakumar, Razvan Pascanu, Peter E Latham, and Yee Teh. Powerpropagation: A sparsity inducing weight reparameterisation. *Advances in neural information processing systems*, 34:28889–28903, 2021.
- [77] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarczyk, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.
- [78] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.
- [79] Marek Śmieja, Maciej Wołczyk, Jacek Tabor, and Bernhard C Geiger. Segma: Semi-supervised gaussian mixture autoencoder. *IEEE transactions on neural networks and learning systems*, 32(9):3930–3941, 2020.
- [80] Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and policy considerations for deep learning in nlp. *arXiv preprint arXiv:1906.02243*, 2019.
- [81] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.
- [82] Jacek Tabor, Marek Śmieja, Łukasz Struski, Przemysław Spurek, and Maciej Wołczyk. *Głębokie uczenie : wprowadzenie*. Helion, Gliwice, 2022.
- [83] Nima Tajbakhsh, Jae Y Shin, Suryakanth R Gurudu, R Todd Hurst, Christopher B Kendall, Michael B Gotway, and Jianming Liang. Convolutional neural networks for medical image analysis: Full training or fine tuning? *IEEE transactions on medical imaging*, 35(5):1299–1312, 2016.
- [84] Surat Teerapittayanon, Bradley McDanel, and Hsiang-Tsung Kung. Branchynet: Fast inference via early exiting from deep neural networks. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 2464–2469. IEEE, 2016.

- [85] Sergey Tulyakov, Ming-Yu Liu, Xiaodong Yang, and Jan Kautz. Mocogan: Decomposing motion and content for video generation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1526–1535, 2018.
- [86] Gido M Van de Ven and Andreas S Tolias. Three scenarios for continual learning. *arXiv preprint arXiv:1904.07734*, 2019.
- [87] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.
- [88] Sean Vasequez and Mike Lewis. Melnet: A generative model for audio in the frequency domain. *arXiv preprint arXiv:1906.01083*, 2019.
- [89] 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.
- [90] 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.
- [91] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International conference on machine learning*, pages 3987–3995. PMLR, 2017.
- [92] Barret Zoph, Irwan Bello, Sameer Kumar, Nan Du, Yanping Huang, Jeff Dean, Noam Shazeer, and William Fedus. Designing effective sparse expert models. *arXiv preprint arXiv:2202.08906*, 2022.

A. Full text of the Publications

This chapter contains the Publications described in the main text, in the following order:

- [I] Wołczyk, Maciej, Michał Zając, Razvan Pascanu, Łukasz Kuciński, and Piotr Miłoś. “Continual World: A robotic benchmark for continual reinforcement learning.” *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 28496-28510, 2021, CORE A*, 200 MEiN points.
- [II] Wołczyk, Maciej, Michał Zając, Razvan Pascanu, Łukasz Kuciński, and Piotr Miłoś. “Disentangling Transfer in Continual Reinforcement Learning.” *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 6304-6317, 2022, CORE A*, 200 MEiN points.
- [III] Wołczyk, Maciej, Karol J. Piczak, Bartosz Wójcik, Łukasz Pustelnik, Paweł Morawiecki, Jacek Tabor, Tomasz Trzcíński, and Przemysław Spurek. “Continual Learning with Guarantees via Weight Interval Constraints.” *International Conference on Machine Learning (ICML)*, pp. 23897-23911, 2022, CORE A*, 200 MEiN points.
- [IV] Wołczyk, Maciej, Bartosz Wójcik, Klaudia Bałazy, Igor T. Podolak, Jacek Tabor, Marek Śmieja, and Tomasz Trzcíński. “Zero Time Waste: Recycling Predictions in Early Exit Neural Networks.” *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 2516-2528, 2021, CORE A*, 200 MEiN points.
- [V] Wołczyk, Maciej, Magdalena Proszewska, Łukasz Maziarka, Maciej Zięba, Patryk Wielopolski, Rafał Kurczab, and Marek Śmieja. “PluGeN: Multi-Label Conditional Generation From Pre-Trained Models.” *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pp. 8647-8656. 2022, CORE A*, 200 MEiN points.

Continual World: A Robotic Benchmark For Continual Reinforcement Learning

Maciej Wołczyk*

Jagiellonian University
Kraków, Poland

maciej.wolczyk@doctoral.uj.edu.pl

Michał Zając*

Jagiellonian University
Kraków, Poland

emzajac@gmail.com

Razvan Pascanu

DeepMind
London, UK
razp@google.com

Łukasz Kuciński

Polish Academy of Sciences
Warsaw, Poland
lkucinski@impan.pl

Piotr Miłoś

Polish Academy of Sciences,
University of Oxford,
deepsense.ai
Warsaw, Poland
pmilos@impan.pl

Abstract

Continual learning (CL) — the ability to continuously learn, building on previously acquired knowledge — is a natural requirement for long-lived autonomous reinforcement learning (RL) agents. While building such agents, one needs to balance opposing desiderata, such as constraints on capacity and compute, the ability to not catastrophically forget, and to exhibit positive transfer on new tasks. Understanding the right trade-off is conceptually and computationally challenging, which we argue has led the community to overly focus on *catastrophic forgetting*. In response to these issues, we advocate for the need to prioritize forward transfer and propose *Continual World*, a benchmark consisting of realistic and meaningfully diverse robotic tasks built on top of Meta-World [54] as a testbed. Following an in-depth empirical evaluation of existing CL methods, we pinpoint their limitations and highlight unique algorithmic challenges in the RL setting. Our benchmark aims to provide a meaningful and computationally inexpensive challenge for the community and thus help better understand the performance of existing and future solutions. Information about the benchmark, including the open-source code, is available at <https://sites.google.com/view/continualworld>.

1 Introduction

Change is ubiquitous. Unsurprisingly, due to evolutionary pressure, humans can quickly adapt and creatively reuse their previous experiences. In contrast, although biologically inspired, deep learning (DL) models excel mostly in static domains that satisfy the i.i.d. assumption, as for example in image processing [28, 49, 10, 40], language modelling [52, 11] or biological applications [47]. As the systems are scaled up and deployed in open-ended settings, such assumptions are increasingly questionable; imagine, for example, a robot that needs to adapt to the changing environment and the wear-and-tear of its hardware. *Continual learning* (CL), an area that explicitly focuses on such problems, has been gaining more attention recently. The progress in this area could offer enormous advantages for deep neural networks [19] and move the community closer to the long-term goal of building intelligent machines [20].

*equal contribution

Evaluation of CL methods is challenging. Due to the sequential nature of the problem that disallows parallel computation, evaluation tends to be expensive, which has biased the community to focus on toy tasks. These are mostly in the domain of supervised learning, often relying on MNIST. In this work, we expand on previous discussions on the topic [45, 16, 30, 46] and introduce a new benchmark, *Continual World*. The benchmark is built on realistic robotic manipulation tasks from Meta-World [54], benefiting from its diversity but also being computationally cheap. Moreover, we provide shorter auxiliary sequences, all of which enable a quick research cycle. On the conceptual level, a fundamental difficulty of evaluating CL algorithms comes from the different desiderata for a CL solution. These objectives are often opposing each other, forcing practitioners to explicitly or implicitly make trade-offs in their algorithmic design that are data-dependent. *Continual World* provides more meaningful relationships between tasks, answering recent calls [19] to increase attention on forward transfer.

Additionally, we provide an extensive evaluation of a spectrum of commonly used CL methods. It highlights that many approaches can deal relatively well with *catastrophic forgetting* at the expense of other desiderata, in particular forward transfer. This emphasizes our call for focusing on *forward transfer* and the need for more benchmarks that allow for common structure among the tasks.

The main contribution of this work is a CL benchmark that poses optimizing forward transfer as the central goal and shows that existing methods struggle to outperform simple baselines in terms of the forward transfer capability. We release the code² both for the benchmark and 7 CL methods, which aims to provide the community helpful tools to better understand the performance of existing and future solutions. We encourage to visit the website³ of the project and participate in the Continual World Challenge.

2 Related work

The field of continual learning has grown considerably in the past years, with numerous works forming new subfields [23] and finding novel applications [48]. For brevity, we focus only on the papers proposing RL-based benchmarks and point to selected surveys of the entire field. [19] provide a high-level overview of CL and argue that learning in a non-stationary setting is a fundamental problem for the development of AI, highlighting the frequent connections to neuroscience. On the other hand, [13, 38] focus on describing, evaluating, and relating CL methods to each other, providing a taxonomy of CL solutions that we use in this work.

The possibility of applying CL methods in reinforcement learning scenarios has been explored for a long time, see [25] for a recent review. However, no benchmark has been widely accepted by the community so far, which is the aim of this work. Below we discuss various benchmarks and environments considered in the literature.

Supervised settings MNIST has been widely used to benchmark CL algorithms in two forms [27]. In the permuted MNIST, the pixels of images are randomly permuted to form new tasks. In the split MNIST, tasks are defined by classifying non-overlapping subsets of classes, e.g. 0 vs. 1 followed by 2 vs. 3. A similar procedure has been applied to various image classification tasks like CIFAR-10, CIFAR-100, Omniglot or mini-ImageNet [2, 46, 5]. Another benchmark is CORE50 [31], a dataset for continuous object recognition. Recent work [29] proposes a benchmark based on language modeling. We find that many of these benchmarks are challenging and allow to measure forgetting. However, we argue they are not geared towards measuring forward transfer or for highlighting important RL-specific characteristics of the CL problem.

Atari The Atari 2600 suite [8] is a widely accepted RL benchmark. Sequences of different Atari games have been used for evaluating continual learning approaches [43, 27]. Using Atari can be computationally expensive, e.g., training a sequence of ten games typically requires 100M steps or more. More importantly, as [43] notes, these games lack a meaningful overlap, limiting their relevance for studying transfers. **Continuous control** [32, 24] use continuous control tasks such as Humanoid or Walker2D. However, the considered sequences are short, and the range of experiments is limited. [34] use Meta-World tasks, similarly to us, for evaluations of their continual learning method, but the work is not aimed at building a benchmark. As such, it uses the Meta-World’s MT10 preset and

²https://github.com/awarelab/continual_world

³<https://sites.google.com/view/continualworld/home>

does not provide an in-depth analysis of the tasks or other CL methods. **Maze navigation** A set of 3D maze environments is used in [43]. The map structure and objects that the agent needs to collect change between tasks. It is not clear, though, if the tasks provide enough diversity. [30] propose CRLMaze, 3D navigation scenarios for continual learning, which solely concentrate on changes of the visual aspects. **StarCraft** [45] present a StarCraft campaign (11 tasks) to evaluate a high-level transfer of skills. The main drawback of this benchmark is excessive computational demand (often more than 1B frames). **Minecraft** [50] propose simple scenarios within the Minecraft domain along with a hierarchical learning method. The authors phrase the problem as lifelong learning and do not use typical CL methods. **Lifelong Hanabi** [36] consider a multi-agent reinforcement learning setting based on Hanabi, a cooperative game requiring significant coordination between agents. On the other hand, we focus on the single agent setting with changing environment, which allows us to bypass the computational complexity needed to model interactions between agents and highlight issues connected to learning in a changing world. **Causal World** [3] propose an environment for robotic manipulation tasks which share causal structure. Although they investigate issues deeply connected to learning in a changing world, such as generalization to new tasks and curricula, they do not directly consider continual learning. **Jelly Bean World** [39] provide interesting procedurally generated grid world environments. The suite is configurable and can host a non-stationary setting. It is unclear, however, if such environments reflect the characteristics of real-world challenges.

3 Continual learning background

Continual learning (CL) is an area of research which focuses on building algorithms capable of handling non-stationarity. They should be able to sequentially acquire new skills and solve novel tasks without forgetting the previous ones. Such systems are desired to accommodate over extended periods swiftly, which is often compared to human capabilities and alternatively dubbed as lifelong learning. CL is intimately related to multi-task learning, curriculum learning, meta-learning, with some key differences. Multi-task assumes constant access to all tasks, thus ignoring non-stationarity. Curriculum learning focuses on controlling the task ordering and often the learning time-span. Meta-learning, a large field of its own, sets the objective to develop procedures that allow fast adaptation within a task distribution and usually ignores the issue of non-stationarity.

The CL objective is operationalized by the training and evaluation protocols. The former typically consists of a sequence of tasks (their boundaries might be implicit and smooth). The latter usually involves measuring *catastrophic forgetting*, *forward transfer*, and *backward transfer*. The learning system might also have constrained resources: *computations*, *memory*, *size of neural networks*, and the *volume of data samples*. A fundamental observation is that the above aspects and desiderata are conflicting. For example, given unlimited resources, one might mitigate forgetting simply by storing everything in memory and paying a high computational cost of rehearsing all samples from the past.

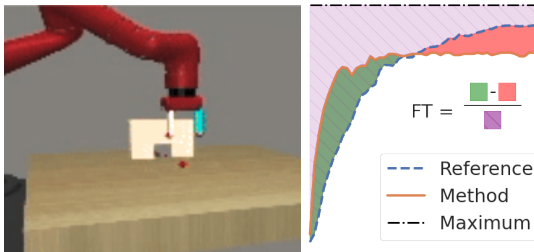


Figure 1: Left graph shows task PEG-UNPLUG-SIDE-V1 and the right graph presents forward transfer from SHELF-PLACE-V1 to PEG-UNPLUG-SIDE-V1. In this case $FT = 0.10$.

Another pair of objectives that are problematic for current methods are forgetting and forward transfer. For neural networks, existing methods propose to limit network plasticity. These alleviate the problem of forgetting, however, at the cost of choking the further learning process. We advocate for more nuanced approaches. Importantly, to make the transfer possible, our benchmark is composed of related tasks. We also put modest bounds on resources. This requirement is in line with realistic scenarios, demanding computationally efficient adaptation and inference. In a broader sense, we hope to address a data efficiency challenge, one of the most significant limitations of the current deep (reinforcement) learning methods. We conjecture that forward transfer might greatly improve the situation and possibly one day enable us to create systems with human-level cognition capabilities, in line with similar thoughts expressed in [19].

4 Continual World benchmark

Continual World is a new benchmark designed to be a testbed for evaluating RL agents on the challenges advocated by the CL paradigm, described in Section 3, as well as highlighting the RL-specific algorithmic challenges for CL (see Section 6.1). As such it is aimed at being valuable to both the CL and RL communities. Continual World consists of realistic robotic manipulation tasks, aligned in a sequence to enable the study of forward transfer. It is designed to be challenging while computationally accessible.⁴ The benchmark is based on Meta-World, a suite of robotic tasks already established in the community. This enables easy comparisons with the related fields of multi-task and meta-learning reinforcement learning, potentially highlighting one benefit of CL framing, namely that of dealing with different reward scales as we discuss more in detail in Appendix H. Continual World comes with open-source code that allows for easy development and testing of new algorithms and provides implementations of 7 existing algorithms. Finally, it allows highlighting RL-specific challenges for the CL setting. We believe that our work is a step in the right direction towards reliable benchmarks of CL. We realize, however, that it will need to evolve as the field progresses. We leave a discussion on future directions and limitations to Section 4.4.

4.1 Metrics

To facilitate further discussion, we start with defining metrics. These are rather standard in the CL setting [42]. Assume $p_i(t) \in [0, 1]$ to be the performance (success rate) of task i at time t . As a measure of performance, we take the average success rate of achieving a goal specified by a given task when using randomized initial conditions and stochastic policies (see also Section 4.3).⁵ Each task is trained for $\Delta = 1M$ steps. The main sequence has $N = 20$ tasks and the total sample budget is $T = N \cdot \Delta = 20M$. The i -th task is trained during the interval $t \in [(i-1) \cdot \Delta, i \cdot \Delta]$. We report the following metrics:

Average performance. The average performance at time t is (see Figure 3)

$$P(t) := \frac{1}{N} \sum_{i=1}^N p_i(t). \quad (1)$$

Its final value, $P(T)$, is a traditional metric used in the CL research. This is the objective we use for tuning hyperparameters. We have $P(t) \in [0, 1]$ for each t .

Forward transfer. We measure the forward transfer of a method as the normalized area between its training curve and the training curve of the reference, single-task, experiment, see Figure I. Let $p_i^b \in [0, 1]$ be the reference performance⁶ then the forward transfer for the task i , denoted by FT_i , is

$$FT_i := \frac{AUC_i - AUC_i^b}{1 - AUC_i^b}, \quad AUC_i := \frac{1}{\Delta} \int_{(i-1) \cdot \Delta}^{i \cdot \Delta} p_i(t) dt, \quad AUC_i^b := \frac{1}{\Delta} \int_0^\Delta p_i^b(t) dt,$$

The average forward transfer for all tasks, FT , is defined as

$$FT = \frac{1}{N} \sum_{i=1}^N FT_i. \quad (2)$$

We note that $FT_i \leq 1$ and they might be negative. In our experiments, we also measure backward transfer. As it is negligible, see Appendix E.1

Forgetting. For task i , we measure the decrease of performance after ending its training, i.e.

$$F_i = p_i(i \cdot \Delta) - p_i(T). \quad (3)$$

Similarly to FT , we report $F = \frac{1}{N} \sum_{i=1}^N F_i$. We have $F_i \leq 1$ for any i and consequently $FT \leq 1$. It is possible that F_i are negative, which would indicate backward transfer. We do not observe this in practice, see Appendix E.1

⁴We use 8-core machines without GPU. Training the CW20 sequence of twenty tasks takes about 100 hours. We also provide shorter 10 and 3 task sequences to speed up the experimental loop further.

⁵Stochastic evaluations are slightly more smooth and have little difference to the deterministic ones.

⁶Note that we avoid trivial tasks, for which $AUC_i^b = 1$ is making the metric ill-defined. Additionally, we acknowledge the dependency on the hyperparameters of the learning algorithm and that there are alternative quantities of interest like relative improvement in performance rather than faster learning.

4.2 Continual World tasks

This section describes the composition of Continual World benchmark and the rationale behind its design. We decided to base on Meta-World [54], a fairly new but already established robotic benchmark for multi-task and meta reinforcement learning. From a practical standpoint, Meta-World utilizes the open-source MuJoCo physics engine [51], prized for speed and accuracy. Meta-World provides 50 distinct manipulation tasks with everyday objects using a simulated robotic Sawyer arm. Although the tasks vary significantly, the structure and semantics of observation and action spaces remain the same, allowing for transfer between tasks. Each observation is a 12-dimensional vector containing (x, y, z) coordinates of the robot’s gripper and objects of interest in the scene. The 4-dimensional action space describes the direction of the arm’s movement in the next step and the gripper actuator delta. Reward functions are shaped to make each task solvable. In evaluations, we use a binary *success metric* based on the distance of the task-relevant object to its goal position. This metric is interpretable and enables comparisons between tasks. For more details about the rewards and evaluation metrics, see [54, Section 4.2, Section 4.3].

CW20, CW10, triplets sequences The core of our benchmark is CW20 sequence. Out of 50 tasks defined in Meta-World, we picked those that are not too easy or too hard in the assumed sample budget $\Delta = 1M$. Aiming to strike a balance between the difficulty of the benchmark and computational requirements, we selected 10 tasks. The tasks and their ordering were based on the transfer matrix (see the next paragraph), so that there is a high variation of forward transfers (both in the whole list and locally). We refer to these ordered tasks as CW10, and CW20 is CW10 repeated twice. We recommend using CW20 for final evaluation; however, CW10 is already very informative in most cases. Due to brevity constraints, we present an ablation with an alternative ordering of the tasks and a longer sequence of 30 tasks in Appendix G, however, these experiments do not alter our findings. Additionally, to facilitate a fast development cycle, we propose a set of triplets, sequences of three tasks which exhibit interesting learning dynamics.

The CW10 sequence is: HAMMER-V1, PUSH-WALL-V1, FAUCET-CLOSE-V1, PUSH-BACK-V1, STICK-PULL-V1, HANDLE-PRESS-SIDE-V1, PUSH-V1, SHELF-PLACE-V1, WINDOW-CLOSE-V1, PEG-UNPLUG-SIDE-V1.

Transfer matrix Generally, the relationship between tasks and its impact on learning dynamics of neural networks is hard to quantify, where semantic similarity does not typically lead to transfer [14]. To this end, we consider a minimal setting, in which we finetune on task t_2 a model pretrained on t_1 , using the same protocol as the benchmark (e.g., different output heads, see Section 4.3). This provides neural network-centric insight into the relationship between tasks summarized in Figure 2 and allows us to measure *low-level transfer* between tasks, i.e., the ability of the model to reuse previously acquired features. See Appendix D for more results and extended discussion.

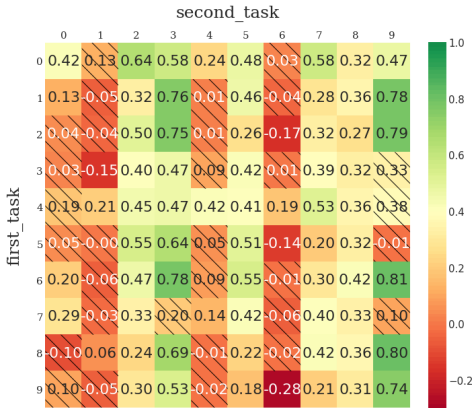


Figure 2: Transfer matrix, see Section 4.2. Each cell represents the forward transfer from the first task to the second one. We shaded the cells for which 0 belongs to their 90% confidence interval.

Notice that there are only a few negative forward transfer cases, and those are of a rather small magnitude (perhaps unsurprisingly, as the tasks are related). There are also visible patterns in the matrix. For instance, some tasks such as PEG-UNPLUG-SIDE-V1 or PUSH-BACK-V1 benefit from a relatively large forward transfer, (almost) irrespective of the first task. Furthermore, the average forward transfer given the second task (columns) is more variable than the corresponding quantity for the first task (rows).

Note that some transfers on the diagonal (i.e., between the same tasks) are relatively small. We made a detailed analysis of possible reasons, which revealed that the biggest negative impact is due to the replay buffer resets, which seems, however, unavoidable for off-diagonal cases, see Section 6.1 for details.

Importantly, we use this matrix to estimate what level of forward transfer a good CL method should be able to achieve. We expect that a model which is able to remember all meaningful

aspects of previously seen tasks would transfer at least as well as if one were just fine-tuning after learning the best choice between the previous tasks. For a sequence $t_1, \dots, t_N$ we set the reference forward transfer, RT, to be

$$\text{RT} := \frac{1}{N} \sum_{i=2}^N \max_{j < i} \text{FT}(t_j, t_i), \quad (4)$$

where $\text{FT}(t_j, t_i)$ is the transfer matrix value for t_j, t_i . For the CW20 sequence, the value is $\text{RT} = 0.46$. Note that a *model can do better* than this by composing knowledge from multiple previous tasks.

4.3 Training and evaluation details

We adapt the standard Meta-World setting to CL needs. First, we use separate policy heads for each task, instead of the original one-hot task ID inputs (we provide ablation experiments for this choice in Appendix G). Second, in each episode, we randomize the positions of objects in the scene to encourage learning more robust policies. We use an MLP network with 4 layers of 256 neurons.

For training, we use soft actor-critic (SAC) [17], a popular and efficient RL method for continuous domains. SAC is an off-policy algorithm using replay buffer, which is an important aspect for CL, particularly for methods relying on rehearsing old trajectories. SAC is based on the so-called maximum entropy principle; this results in policies which explore better and are more robust to changes in the environment dynamics. Both of these qualities might be beneficial in CL.

We note that the size of the neural network and optimization details of the SAC algorithm (like batch size) put constraints on "the amount of compute". Intentionally, these are rather modest, which is in line with CL desiderata, see Section 3. Similarly, we limit the number of timesteps to $1M$, which is a humble amount for modern-day deep reinforcement learning. We picked tasks to be challenging but not impossible within this budget. We note that training in the RL setting tends to be less stable than in the supervised one. We recommend using multiple seeds, in our experiments, we typically used 20 and calculate confidence intervals; we used the bootstrap method. We choose hyperparameters that maximize average performance [1]. In our experiments, we tune common parameters for SAC and the method-specific hyperparameters separately. All details of the training and evaluation setup are presented in Appendix A.

4.4 Limitations of Continual World

As any benchmark, we are fully aware that ours will not cover the entire spectrum of problems that one might be interested in. Here we summarize a few limitations that we hope to overcome in a future instantiation of this benchmark:

Input space We use a small 12 dimensional observation space. This is key to achieve modest computational demand. However, richer inputs could allow for potentially more interesting forms of transfer (e.g., based on visual similarity of objects) and would allow inferring the task from the observation, which is currently impossible.

Reliance on SAC We use the SAC algorithm [17], which is considered a standard choice for continuous robotic tasks. However, there is a potential risk of overfitting to the particularities of this algorithm and exploring alternative RL algorithms is important.

Task boundaries We rely on task boundaries. One can rely on task inference mechanisms (e.g. [35, 41]) to resolve this limitation, though we acknowledge the importance to extend the benchmark towards allowing and testing for task inference capabilities. Also, testing for algorithms dealing with continuous distributional drift is not possible in the current format.

Output heads We rely on using a separate head for each new task, similar to many works on continual learning. We opt for this variant based on its simplicity and better performance than using one-hot encoding to indicate a task. We believe that the lack of semantics of the one-hot encoding would further impede transfer, as the relationship between tasks can not be inferred. We carry ablation studies with using one-hot encoding as an input and a single head architecture, a setting that is already compatible with our benchmark. We regard this aspect as an important future work, and in particular, we are exploring alternative encoding of input to make this choice more natural. A coherent domain, like Continual World, provides a unique opportunity to exploit a consistent output layer as its semantics does not change between tasks.

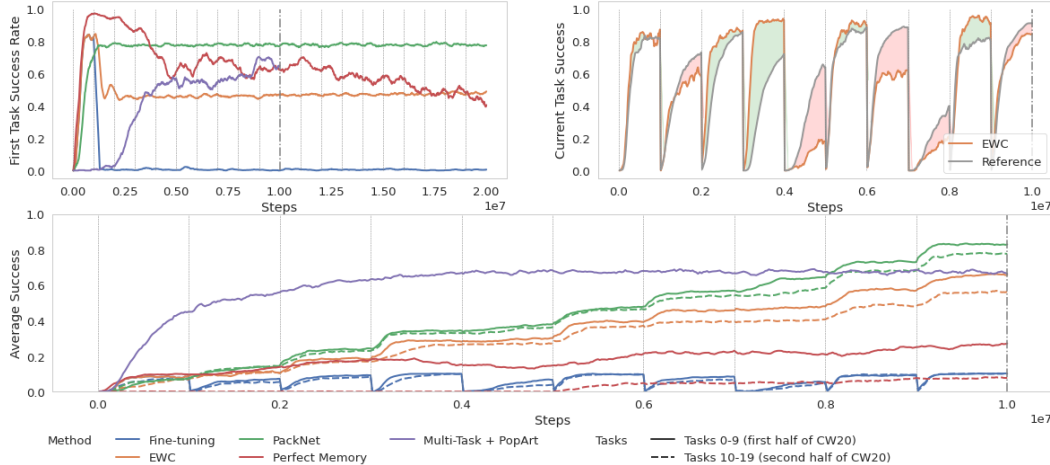


Figure 3: Training curves for selected CL methods and multi-task. The upper left panel shows the performance on the first task for a subset of methods throughout the whole training. Note that due to the use of different output heads, we do not see a second bump when revisiting this task at time $10M$. The upper right panel shows the performance on the current task being trained for EWC compared to a reference (a model learning only that task from scratch). The bottom plot shows the average performance. Solid lines show the performance of the model training on the first 10 tasks (where 1 means being able to solve all of them). Dashed lines show the performance of learning the same tasks in the second half of the benchmark. Note that dashed lower are below solid ones, indicating lower performance on the second pass, even if the agent has already previously learned the tasks and has access to relevant features.

The difficulty and number of tasks The number of tasks is relatively small. CW20, the main sequence we use, consists of only 10 different tasks, which are then repeated. We believe the repetition of tasks is important for a CL benchmark, leading to interesting observations. We check also that results are quantitatively similar on a sequence of 30 tasks, see Appendix G. However, longer sequences, potentially unbounded, are needed to understand the various limitations of existing algorithms. For example, the importance of *graceful forgetting* or dealing with systems that run out of capacity, a scenario where *there is no multi-task solution for the sequence of observed tasks*. This is particularly of interest for methods such as PackNet [33]. Additionally, we provide the number of tasks in advance. Dealing with an unknown number of tasks might raise further interesting questions. Finally, in future iterations of the benchmark, it is important to consider more complex tasks or more complex relationships between tasks to remain a challenge to existing methods. Our goal was to provide a benchmark that is approachable by existing methods, as not to stifle progress.

Low-level transfer We focus on low-level transfers via neural network features and weights. As such, we do not explicitly explore the ability of the learning process to exploit the compositionality of behavior or to rely on a more interesting semantic level. While we believe such research is crucial, we argue that solving low-level transfer is equally important and might be a prerequisite. So, for now, it is beyond the scope of this work, though future iterations of the benchmark could contain such scenarios.

5 Methods

We now sketch 7 CL methods evaluated on our benchmark. Some of them were developed for RL, while others were meant for the supervised learning context and required non-trivial adaptation. We aimed to cover different families of methods; following [13], we consider three classes: regularization-based, parameter isolation and replay methods. An extended description and discussion of these methods are provided in Appendix B.

Regularization-based Methods This family builds on the observation that one can reduce forgetting by protecting parameters that are important for the previous tasks. The most basic approach often

dubbed **L2** [27] simply adds a L_2 penalty, which regularizes the network not to stray away from the previously learned weights. In this approach, each parameter is equally important. **Elastic Weight Consolidation (EWC)** [27] uses the Fisher information matrix to approximate the importance of each weight. **Memory-Aware Synapses (MAS)** [4] also utilizes a weighted penalty, but the importance is obtained by approximating the impact each parameter has on the output of the network. **Variational Continual Learning (VCL)**, follows a similar path but uses variational inference to minimize the Kullback-Leibler divergence between the current distribution of parameters (posterior) and the distribution for the previous tasks (prior).

Parameter Isolation Methods This family (also called modularity-based) forbids any changes to parameters that are important for the previous tasks. It may be considered as a “hard” equivalent of regularization-based methods. **PackNet** [33] “packs” multiple tasks into a single network by iteratively pruning, freezing, and retraining parts of the network at task change. PackNet is closely related to progressive neural networks [43], developed in the RL context.

Replay Methods Methods of this family keep some samples from the previous tasks and use them for training or as constraints to reduce forgetting. We use a **Perfect Memory** baseline, a modification of our setting which remembers all the samples from the past (i.e., without resetting the buffer at the task change). We also implemented **Averaged Gradient Episodic Memory (A-GEM)** [12], which projects gradients from new samples as to not interfere with previous tasks. We find that A-GEM does not perform well on our benchmark.

Multi-task learning In multi-task learning, a field closely related to CL, tasks are trained simultaneously. By its design, it does not suffer from forgetting, however, it is considered to be hard as multiple tasks “compete for the attention of a single learning system”, see [21, 44]. We find that using reward normalization as in PopArt [21] is essential to achieve good performance. See Appendix E

6 Experiments

Now we present empirical results; these are evaluations of a set of 7 representative CL methods (as described in Section 5) on our Continual World benchmark. We focus on *forgetting and transfers* while keeping fixed constraints on computation, memory, number of samples, and neural network architecture. Our main empirical contributions are experiments on the long CW20 sequence and following high-level conclusions. For a summary see Table 1, Figure 3 and for an extensive discussion, we refer to Appendix E (including results for the shorter sequence, CW10). In Appendix C we provide various ablations and detailed analysis of sensitivity to the CL-specific hyperparameters.

Performance The performance (success rate) averaged over tasks (eq. (1)) is a typical metric for the CL setting. PackNet seems to outperform other methods, approaching 0.8 from the maximum of 1.0, outperforming multi-task solutions which might struggle with different reward scales, a problem elegantly avoided in the CL framing. Other methods perform considerably worse. A-GEM and Perfect Memory struggle. We further discuss possible reasons in Section 6.1

Forgetting We observe that most CL methods are usually efficient in mitigating forgetting. However, we did not notice any boost when revisiting a task (see Figure 3). Even if a different output head was employed, relearning the internal representation should have had an impact unless it changed considerably when revisiting the task. Additionally, we found A-GEM difficult to tune; consequently, with the best hyperparameter settings, it is relatively similar to the baseline fine-tuning method (see details in Appendix C).

method	performance	forgetting	f. transfer
Fine-tuning	0.05 [0.05, 0.06]	0.73 [0.72, 0.75]	0.20 [0.17, 0.23]
L2	0.43 [0.39, 0.47]	0.02 [0.00, 0.03]	-0.71 [-0.87, -0.57]
EWC	0.60 [0.57, 0.64]	0.02 [-0.00, 0.05]	-0.17 [-0.24, -0.11]
MAS	0.51 [0.49, 0.53]	0.00 [-0.01, 0.02]	-0.52 [-0.59, -0.47]
VCL	0.48 [0.46, 0.50]	0.01 [-0.01, 0.02]	-0.49 [-0.57, -0.42]
PackNet	0.80 [0.79, 0.82]	0.00 [-0.01, 0.01]	0.19 [0.15, 0.23]
Perfect Memory	0.12 [0.09, 0.15]	0.07 [0.05, 0.10]	-1.34 [-1.42, -1.27]
A-GEM	0.07 [0.06, 0.08]	0.71 [0.70, 0.73]	0.13 [0.10, 0.16]
MT	0.51 [0.48, 0.53]	—	—
MT (PopArt)	0.65 [0.63, 0.67]	—	—
RT	—	—	0.46

Table 1: Results on CW20, for CL methods and multi-task training. Metrics are defined in Section 4.1. RT is eq. (4). We used 20 seeds and provide 90% confidence intervals.

Transfers For all methods, forward transfer for the second ten tasks (and the same tasks are revisited) drops compared to the first ten tasks. This is in stark contrast to forgetting, which seems to be well under control. Among all methods, only fine-tuning and PackNet are able to achieve positive forward transfer (0.20 and 0.19, resp.) as well as on the first (0.32 and 0.21, resp.) and the second (0.08 and 0.17, resp.) half of tasks. However, these are considerably smaller than $RT = 0.46$, which in principle can even be exceeded, and which should be reached by a model that remembers all meaningful aspects of previously seen tasks, see (4). These results paint a fairly grim picture: we would expect improvement, rather than deterioration in performance, when revisiting previously seen tasks. There could be multiple reasons for this state of affairs. It could be attributed to the loss of plasticity, similar to the effect observed in [6]. Another reason could be related to the interference between CL mechanisms or setting and RL, for instance, hindering exploration. We did not observe any substantial cases of **backward transfer**, even though the benchmark is well suited to study this question due to the revisiting of tasks. See Appendix E.1

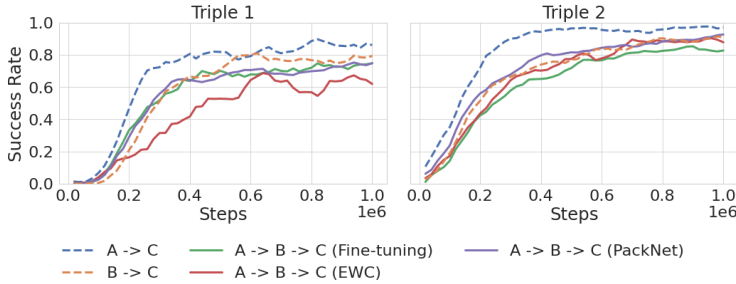


Figure 4: **How forgetting impacts the forward transfer.** Two different triplets of tasks learnt in sequence. An ideal agent learning on a sequence $A \rightarrow B \rightarrow C$ should have at least as good performance on task C as an agent which just learns $A \rightarrow C$. In reality, an interfering task B reduces this transfer, even when continual learning approaches are used.

final forward transfer (see Figure 4). The drive for reducing forgetting in CL agents has been primarily to perform well on previous tasks when we revisit them. With this example, we argue that an equally important reason to improve the memory of CL agents is to efficiently use past experiences to learn faster on new tasks. Currently, the tested CL methods often are not able to outperform the forgetful fine-tuning baseline. Observe that even the modularity-based PackNet approach struggles with this task. This possibly indicates that using the activation mask from task B is enough to deteriorate the performance.

PackNet PackNet stands out in our evaluations. We conjecture that developing related methods might be a promising research direction. Besides further increasing performance, one could mitigate the limitations of PackNet. PackNet relies on knowing task identity during evaluation. While this assumption is met in our benchmark, it is an interesting topic for future research to develop methods that cope without task identity. Another nuisance is that PackNet assigns some fixed fraction of parameters to a task. This necessitates knowledge of the length of the sequence in advance. Additionally, when the second ten tasks of CW20 start, PackNet performance degrades, showing its potentially inefficient use of capacity and past knowledge, given that the second ten tasks are identical with the first ten and hence no additional capacity is needed. In a broader context, we speculate that parameter isolation methods might be a promising direction towards better CL methods.

Resources usage In practical applications it is important to consider resources usage. All tested methods have relatively small overheads. For example, PackNet needs only 15% more time than the baseline fine-tuning and it requires 50% more neural network parameters (which is negligible when small networks like ours). See Appendix B.4 for details concerning other methods.

Other observations In stark contrast with the supervised learning setting, we found that replay based methods (Perfect Memory and A-GEM) suffer from poor performance. This is even though we allow for a generous replay, which could store the whole experience. Explaining and amending this situation is, in our view, an important research question. We conjecture that this happens due to the regularization of the critic network (which was unavoidable for these methods). We found multi-task

Triplets experiments We illustrate how forgetting and forward transfer interact with each other in a simpler setting of three task sequences, see Figure 4 and Appendix F. We focus on sequences of tasks $A \rightarrow B \rightarrow C$, where $A \rightarrow C$ has significant positive forward transfer and $B \rightarrow C$ has a smaller or even negative transfer. An efficient CL agent should be able to use information from A to get good performance on C . However, interference introduced by B reduces the final

learning attaining lower scores than PackNet, the best CL method and comparable to the second one, EWC. We think this suggests interesting research directions for multi-task learning.

6.1 RL-Related Challenges

Reinforcement learning brings a set of issues not present in the supervised learning setting, e.g., exploration, varying reward scales, and stochasticity of environments. We argue that it is imperative to have a reliable benchmark to assess the efficiency of CL algorithms with respect to these problems. We find that some current methods are not well adjusted to the RL setting and require non-trivial conceptual considerations and careful tuning of hyperparameters, see details in Appendix C

An important design choice is whether or not to regularize the critic in the actor-critic framework (e.g. in SAC). We find it beneficial to focus on reducing forgetting in the actor while allowing the critic to freely adapt to the current task (note that critic is used only in training of the current task), similar to [46]. On the other hand, a forgetful critic is controversial. This can be sharply seen when the same task is repeated and the critic needs to learn from scratch. Additionally, not all methods can be trivially adapted to the 'actor-only regularization' setting, as for example replay based methods. In Appendix C we examine these issues empirically, by showing experiments with critic regularization for EWC.

Another aspect is the exploration and its non-trivial impact on transfers. As it was observed, transfers from a given task to the same one are sometimes poor. We show in Appendix D.1 that this results from the fact that at the task change the replay buffer is emptied and SAC collects new samples from scratch, usually by using the uniform policy. Learning on these random samples reduces performance on the current task and thus the forward transfer. Experimentally, we find that not resetting the buffer or using the current policy for exploration improves the transfer on the diagonal.

7 Conclusions and Future Work

In this work, we present Continual World, a continual reinforcement learning benchmark, and an in-depth analysis of how existing methods perform on it. The benchmark is aimed at facilitating and standardizing the CL system evaluation, and as such, is released with code, including implementation of 7 representative CL algorithms. We argue for more attention to *forward transfer* and the interaction between forgetting and transfer, as many existing methods seem to sacrifice transfer to alleviate forgetting. In our opinion, this should not be the aim of CL, and we need to strike a different balance between these objectives.

We made several observations, both conceptual and empirical, which open future research directions. In particular, we conjecture that parameter isolation methods are a promising direction. Further, we identified a set of critical issues at the intersection of RL and CL. Resolving critic regularization and efficient use of multi-task replays seem to be the most pressing ones. Our benchmark highlights some challenges, which in our view are relevant and tangible now. In the long horizon, achieving high-level transfers, removing task boundaries, and scaling up are among significant goals for future editions of Continual World. Our work is foundational research and does not lead to any direct negative applications.

Acknowledgments and Disclosure of Funding

We would like to thank Stanisław Jastrzębski for stimulating talks and help while preparing the manuscript. The work of PM was supported by the Polish National Science Center grant UMO-2017/26/E/ST6/00622. The work of MW was funded by Foundation for Polish Science (grant no POIR.04.04.00-00-14DE/18-00 carried out within the Team-Net program co-financed by the European Union under the European Regional Development Fund. This research was supported by the PL-Grid Infrastructure. Our experiments were managed using <https://neptune.ai>. We would like to thank the Neptune team for providing us access to the team version and technical support.

References

- [1] Joshua Achiam. Spinning Up in Deep Reinforcement Learning. 2018.
- [2] Tameem Adel, Han Zhao, and Richard E. Turner. Continual learning with adaptive weights (CLAW). In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.
- [3] Ossama Ahmed, Frederik Träuble, Anirudh Goyal, Alexander Neitz, Manuel Wuthrich, Yoshua Bengio, Bernhard Schölkopf, and Stefan Bauer. Causalworld: A robotic manipulation benchmark for causal structure and transfer learning. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.
- [4] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part III*, volume 11207 of *Lecture Notes in Computer Science*, pages 144–161. Springer, 2018.
- [5] Rahaf Aljundi, Eugene Belilovsky, Tinne Tuytelaars, Laurent Charlin, Massimo Caccia, Min Lin, and Lucas Page-Caccia. Online continual learning with maximal interfered retrieval. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d’Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 11849–11860, 2019.
- [6] Jordan T. Ash and Ryan P. Adams. On warm-starting neural network training. In Hugo Larochelle, Marc’Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.
- [7] Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *CoRR*, abs/1607.06450, 2016.
- [8] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, jun 2013.
- [9] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In Francis R. Bach and David M. Blei, editors, *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*, pages 1613–1622. JMLR.org, 2015.
- [10] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.
- [11] 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, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.
- [12] Arslan Chaudhry, Marc’Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with A-GEM. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

- [13] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *arXiv preprint arXiv:1909.08383*, 2019.
- [14] Yunshu Du, Wojciech M. Czarnecki, Siddhant M. Jayakumar, Razvan Pascanu, and Balaji Lakshminarayanan. Adapting auxiliary losses using gradient similarity. *CoRR*, abs/1812.02224, 2018.
- [15] Bradley Efron and Robert J Tibshirani. *An introduction to the bootstrap*. CRC press, 1994.
- [16] Sebastian Farquhar and Yarin Gal. Towards robust evaluations of continual learning. *CoRR*, abs/1805.09733, 2018.
- [17] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Jennifer G. Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 1856–1865. PMLR, 2018.
- [18] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Soft actor-critic algorithms and applications. *CoRR*, abs/1812.05905, 2018.
- [19] Raia Hadsell, Dushyant Rao, Andrei A. Rusu, and Razvan Pascanu. Embracing change: Continual learning in deep neural networks. *Trends in Cognitive Sciences*, 24(12):1028 – 1040, 2020.
- [20] Demis Hassabis, Dharshan Kumaran, Christopher Summerfield, and Matthew Botvinick. Neuroscience-inspired artificial intelligence. *Neuron*, 95(2):245 – 258, 2017.
- [21] Matteo Hessel, Hubert Soyer, Lasse Espeholt, Wojciech Czarnecki, Simon Schmitt, and Hado van Hasselt. Multi-task deep reinforcement learning with popart. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 3796–3803. AAAI Press, 2019.
- [22] Ferenc Huszár. Note on the quadratic penalties in elastic weight consolidation. *Proceedings of the National Academy of Sciences*, page 201717042, 2018.
- [23] Khurram Javed and Martha White. Meta-learning representations for continual learning. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d’Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 1818–1828, 2019.
- [24] Christos Kaplanis, Murray Shanahan, and Claudia Clopath. Policy consolidation for continual reinforcement learning. 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 3242–3251. PMLR, 09–15 Jun 2019.
- [25] Khimya Khetarpal, Matthew Riemer, Irina Rish, and Doina Precup. Towards continual reinforcement learning: A review and perspectives, 2020.
- [26] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.
- [27] James Kirkpatrick, Razvan Pascanu, Neil C. Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *CoRR*, abs/1612.00796, 2016.

- [28] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In Peter L. Bartlett, Fernando C. N. Pereira, Christopher J. C. Burges, Léon Bottou, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*, pages 1106–1114, 2012.
- [29] Germán Kruszewski, Ionut-Teodor Sorodoc, and Tomas Mikolov. Evaluating online continual learning with calm, 2021.
- [30] Vincenzo Lomonaco, Karan Desai, Eugenio Culurciello, and Davide Maltoni. Continual reinforcement learning in 3d non-stationary environments. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR Workshops 2020, Seattle, WA, USA, June 14-19, 2020*, pages 999–1008. IEEE, 2020.
- [31] Vincenzo Lomonaco and Davide Maltoni. Core50: a new dataset and benchmark for continuous object recognition. In *1st Annual Conference on Robot Learning, CoRL 2017, Mountain View, California, USA, November 13-15, 2017, Proceedings*, volume 78 of *Proceedings of Machine Learning Research*, pages 17–26. PMLR, 2017.
- [32] Kevin Lu, Igor Mordatch, and Pieter Abbeel. Adaptive online planning for continual lifelong learning. *CoRR*, abs/1912.01188, 2019.
- [33] Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 7765–7773. IEEE Computer Society, 2018.
- [34] Jorge A. Mendez, Boyu Wang, and Eric Eaton. Lifelong policy gradient learning of factored policies for faster training without forgetting. In Hugo Larochelle, Marc’Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.
- [35] Kieran Milan, Joel Veness, James Kirkpatrick, Michael H. Bowling, Anna Koop, and Demis Hassabis. The forget-me-not process. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 3702–3710, 2016.
- [36] Hadi Nekoei, Akilesh Badrinarayanan, Aaron C. Courville, and Sarath Chandar. Continuous coordination as a realistic scenario for lifelong learning. In Marina Meila and Tong Zhang, editors, *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 8016–8024. PMLR, 2021.
- [37] Cuong V. Nguyen, Yingzhen Li, Thang D. Bui, and Richard E. Turner. Variational continual learning. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.
- [38] German Ignacio Parisi, Ronald Kemker, Jose L. Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, 2019.
- [39] Emmanouil Antonios Platanios, Abulhair Saparov, and Tom M. Mitchell. Jelly bean world: A testbed for never-ending learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.
- [40] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. *CoRR*, abs/2102.12092, 2021.
- [41] Dushyant Rao, Francesco Visin, Andrei A. Rusu, Razvan Pascanu, Yee Whye Teh, and Raia Hadsell. Continual unsupervised representation learning. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d’Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances*

- in *Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 7645–7655, 2019.
- [42] Natalia Díaz Rodríguez, Vincenzo Lomonaco, David Filliat, and Davide Maltoni. Don’t forget, there is more than forgetting: new metrics for continual learning. *CoRR*, abs/1810.13166, 2018.
 - [43] Andrei A. Rusu, Neil C. Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *CoRR*, abs/1606.04671, 2016.
 - [44] Tom Schaul, Diana Borsa, Joseph Modayil, and Razvan Pascanu. Ray interference: a source of plateaus in deep reinforcement learning. *CoRR*, abs/1904.11455, 2019.
 - [45] Jonathan Schwarz, Daniel Altman, Andrew Dudzik, Oriol Vinyals, Yee Whye Teh, and Razvan Pascanu. Towards a natural benchmark for continual learning. In *Continual learning Workshop, Neurips 2018*, 2018.
 - [46] Jonathan Schwarz, Wojciech Czarnecki, Jelena Luketina, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Progress & compress: A scalable framework for continual learning. In Jennifer G. Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 4535–4544. PMLR, 2018.
 - [47] Andrew W. Senior, Richard Evans, John Jumper, James Kirkpatrick, Laurent Sifre, Tim Green, Chongli Qin, Augustin Zidek, Alexander W. R. Nelson, Alex Bridgland, Hugo Penedones, Stig Petersen, Karen Simonyan, Steve Crossan, Pushmeet Kohli, David T. Jones, David Silver, Koray Kavukcuoglu, and Demis Hassabis. Improved protein structure prediction using potentials from deep learning. *Nat.*, 577(7792):706–710, 2020.
 - [48] Fan-Keng Sun, Cheng-Hao Ho, and Hung-Yi Lee. LAMOL: language modeling for lifelong language learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.
 - [49] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *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*, pages 6105–6114. PMLR, 2019.
 - [50] Chen Tessler, Shahar Givony, Tom Zahavy, Daniel J. Mankowitz, and Shie Mannor. A deep hierarchical approach to lifelong learning in minecraft. In Satinder P. Singh and Shaul Markovitch, editors, *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 1553–1561. AAAI Press, 2017.
 - [51] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2012, Vilamoura, Algarve, Portugal, October 7-12, 2012*, pages 5026–5033. IEEE, 2012.
 - [52] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008, 2017.
 - [53] Jeffrey Scott Vitter. Random sampling with a reservoir. *ACM Trans. Math. Softw.*, 11(1):37–57, 1985.
 - [54] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In Leslie Pack Kaelbling, Danica Kragic, and Komei Sugiura, editors, *3rd Annual*

Checklist

1. For all authors...
 - (a) Do the main claims made in the abstract and introduction accurately reflect the paper’s contributions and scope? [Yes]
 - (b) Did you describe the limitations of your work? [Yes] see Section 4.4
 - (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Section 7
 - (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
 - (a) Did you state the full set of assumptions of all theoretical results? [N/A]
 - (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
 - (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] The codes is included in the supplemental material.
 - (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] see details in Appendix A
 - (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] we used 20 random seeds, see also details in Appendix A.6
 - (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] see Appendix A.7
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
 - (a) If your work uses existing assets, did you cite the creators? [Yes] We base on the MetaWorld benchmark [54], which we clearly indicate a few times, including the abstract.
 - (b) Did you mention the license of the assets? [Yes] We use MIT licence; see Appendix A.1
 - (c) Did you include any new assets either in the supplemental material or as a URL? [Yes]
 - (d) Did you discuss whether and how consent was obtained from people whose data you’re using/curating? [N/A]
 - (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
 - (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
 - (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
 - (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

Disentangling Transfer in Continual Reinforcement Learning

Maciej Wołczyk*

Faculty of Mathematics and Computer Science
Jagiellonian University
Kraków, Poland
maciej.wolczyk@doctoral.uj.edu.pl

Michał Zając*

Faculty of Mathematics and Computer Science
Jagiellonian University
Kraków, Poland
emzajac@gmail.com

Razvan Pascanu

DeepMind
London, UK
razp@google.com

Łukasz Kuciński

Polish Academy of Sciences
Warsaw, Poland
lkucinski@impan.pl

Piotr Miłoś

Ideas NCBR,
Polish Academy of Sciences,
deepsense.ai
Warsaw, Poland
pmilos@impan.pl

Abstract

The ability of continual learning systems to transfer knowledge from previously seen tasks in order to maximize performance on new tasks is a significant challenge for the field, limiting the applicability of continual learning solutions to realistic scenarios. Consequently, this study aims to broaden our understanding of transfer and its driving forces in the specific case of continual reinforcement learning. We adopt SAC as the underlying RL algorithm and Continual World as a suite of continuous control tasks. We systematically study how different components of SAC (the actor and the critic, exploration, and data) affect transfer efficacy, and we provide recommendations regarding various modeling options. The best set of choices, dubbed ClonEx-SAC, is evaluated on the recent Continual World benchmark. ClonEx-SAC achieves 87% final success rate compared to 80% of PackNet, the best method in the benchmark. Moreover, the transfer grows from 0.18 to 0.54 according to the metric provided by Continual World.

1 Introduction

The ability of continual learning (CL) systems ([17, 22]) to utilize knowledge from previously seen tasks in order to maximize transfer on the current task is a significant challenge for the field. Achieving progress in this area would bring benefits both for real-life applications and multiple machine learning domains [24, 18, 46, 10], including reinforcement learning (RL), as advocated in [47]. In particular, it would constitute a critical step towards making efficient lifelong learning agents a reality.

The goal of this paper is to expand our understanding of transfer and its driving factors in continual reinforcement learning (CRL). As the underlying RL algorithm, we assume soft actor-critic (SAC), see [16], and use Continual World [47] as the suite of continuous control environments. We systematically study the critic and actor networks, the key components of SAC, with regard to their influence on transfer. Similarly, we measure the impact of various choices regarding exploration and buffer data usage. The low-level mechanisms of transfer are not yet fully understood even in the supervised

*equal contribution

learning case [28]. To the best of our knowledge, our work is the first one that undertakes a comprehensive study of this important topic in RL. To this end, we proceed in two stages: exploring a two-task setting and a full continual learning scenario.

We start by investigating a simplified two-tasks setting in Section 4. This allows us to leave out the impact of forgetting, as well as limit the choices regarding exploration and data handling. We use 100 pairs of robotic tasks from the Continual World benchmark. Here, we render two key observations: 1) the role of the critic is the most important for transfer, while exploration and actor play smaller, but non-negligible, parts; 2) contributions of the individual components are mostly independent. Additionally, we show that the concept of feature reuse which is often utilized to explain supervised transfer learning [28, 33] might not be directly applicable in RL.

In Section 5, we aim to understand new effects which emerge for the full continual learning scenario, typically in longer sequences. In CL context, we need to take into account forgetting, being mindful of the fact that existing CL methods often favor mitigating forgetting at the expense of transfer, see [47]. Main results include 1) reusing policies from previous tasks for exploration considerably improves performance; 2) behavioral cloning to rehearse past tasks is beneficial for both average performance and forward transfer, outperforming other considered methods; 3) regularizing the critic typically does not help for the performance of CL methods.

The result of our comprehensive analysis is a set of general recommendations. We also determine the combination of design choices that outperforms all other options, dubbed **ClonEx-SAC**. This method utilizes behavioral *cloning* to mitigate catastrophic forgetting for the actor. Moreover, at the beginning of each task, **ClonEx-SAC** queries all previous policies, the best of which generates initial *exploration* data. **ClonEx-SAC** achieves 87% final success rate compared to 80% of PackNet, the best method in the Continual World benchmark, see Figure 1. Importantly, we observe a sharp transfer increase from 0.18 to 0.54 in the metric provided in the benchmark. Notably, the value of forward transfer closely matches the reference forward transfer adjusted for exploration, which is a soft upper bound for transfer, as introduced in [47].

2 Related work

Continual learning algorithms are often categorized into three classes: regularization-based e.g. [2, 23, 31], parameter isolation e.g. [26] and rehearsal methods e.g. [6, 7]; see also CL survey papers [8, 17, 32]. [44, 4] advocate the need to develop CL methods suitable for reinforcement learning training as a necessary step towards learning artificial intelligent agents to operate in open-ended and changing environments. [22] provides a detailed review of this combination and a taxonomy of possible setups. [47] proposes a sequence of robotic tasks as a benchmark, comparing popular CL methods adapted to RL and advocating for putting more emphasis on transfer.

The authors of [20] show how the synaptic Benna-Fusi model can be added on top of value-based RL methods to mitigate forgetting at both intra- and inter-task scales. A simple approach to cloning policies from previous tasks is employed in [45], and a similar replay strategy has been used in [35]. [21] tackles the case when task boundaries are not provided. Although most of the research is concerned with model-free continual reinforcement learning, an approach to model-based continual RL was presented in [19].

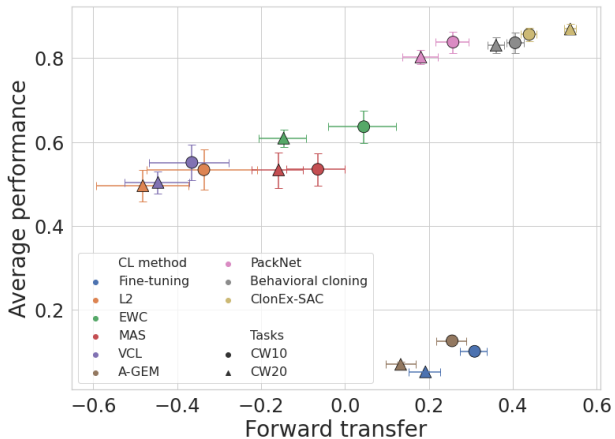


Figure 1: Performance of the ClonEx-SAC method compared with competitive baselines, on CW10 and CW20 task sequences. Average performance and forward transfer are shown, together with 90% bootstrap confidence intervals.

Transfer learning, which focuses on the reuse of machine learning models, has been extremely successful recently. In computer vision, convolutional neural networks [24, 18] and vision transformers [11] pre-trained on large datasets can be repurposed and fine-tuned on the target task. Modern transformer-based models [46, 10] trained on large natural language corpora turned out to be very flexible and can be adapted to diverse downstream tasks with surprising efficiency [34, 25]. General surveys of transfer learning techniques are provided in [52, 41]. Interestingly, recent research [29, 50] suggests that there are still some gaps in our understanding of transfer learning. [29] analyzes the low-level reasons for transfer, exhibiting surprising phenomena such as transfer between datasets with permuted images. [50] performs large-scale experiments investigating representation transfer in a wide variety of visual tasks.

In reinforcement learning scenarios, the structure of the underlying MDP can be exploited to facilitate the transfer. [42, 51, 43, 40] present methods on how to find and use mappings between different domains. [30, 5] apply reward function reshaping. [27, 39] achieve transfer by means of high-level skills and hierarchical RL. Other lines of work exploit the model structure [36, 13] or enforce modularity [3, 9]. In this work, we aim to complement these studies, by focusing on the benefits of reusing neural network parameters, and other choices that exploit the RL structure, like exploration and data rehearsal.

3 Background

3.1 Continual learning and reinforcement learning

Continual learning tackles the problem of learning in non-stationary settings [8]. Typically, the solution is expected to perform well on all encountered tasks, although various metrics expressing different requirements are formulated. The popular CL desiderata include reducing the *forgetting* on previous tasks and increasing the *forward transfer* on the new tasks, i.e. speeding up the learning by reusing knowledge from previous tasks [12, 47]. Other desiderata focus on limiting resources, such as the number of samples, computation time, model size, or additional memory size. These requirements are often conflicting, so usually some trade-offs have to be made [17, 47, 32].

Combining CL with RL adds another layer of complexity. In this work, we focus on the SAC algorithm [16], which is often considered to be the method of choice for continuous control RL [49, 48, 38]. As an actor-critic algorithm, it is based on the interplay between its two parts, see Section 3.2. This is a fairly complicated algorithmic setup, which presents a number of challenges when used jointly with CL.

In particular, since the optimization of the critic and actor networks is intertwined, it is hard to understand and decouple the impact of individual components. Additionally, because of this interplay, training biases get easily exacerbated, often leading to inferior performance or even a collapse. Another complication is that the training objectives for the actor and the critic are different. The critic minimizes the Bellman error which is known to be a fragile objective [14] susceptible to training biases and might correlate poorly with the value error (which we would like to minimize). As the actor optimizes over predictions of the critic, it might also suffer from these problems, even if less directly. Finally, since the policy and the data we see change during the training, there is an inherent distribution shift present, even within a single task.

3.2 SAC

In our study, we focus on soft actor-critic (SAC) [15], an off-policy actor-critic RL algorithm, based on the maximum entropy principle. **The critic** strives to approximate the entropy-corrected Q -function under the current policy, optimizing the Bellman error. **The actor** tries to find actions that maximize the Q -function. **The replay buffer** holds the seen experience and provides data for the actor and critic updates at each learning step. **The exploration policy** is used to gather data at the beginning of each task for a set number of K steps. By default, in most SAC implementations, this means sampling actions uniformly over the action space.

3.3 Continual World

We perform our experiments on the Continual World [47] benchmark. It contains a set of realistic robotic tasks, where a simulated Sawyer robot manipulates everyday objects. The structure of the observation and action spaces remains the same between the tasks; an observation is a 12-dimensional vector describing the coordinates of the robot’s gripper and relevant objects. The 4-dimensional action space describes the gripper movement. In training, a dense reward function is used to make the tasks solvable; in evaluation, the binary success metric is used to indicate whether the desired goal has been reached. The tasks are arranged in sequences and training in each task lasts for 1M steps. CW10 sequence contains 10 different tasks arranged in a fixed order. CW20 consists of CW10 repeated twice, allowing to measure how much knowledge can be transferred in case of task repetitions. We use both CW10 and CW20 in our evaluations, as well as shorter sequences containing pairs of tasks from CW10.

3.4 Metrics

Following standard practice in continual learning literature, we report average performance and forgetting metrics. We also measure transfer as defined in [47]. Below we briefly recall these three metrics. Assume $p_i(t) \in [0, 1]$ to be the performance (success rate) of task i at time t , and that each of the N tasks is trained for Δ steps, so the total number of steps is $T = N \cdot \Delta$.

Average performance The average performance at time t is defined as $P(t) := \frac{1}{N} \sum_{i=1}^N p_i(t)$. Its final value, $P(T)$, is a scalar summary of the performance and is presented in the result tables.

Forward transfer The forward transfer is computed as a normalized area between the training curve of the measured run and the training curve of a reference curve from training from scratch. Let us denote by $p_i^b \in [0, 1]$ the reference performance. Then the forward transfer on task i , FT_i , is defined as

$$FT_i := \frac{AUC_i - AUC_i^b}{1 - AUC_i^b}, \quad AUC_i := \frac{1}{\Delta} \int_{(i-1) \cdot \Delta}^{i \cdot \Delta} p_i(t) dt, \quad AUC_i^b := \frac{1}{\Delta} \int_0^{\Delta} p_i^b(t) dt.$$

The average forward transfer for all tasks, FT, is defined as $FT = \frac{1}{N} \sum_{i=1}^N FT_i$.

Forgetting For the task i , one can measure a drop in performance after the end of learning on this task as $F_i = p_i(i \cdot \Delta) - p_i(T)$. Forgetting metric is defined as $F = \frac{1}{N} \sum_{i=1}^N F_i$.

3.5 Experimental setup

We follow the experimental setup from [47]. The actor and the critic are implemented as two separate MLP networks, each with 4 hidden layers of 256 neurons. We refer to the 4 hidden layers as the *backbone* and the last output layer as the *head*. By default, we assume the *multi-head* (MH) setting, where each task has its separate output head, but we also consider the *single-head* (SH) setting, where only a single head is used for all tasks. The SAC exploration phase takes $K = 10k$ steps. All experiments in this paper were performed with 10 different seeds unless noted otherwise. We compute 90% confidence intervals through bootstrapping. More details on the experimental setup can be found in Appendix A.

4 Transfer in isolation

In this section, we study *what enables transfer between RL tasks*. We assume a two-task setting, where we measure the forward transfer from the first to the second task, disregarding issues specific to continual learning (e.g. forgetting), which we defer to the next section. We utilize all 100 pairs of CW10 tasks, see Section 3.3, to evaluate the impact of *critic*, *actor*, and *exploration* given by SAC.

We will say that the actor or the critic are *carried over* (from the previous tasks) if their parameters are reused as the initialization in the next task; otherwise, the parameters are re-initialized. We also refer to the exploration policy as being carried over, if we use the policy from the previous task (or tasks) to gather the data during the first 10k steps of the SAC exploration phase (see Section 3.2);

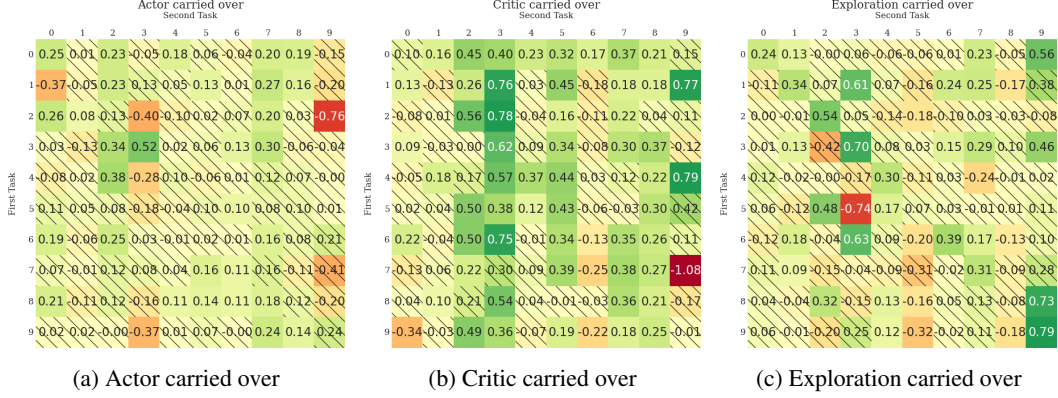


Figure 2: The effect of carrying over different components on the performance on pairs of tasks from CW10. We shade an entry if the 90% confidence interval contains 0, indicating that we cannot be sure whether the component which was carried over makes a difference.

otherwise, a uniform random exploration policy is being used. We use both multi-head (MH) and single-head (SH) settings, with the former being default.

Figure 2 illustrates the impact of the individual components on transfer for each pair. The (i, j) -th entry in the matrix contains the forward transfer value when carrying over components from task i to task j . Table 1 presents the aggregated statistics from the matrices given in Figure 2: the average FT (including and excluding the diagonal), and the number of pairs with positive, negative, and neutral FT². Table 2 reports the transfer properties for all possible combinations of components present in Table 1, omitting single-head critic (since it performs worse in Table 1).

4.1 Carrying over SAC components

From the results presented above, we draw two key observations. First, the role of the critic is the most important for FT, while exploration and actor play smaller, but non-negligible, parts. Second, the components are "transfer independent", in the sense that the transfer of the combination of the components is close to the sum of transfers yielded by each component alone.

The evidence for the first finding is presented in detail for each pair in Figure 2 and summarized in Table 1. More precisely, the average forward transfer across all pairs attributed to carrying over of the critic equals 0.2 (resp. 0.15) for MH (resp. SH) setup. This separates the critic from the actor and exploration, which yield (for the default MH setup) 0.06 and 0.09, respectively.

The importance of the critic is further emphasized by showing that restraining its learning capabilities, even when the weights are initialized to the parameters learned in the previous task, negatively impacts FT. This is shown in the last row of Table 1, where only the critic's head is allowed to train, while the body of the network is kept frozen and carried over from the previous task. This result goes against our understanding of transfer in supervised learning, where feature reuse is a common technique (e.g. in vision [28, 33]). However, the deterioration in FT can be explained by RL-specific factors. Namely, freezing the backbone can hinder both the policy training (since the mechanics of SAC intertwines actor and critic) and the critic training (due to inflated Bellman errors).

As to the second finding, i.e. the "transfer independence" of the components, the results of the underlying analysis are presented in Table 2. We observe that the reported FT for the combination of components follows closely the sum of FTs for individual components (reported in Table 1). Furthermore, we observe that including all the components results in the highest transfer of 0.35.

There is a couple of remaining interesting observations. First, Figure 2 exhibits several vertical patterns, meaning that transfer depends more on the second task. Second, the effect on transfer increases on the diagonal, when the exploration is carried over. This seems reasonable since the

²We say that a pair has positive (resp. negative) FT if the corresponding confidence interval is above (resp. below) 0. Otherwise, we mark it as neutral.

Table 1: Summary of the transfer statistics from the transfer matrices when transferring only a single component. FT and FT (no diag) represent average forward transfer across all pairs with and without considering the diagonal (transfer from a task to the same task), respectively. Subsequent columns denote the number of pairs with the positive, negative, and neutral transfer.

name	FT	FT (no diag)	# pos.	# neg.	# neutral
Actor (MH)	0.06 [0.03, 0.10]	0.05 [0.01, 0.09]	30	5	65
Critic (MH)	0.20 [0.17, 0.23]	0.19 [0.16, 0.23]	54	5	41
Exploration	0.09 [0.06, 0.13]	0.06 [0.03, 0.10]	28	9	63
Actor (SH)	0.12 [0.09, 0.15]	0.12 [0.09, 0.15]	37	1	62
Critic (SH)	0.15 [0.12, 0.18]	0.13 [0.10, 0.16]	41	19	40
Critic (train only head)	-1.29 [-1.33, -1.25]	-1.30 [-1.35, -1.26]	0	100	0

Table 2: Summary of transfer statistics when multiple components are carried over. We observe that impact of each component is largely independent of other components. That is, FT when carrying over multiple components is close to the sum of FT when carrying over each of them separately.

name	FT	FT (no diag)	# pos.	# neg.	# neutral
Actor (MH) + Critic (MH)	0.27 [0.24, 0.30]	0.25 [0.22, 0.29]	58	4	38
Actor (SH) + Critic (MH)	0.29 [0.26, 0.32]	0.28 [0.25, 0.31]	59	2	39
Actor (MH) + Exp.	0.16 [0.12, 0.20]	0.14 [0.10, 0.18]	39	3	58
Actor (SH) + Exp.	0.21 [0.17, 0.24]	0.18 [0.15, 0.22]	53	0	47
Critic (MH) + Exp.	0.30 [0.27, 0.33]	0.28 [0.25, 0.31]	64	2	34
Actor (SH) + Critic (MH) + Exp.	0.36 [0.33, 0.38]	0.33 [0.29, 0.36]	68	0	32
Actor (MH) + Critic (MH) + Exp.	0.35 [0.31, 0.38]	0.32 [0.29, 0.36]	70	1	29

policy in the new task is initialized to the already learned policy on the same task. Finally, resetting the head (MH setup) is beneficial in the case of the critic, while it hurts the actor.

5 Transfer in continual learning

In Section 4.1, we focused on direct transfer in the two-task setting. Now, we move to the full continual learning scenario, which brings two main differences: 1) we measure the performance of all tasks in the sequence, so forgetting now plays a significant role; 2) typically, we consider longer sequences of tasks of length 10 and 20 (CW10 and CW20, respectively). For longer sequences, forgetting and transfer may have complex mutual interactions. To reduce forgetting, CL methods usually apply some kind of regularization to the model, which in turn may be harmful to transfer. On the other hand, transfer benefits from accumulated knowledge – if forgetting is not mitigated, there might be nothing to transfer from.

We will investigate three main themes. The first one is reusing previous policies for exploration. For long sequences, there are multiple design choices available compared to the two-task scenarios. Secondly, we investigate CL with data reuse, an approach successful in supervised learning. We show that the CRL setup is more complex and requires careful investigation. Finally, given the importance of the critic for transfer (see Section 4), we study whether the critic should be regularized or not, and conclude that typically, the answer is negative.

We study these issues in conjunction with various CL methods: Fine-tuning, Perfect memory, EWC, PackNet, L2, A-GEM, MAS, and VCL. These are standard CL approaches adopted and tested in the RL setting [47], see details in Appendix B. We note that CL methods used here are mostly successful in mitigating forgetting; in this section, we report average performance and forward transfer, deferring forgetting to the Appendices C and E.

5.1 Exploration

When using the SAC algorithm, at the beginning of each task, there is a short period of exploration with a random policy, see Section 3.2. The experiments in Section 4.1 showed that the transfer

increases if the policy from the previous task is used instead. Now, we pass from two-task scenarios to longer ones, and analyze the following options for choosing exploration policy, which we call: *random*, *preceding*, *uniform-previous*, and *best-return*, and define them as follows. In the first task, we always use a random policy, and assume that the tasks are numbered from 1 to N .

Consider now $i \in \{2, \dots, N\}$. For the *random* strategy, we randomly sample from the action space, which is a default choice for SAC. For the other strategies, at the beginning of each exploration episode, we choose a previous actor head to generate data instead of the random policy. In the case of the *preceding* strategy, we use the $(i - 1)$ -th actor’s head. For *uniform-previous* policy, we use the j -th actor’s head, where $j := \text{RANDOM_UNIFORM}(\{1, \dots, i - 1\})$. Finally, in *best-return* strategy, we first try every possible head, and then act using the $j_{\max}$ -th actor’s head, where $j_{\max} := \arg\max_{j \in \{1, \dots, i-1\}} R_j^i$; R_j^i is the return of the j -th head policy on the i -th task.

We evaluate how these strategies interact with various CL methods. We pick Fine-tuning, Behavioral cloning, L2, EWC, and PackNet. The results for two well-performing methods, EWC and PackNet, are presented in Table 3, with the rest being deferred to Appendix E. For EWC, choosing any non-random policy significantly improves upon the baseline random strategy. This is particularly visible in the CW20 sequence, which contains repeated tasks, and arguably can benefit more from an informed strategy like best-return. Interestingly, the results for the rather simple uniform-previous approach are quite competitive. We observe increased performance also for other methods except for PackNet, for which effects are negligible.

Table 3: Average performance and forward transfer for different exploration strategies on CW10 and CW20 sequences. Strategies are added on top of EWC and PackNet methods.

Method, exploration	CW10 perf.	CW10 f. transfer	CW20 perf.	CW20 f. transfer
EWC, random	0.63 [0.60, 0.66]	0.03 [-0.04, 0.09]	0.60 [0.59, 0.62]	-0.14 [-0.19, -0.09]
EWC, preceding	0.70 [0.67, 0.73]	0.09 [0.03, 0.15]	0.61 [0.59, 0.64]	-0.14 [-0.19, -0.09]
EWC, uniform-previous	0.72 [0.69, 0.75]	0.24 [0.19, 0.28]	0.70 [0.68, 0.73]	0.21 [0.17, 0.25]
EWC, best-return	0.70 [0.68, 0.73]	0.25 [0.21, 0.28]	0.71 [0.69, 0.73]	0.28 [0.25, 0.31]
PackNet, random	0.84 [0.81, 0.86]	0.26 [0.22, 0.29]	0.80 [0.79, 0.82]	0.18 [0.14, 0.22]
PackNet, preceding	0.84 [0.82, 0.85]	0.24 [0.20, 0.27]	0.81 [0.80, 0.83]	0.20 [0.16, 0.24]
PackNet, uniform-previous	0.84 [0.81, 0.86]	0.21 [0.15, 0.26]	0.80 [0.78, 0.82]	0.23 [0.18, 0.27]
PackNet, best-return	0.85 [0.83, 0.86]	0.23 [0.20, 0.26]	0.82 [0.81, 0.83]	0.23 [0.21, 0.25]

5.2 Data rehearsal

Rehearsal techniques work very well in supervised continual learning [7]. In RL, two major approaches to utilizing previous data have been studied: applying them as offline data using SAC loss, and behavioral cloning of the previous policies. The former, dubbed Perfect memory, was reported to perform poorly [47]. Behavioral cloning achieves more promising results [45, 35]. We study these two approaches with an emphasis on the effects on transfer.

In Perfect memory, all the experiences are kept in the buffer. SAC training is applied to data from the current task and offline data from the previous ones. In Behavioral cloning, an additional small buffer is filled at the end of training on each task. We annotate a subset of samples from the main SAC buffer using the trained actor and critic networks. When training the new task, we sample data from expert buffers and apply auxiliary losses (with different weights), minimizing the KL divergence between current and saved outputs for the actor and L2 distance for the critic; see details in Appendix B.

Firstly, we study the effect of rehearsal on transfer in the two-task scenario, using 100 task pairs from CW10, as in Section 4.1. We observe that using Perfect memory or cloning both actor and the critic has a detrimental effect on transfer, providing more evidence that critic regularization can be catastrophic. On the other hand, cloning only the actor has a neutral effect; we report results for these and more setups in Appendix E. As such, in the remaining Behavioral cloning experiments, we regularize only the actor, unless noted otherwise.

Secondly, we perform experiments on longer sequences, CW10 and CW20; see Table 4. For reference, we include two methods tested in [47], Fine-tuning and PackNet. Fine-tuning achieves the highest transfer and PackNet the highest overall performance out of the methods tested in [47]. Behavioral

Table 4: Average performance and forward transfer for Perfect memory and Behavioral cloning methods, as described in Section 5.2. Fine-tuning and PackNet are included for reference.

method	CW10 perf.	CW10 f. transfer	CW20 perf.	CW20 f. transfer
Perfect memory	0.27 [0.24, 0.30]	-1.13 [-1.23, -1.04]	0.09 [0.06, 0.12]	-1.32 [-1.41, -1.24]
Behavioral cloning	0.84 [0.81, 0.86]	0.41 [0.38, 0.43]	0.83 [0.81, 0.85]	0.36 [0.34, 0.38]
Fine-tuning	0.10 [0.10, 0.10]	0.31 [0.27, 0.34]	0.05 [0.05, 0.05]	0.19 [0.15, 0.23]
PackNet	0.84 [0.81, 0.86]	0.26 [0.22, 0.29]	0.80 [0.79, 0.82]	0.18 [0.14, 0.22]

cloning performs very well. In terms of the average performance, it is on par with PackNet on CW10 and better on CW20. Importantly, it significantly outperforms the baselines in terms of transfer. We can see that Perfect memory works poorly, in line with the existing literature. In Appendix C, we present the results for five other CL methods benchmarked in [47].

In the end, we observe an interesting phenomenon. While behavioral cloning does not improve transfer in two-task scenario, it has a positive effect for the longer sequences. This result hints that perhaps the learner accumulates knowledge of the previous tasks and, thus, can reuse the most relevant parts of the past to improve the training of the current task. Additionally, perhaps behavioral cloning loss acts as a regularizer and helps shape more general features, thus further improving transfer.

5.3 Regularizing the critic

This section is devoted to the study of critic regularization in CRL methods. Since in our formulation of the problem, the primary objective of CRL is the final performance of the actor, we have some flexibility in how we treat the critic. We can even completely ignore forgetting in the critic, as recommended in [47]. Other works suggest that regularization might be beneficial [21].

To understand this issue better, we carefully measure the performance while varying the strength of the regularization, by changing the critic regularization coefficients for EWC, L2, and Behavioral cloning. We first find a good value for the actor regularization coefficient, with the critic regularization coefficient being set to 0. Then, with this value, we perform the sweep over the critic coefficients, covering a wide range from 1×10^{-10} to 100, and run training on the CW10 sequence. For all three methods, we observe that for the smallest values of critic regularization, the performance is similar to the version without critic regularization, and then after some threshold, performance visibly deteriorates. In the case of Behavioral cloning, it drops from 0.82 (no critic regularization) to 0.77 (critic regularization coefficient = 0.001) and then further, see Table 15. The complete results are presented in Appendix E.3.

This confirms the practical recommendation from [47] to regularize only the actor. One possible explanation is that TD -learning used for the critic is highly sensitive to biases introduced by regularization.

6 Combining the improvements: ClonEx-SAC

Based on the experimental findings presented so far, we propose to combine the discovered enhancements in a simple method for continual reinforcement learning. This method significantly improves the performance in the Continual World benchmark [47]. In particular, we observe a sharp transfer increase to a value that matches a soft upper bound for transfer introduced in [47].

We incorporate the following choices in the proposed method:

- We use behavioral cloning for the actor, which, as we showed in Section 5.2, effectively mitigates forgetting and increases transfer.
- We use best-return exploration, as described in Section 5.1, which efficiently reuses old policy heads for faster exploration.
- As indicated in Section 5.3, we do not use any CL regularization for the critic.

- We use multiple output heads for both actor and critic to profit from transferred representations without introducing too much bias in the new tasks, as discussed in Section 4.1.

We dub the method **ClonEx-SAC** to reflect the usage of the behavioral **cloning**, improved **exploration**, and SAC algorithm.

We compare **ClonEx-SAC** with the behavioral cloning and 7 methods considered in [47], on the CW10 and CW20 sequences. We present results in Figure 1 (see Introduction) and Appendix C. **ClonEx-SAC** achieves 87% final performance compared to 80% of PackNet, the best method in [47].

The forward transfer of **ClonEx-SAC**, improves sharply from 0.19, the best previous result, to 0.54. Notably, **ClonEx-SAC**’s result closely matches the reference forward transfer, see below. We conjecture that this excellent transfer is an important factor in the final performance. We also notice that improvements brought separately by behavioral cloning and the best-return exploration strategy work well together.

Reference forward transfer (RT) was introduced in [47] as a soft upper bound for transfer. For a sequence of tasks $t_1, \dots, t_N$, it is defined as $RT := \frac{1}{N} \sum_{i=2}^N \max_{j < i} FT(t_j, t_i)$, where $F(t_j, t_i)$ denotes the forward transfer for the pair of tasks t_j, t_i .

Intuitively, RT estimates the level of forward transfer, which could be achieved when a method is able to remember and transfer all meaningful aspects of previously seen tasks. Note that in principle, higher values of RT could still be achievable if the knowledge from the previous tasks is composed. In our setup, the values of RT are 0.44 for CW10 and 0.55 for CW20. In both cases, they are closely matched by the forward transfer of **ClonEx-SAC**. We note that our RT values are higher than the one reported in [47], since their work does not take into account the effects of improved exploration.

7 Limitations

We are fully aware that our analyses do not cover the entire spectrum of problems that one might be interested in when studying transfer in CRL. Here, we summarize a few limitations of our work:

- We build on top of the SAC algorithm. There is a risk that some of the conclusions from this paper would differ for another choice of the underlying RL method.
- We focus on the Continual World suite. There is a possibility that some of the results from this paper would differ in environments from other domains or with different, potentially structured state spaces.
- **ClonEx-SAC** requires retaining data from previous tasks, which may not always be feasible (e.g., due to privacy concerns).

8 Conclusions

In this work, we identify and study some of the key factors contributing to transfer in continual reinforcement learning. In the first part of the study, we focus on the transfer alone, disregarding other CL desiderata, and analyze how different components of the SAC algorithm (actor, critic, exploration) contribute to it. We identify the critic as the leading component.

In the second part, we study further effects that are relevant to the full continual learning setup with long task sequences. In particular, we show that behavioral cloning and reusing previous policies for exploration significantly improve both transfer and the final performance. This leads to a new method, **ClonEx-SAC**, which outperforms considered baselines.

We believe that this work constitutes the first step toward understanding the mechanisms behind transfer in continual reinforcement learning. There are still important issues to be resolved, e.g., pinpointing the exact role of feature reuse or the interplay between transfer and forgetting. We hope that these will be addressed by the community in the future.

Acknowledgments and Disclosure of Funding

The work of Maciej Wołczyk was supported by the National Centre of Science (Poland) Grant No. 2021/43/B/ST6/01456. The work of Piotr Miłoś was supported by the Polish National Science Center grant UMO-2017/26/E/ST6/00622 and UMO-2019/35/O/ST6/03464. This research was supported by the PL-Grid Infrastructure. Our experiments were managed using <https://neptune.ai>. We would like to thank the Neptune team for providing us access to the team version and technical support.

References

- [1] Joshua Achiam. Spinning Up in Deep Reinforcement Learning. 2018.
- [2] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part III*, volume 11207 of *Lecture Notes in Computer Science*, pages 144–161. Springer, 2018.
- [3] Jacob Andreas, Dan Klein, and Sergey Levine. Modular multitask reinforcement learning with policy sketches. In *International Conference on Machine Learning*, pages 166–175. PMLR, 2017.
- [4] Bowen Baker, Ingmar Kanitscheider, Todor M. Markov, Yi Wu, Glenn Powell, Bob McGrew, and Igor Mordatch. Emergent tool use from multi-agent autocurricula. *CoRR*, abs/1909.07528, 2019.
- [5] Tim Brys, Anna Harutyunyan, Matthew E Taylor, and Ann Nowé. Policy transfer using reward shaping. In *AAMAS*, pages 181–188, 2015.
- [6] Arslan Chaudhry, Marc’Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with A-GEM. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.
- [7] Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet Kumar Dokania, Philip H. S. Torr, and Marc’Aurelio Ranzato. Continual learning with tiny episodic memories. *CoRR*, abs/1902.10486, 2019.
- [8] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *arXiv preprint arXiv:1909.08383*, 2019.
- [9] Coline Devin, Abhishek Gupta, Trevor Darrell, Pieter Abbeel, and Sergey Levine. Learning modular neural network policies for multi-task and multi-robot transfer. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 2169–2176. IEEE, 2017.
- [10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina N. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. 2018.
- [11] 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. *arXiv preprint arXiv:2010.11929*, 2020.
- [12] Sebastian Farquhar and Yarin Gal. Towards robust evaluations of continual learning. *CoRR*, abs/1805.09733, 2018.
- [13] Chrisantha Fernando, Dylan Banarse, Charles Blundell, Yori Zwols, David Ha, Andrei A Rusu, Alexander Pritzel, and Daan Wierstra. Pathnet: Evolution channels gradient descent in super neural networks. *arXiv preprint arXiv:1701.08734*, 2017.
- [14] Scott Fujimoto, David Meger, Doina Precup, Ofir Nachum, and Shixiang Shane Gu. Why should i trust you, bellman? the bellman error is a poor replacement for value error, 2022.

- [15] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. PMLR, 2018.
- [16] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Soft actor-critic algorithms and applications. *CoRR*, abs/1812.05905, 2018.
- [17] Raia Hadsell, Dushyant Rao, Andrei A. Rusu, and Razvan Pascanu. Embracing change: Continual learning in deep neural networks. *Trends in Cognitive Sciences*, 24(12):1028 – 1040, 2020.
- [18] 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.
- [19] Yizhou Huang, Kevin Xie, Homanga Bharadhwaj, and Florian Shkurti. Continual model-based reinforcement learning with hypernetworks. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 799–805. IEEE, 2021.
- [20] Christos Kaplanis, Murray Shanahan, and Claudia Clopath. Continual reinforcement learning with complex synapses. In *International Conference on Machine Learning*, pages 2497–2506. PMLR, 2018.
- [21] Samuel Kessler, Jack Parker-Holder, Philip J. Ball, Stefan Zohren, and Stephen J. Roberts. Same state, different task: Continual reinforcement learning without interference. *CoRR*, abs/2106.02940, 2021.
- [22] Khimya Khetarpal, Matthew Riemer, Irina Rish, and Doina Precup. Towards continual reinforcement learning: A review and perspectives, 2020.
- [23] James Kirkpatrick, Razvan Pascanu, Neil C. Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *CoRR*, abs/1612.00796, 2016.
- [24] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In Peter L. Bartlett, Fernando C. N. Pereira, Christopher J. C. Burges, Léon Bottou, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*, pages 1106–1114, 2012.
- [25] Kevin Lu, Aditya Grover, Pieter Abbeel, and Igor Mordatch. Pretrained transformers as universal computation engines. *arXiv preprint arXiv:2103.05247*, 2021.
- [26] Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 7765–7773. IEEE Computer Society, 2018.
- [27] Neville Mehta, Sriraam Natarajan, Prasad Tadepalli, and Alan Fern. Transfer in variable-reward hierarchical reinforcement learning. *Machine Learning*, 73(3):289–312, 2008.
- [28] Behnam Neyshabur, Hanie Sedghi, and Chiyuan Zhang. What is being transferred in transfer learning? *Advances in neural information processing systems*, 33:512–523, 2020.
- [29] Behnam Neyshabur, Hanie Sedghi, and Chiyuan Zhang. What is being transferred in transfer learning? In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 512–523. Curran Associates, Inc., 2020.

- [30] Andrew Y Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Icml*, volume 99, pages 278–287, 1999.
- [31] Cuong V. Nguyen, Yingzhen Li, Thang D. Bui, and Richard E. Turner. Variational continual learning. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.
- [32] German Ignacio Parisi, Ronald Kemker, Jose L. Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, 2019.
- [33] Aniruddh Raghu, Maithra Raghu, Samy Bengio, and Oriol Vinyals. Rapid learning or feature reuse? towards understanding the effectiveness of maml. *arXiv preprint arXiv:1909.09157*, 2019.
- [34] Machel Reid, Yutaro Yamada, and Shixiang Shane Gu. Can wikipedia help offline reinforcement learning? *CoRR*, abs/2201.12122, 2022.
- [35] David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. *Advances in Neural Information Processing Systems*, 32, 2019.
- [36] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.
- [37] Jonathan Schwarz, Daniel Altman, Andrew Dudzik, Oriol Vinyals, Yee Whye Teh, and Razvan Pascanu. Towards a natural benchmark for continual learning. In *Continual learning Workshop, Neurips 2018*, 2018.
- [38] Kun Shao, Zhentao Tang, Yuanheng Zhu, Nannan Li, and Dongbin Zhao. A survey of deep reinforcement learning in video games. *arXiv preprint arXiv:1912.10944*, 2019.
- [39] Lorenzo Steccanella, Simone Totaro, Damien Allonsius, and Anders Jonsson. Hierarchical reinforcement learning for efficient exploration and transfer. *arXiv preprint arXiv:2011.06335*, 2020.
- [40] Erik Talvitie and Satinder P Singh. An experts algorithm for transfer learning. In *IJCAI*, pages 1065–1070, 2007.
- [41] Chuanqi Tan, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu. A survey on deep transfer learning. In Věra Kůrková, Yannis Manolopoulos, Barbara Hammer, Lazaros Iliadis, and Ilias Maglogiannis, editors, *Artificial Neural Networks and Machine Learning – ICANN 2018*, pages 270–279, Cham, 2018. Springer International Publishing.
- [42] Matthew E Taylor and Peter Stone. Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, 10(7), 2009.
- [43] Matthew E Taylor, Peter Stone, and Yaxin Liu. Transfer learning via inter-task mappings for temporal difference learning. *Journal of Machine Learning Research*, 8(9), 2007.
- [44] Open Ended Learning Team, Adam Stooke, Anuj Mahajan, Catarina Barros, Charlie Deck, Jakob Bauer, Jakub Sygnowski, Maja Trebacz, Max Jaderberg, Michaël Mathieu, Nat McAleese, Nathalie Bradley-Schmieg, Nathaniel Wong, Nicolas Porcel, Roberta Raileanu, Steph Hughes-Fitt, Valentin Dalibard, and Wojciech Marian Czarnecki. Open-ended learning leads to generally capable agents. *CoRR*, abs/2107.12808, 2021.
- [45] René Traoré, Hugo Caselles-Dupré, Timothée Lesort, Te Sun, Guanghang Cai, Natalia Díaz Rodríguez, and David Filliat. Discorl: Continual reinforcement learning via policy distillation. *CoRR*, abs/1907.05855, 2019.

- [46] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008, 2017.
- [47] Maciej Wołczyk, Michał Zając, Razvan Pascanu, Lukasz Kucinski, and Piotr Miłoś. Continual world: A robotic benchmark for continual reinforcement learning. *Advances in Neural Information Processing Systems*, 34, 2021.
- [48] Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. *Advances in Neural Information Processing Systems*, 33:5824–5836, 2020.
- [49] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In Leslie Pack Kaelbling, Danica Kragic, and Komei Sugiura, editors, *3rd Annual Conference on Robot Learning, CoRL 2019, Osaka, Japan, October 30 - November 1, 2019, Proceedings*, volume 100 of *Proceedings of Machine Learning Research*, pages 1094–1100. PMLR, 2019.
- [50] Xiaohua Zhai, Joan Puigcerver, Alexander Kolesnikov, Pierre Ruysen, 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. The visual task adaptation benchmark. *CoRR*, abs/1910.04867, 2019.
- [51] Zhuangdi Zhu, Kaixiang Lin, and Jiayu Zhou. Transfer learning in deep reinforcement learning: A survey. *arXiv preprint arXiv:2009.07888*, 2020.
- [52] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning, 2021.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to **[Yes]**, **[No]**, or **[N/A]**. You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? **[Yes]** See Section ??.
- Did you include the license to the code and datasets? **[No]** The code and the data are proprietary.
- Did you include the license to the code and datasets? **[N/A]**

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...

- (a) Do the main claims made in the abstract and introduction accurately reflect the paper’s contributions and scope? **[Yes]**
- (b) Did you describe the limitations of your work? **[Yes]** In Section 7
- (c) Did you discuss any potential negative societal impacts of your work? **[No]** We do not see this work as having a significant societal impact.
- (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? **[Yes]**

2. If you are including theoretical results...

- (a) Did you state the full set of assumptions of all theoretical results? [N/A]
 - (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
- (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] The code, including the scripts used to run the experiments from the paper, are in the supplementary materials.
 - (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
 - (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] We conduct each experiment with multiple seeds (at least 10)
 - (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] We describe these details in Appendix F.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
- (a) If your work uses existing assets, did you cite the creators? [N/A]
 - (b) Did you mention the license of the assets? [N/A]
 - (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
 - (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
 - (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
- (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
 - (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
 - (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

Continual Learning with Guarantees via Weight Interval Constraints

Maciej Wołczyk^{*1} Karol J. Piczak^{*1} Bartosz Wójcik¹ Łukasz Pustelnik¹ Paweł Morawiecki² Jacek Tabor¹
Tomasz Trzciniński¹³⁴ Przemysław Spurek¹

Abstract

We introduce a new training paradigm that enforces interval constraints on neural network parameter space to control forgetting. Contemporary Continual Learning (CL) methods focus on training neural networks efficiently from a stream of data, while reducing the negative impact of catastrophic forgetting, yet they do not provide any firm guarantees that network performance will not deteriorate uncontrollably over time. In this work, we show how to put bounds on forgetting by reformulating continual learning of a model as a continual contraction of its parameter space. To that end, we propose Hyperrectangle Training, a new training methodology where each task is represented by a hyperrectangle in the parameter space, fully contained in the hyperrectangles of the previous tasks. This formulation reduces the NP-hard CL problem back to polynomial time while providing full resilience against forgetting. We validate our claim by developing InterContiNet (Interval Continual Learning) algorithm which leverages interval arithmetic to effectively model parameter regions as hyperrectangles. Through experimental results, we show that our approach performs well in a continual learning setup without storing data from previous tasks.

1. Introduction

Learning from a continuous stream of data is natural for humans, as new experiences come sequentially in our life. Yet, artificial neural network models fail to exhibit the very same skill (McCloskey & Cohen, 1989; Ratcliff, 1990; French,

^{*}Equal contribution ¹Faculty of Mathematics and Computer Science, Jagiellonian University, Kraków, Poland ²Institute of Computer Science, Polish Academy of Sciences, Warsaw, Poland ³Warsaw University of Technology, Warsaw, Poland ⁴Tooploox. Correspondence to: Maciej Wołczyk <maciej.wolczyk@doctoral.uj.edu.pl>.

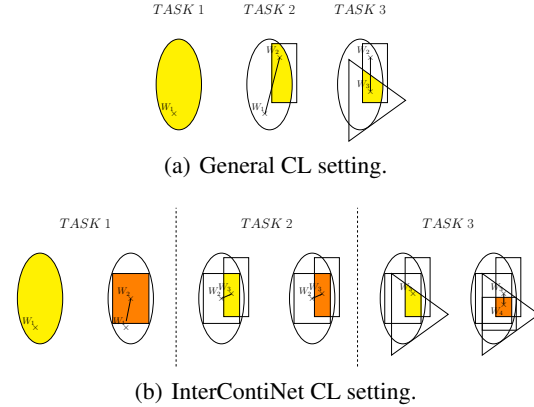


Figure 1. Visualization of parameter space regions that perform well on tasks learned sequentially. For a new task, weights θ_i must lie at the intersection of consecutive regions for the model to perform well on previous tasks. In the general CL setting, shown in Figure 1(a) from (Knoblauch et al., 2020), such areas can have arbitrary shapes. We introduce the Hyperrectangle Training paradigm where the parameter spaces are modeled with hyperrectangles, colored orange in Figure 1(b). This formulation puts guarantees on forgetting while reducing the calculation of NP-hard CL problem back to polynomial time.

1999; Goodfellow et al., 2014). Although they generally deal well with solving increasingly complex tasks, their inability to acquire new knowledge without *catastrophically forgetting* what they learned previously is considered one of the critical roadblocks to reaching human-like intelligence.

Continual learning is a rapidly growing field of machine learning that aims to solve this limitation and to bridge the gap between human and machine intelligence. Although several methods effective at reducing forgetting when learning new tasks were proposed (Kirkpatrick et al., 2017; Lee et al., 2017; Li & Hoiem, 2017; Lopez-Paz & Ranzato, 2017; Shin et al., 2017; Zenke et al., 2017; Aljundi et al., 2018; Masse et al., 2018; Liu et al., 2018; Rolnick et al., 2019; van de Ven et al., 2020), they typically do not provide any firm guarantees about the degree of forgetting experienced by the model, which renders them inapt for safety-critical applications, such as autonomous driving or robotic manipulation. For instance, an autonomous car that was trained online during winter cannot forget how to drive in the snow, after acquiring new driving skills during summer. On the

other hand, methods that do guarantee lack of forgetting, e.g. [Rusu et al. \(2016\)](#); [Mallya & Lazebnik \(2018\)](#); [Mallya et al. \(2018\)](#), operate in conditions that are far from real life — they assume that the task identity is known at inference, and hence we are allowed to adapt the network for each task.

In this work, we identify those limitations of prior research and introduce a new learning paradigm for continual learning which puts strict bounds on forgetting, while being suitable for a more realistic single-head scenario. More specifically, we formulate continual learning as a constrained optimization problem, where when learning a new task T the optimization needs to remain in the parameter region that prevents performance drop on all previous tasks $1, \dots, T-1$. In general, this problem is NP-hard ([Knoblauch et al., 2020](#)) because the shapes of the viable parameter regions are highly irregular and overlaps are hard to identify. To overcome this problem, we propose to look for the intersections of hyperrectangles which are subsets of these regions, as displayed in Figure 1. We show that finding those intersections can be done in polynomial time, as long as we trim the parameter regions to fit the consecutive hyperrectangles. The resulting training paradigm, dubbed Hyperrectangle Training, is able to learn from a sequence of tasks while putting bounds on catastrophic forgetting.

Although this simple idea is theoretically viable, finding hyperrectangles that follow these requirements and performing optimization with constrained parameter space is not trivial with the existing neural network models. To solve this problem, we propose InterContiNet, a novel neural algorithm based on interval arithmetic which we use to implement the Hyperrectangle Training. Instead of optimizing individual network parameters, we consider intervals of parameter values, ensuring that every parameter value within the interval does not lead to performance deterioration with respect to the previous tasks. More specifically, once training on task T is finished, we use the obtained intervals as the hyperrectangle constraints for training the following tasks $(T+1, T+2, \dots)$. We show how InterContiNet can be incorporated within conventional neural network building blocks such as linear layers, convolutional layers, and ReLU activations. The resulting combination of Hyperrectangle Training and InterContiNet algorithm offers competitive results on a range of continual learning tasks we evaluated.

To summarize, our main contributions are the following:

- A novel paradigm of continual learning called Hyperrectangle Training that models optimal parameter space with hyperrectangles, allowing us to control catastrophic forgetting in neural networks and set guaranteed performance. We show that with adequate parameterization we can use this paradigm to train neural networks in polynomial time.

- InterContiNet, a new algorithm that implements Hyperrectangle Training in practice, leveraging interval arithmetic of model parameters.
- Thorough experimental evaluation that confirms the validity of our method and its potential for putting guarantees on catastrophic forgetting in contemporary deep networks.

2. Related Works

Continual learning Traditionally, continual learning approaches are grouped into three general families: regularization, dynamic architectures, and replay ([Parisi et al., 2019](#); [Delange et al., 2021](#)).

Regularization-based methods extend the loss function with additional terms slowing down changes in model parameters crucial for previous tasks. These terms can be introduced individually for each task or as a single overall value representative for the whole sequence (*online* variant). Based on the type of regularization, a further subdivision splits these methods into prior- and data-focused approaches.

Prior-focused methods employ a prior on the model parameters when learning new tasks. The importance of individual parameters is estimated, i.e., with the Fisher information matrix, as shown in the seminal work on Elastic Weight Consolidation (EWC) ([Kirkpatrick et al., 2017](#)). A similar approach, Synaptic Intelligence ([Zenke et al., 2017](#)), utilizes the whole learning trajectory to compute this importance measure. In contrast, the Memory Aware Synapses technique (MAS) ([Aljundi et al., 2018](#)) approximates the importance of the parameters based on the gradients of the squared L_2 -norm of the learned function output. Our proposed method bears a certain resemblance to these methods, as it also regularizes the model’s parameters. However, we approach this problem differently by abandoning the soft quadratic penalties and instead introducing hard constraints based on the interval propagation loss. This allows us to provide guarantees on the worst-case level of forgetting.

Data-focused methods instead perform knowledge distillation from models trained on previous tasks when learning on new data. For instance, the Learning without Forgetting paradigm (LwF) ([Li & Hoiem, 2017](#)) employs an additional distillation loss based on the comparison of new task outputs generated by the new and old models.

Regularization-based methods have a number of advantages. They work without changing the structure of the model and without storing any examples of the old data, which might be crucial due to data privacy issues. However, as they only impose a soft penalty, both the prior- and data-focused regularization methods cannot entirely prevent the forgetting of previous knowledge.

Such guarantees can be provided instead by approaches with dynamic architectures, which dedicate separate model branches to different tasks. These branches can be grown progressively, such as in the case of Progressive Neural Networks (Rusu et al., 2016). Alternatively, a static architecture can be reused with iterative pruning proposed by PackNet (Mallya & Lazebnik, 2018) or by using Supermasks in Superposition (Wortsman et al., 2020). A major practical drawback of these methods is that they require the knowledge of the actual task identity during inference, which is problematic in more realistic scenarios. Therefore, due to this limitation, we do not consider them further in our analysis.

A different, very successful approach in continual learning is replaying some form of old data during incremental training to maintain the knowledge acquired in the past (Lopez-Paz & Ranzato, 2017; Shin et al., 2017; Rolnick et al., 2019). This process can use actual raw examples from the previous tasks or samples generated synthetically by a separate model. However, this paradigm is inapplicable in various applications, such as working with private medical data. Thus, in further experiments, we limit our comparisons to approaches based on regularization.

Finally, a recent work (Mirzadeh et al., 2021) addressed catastrophic forgetting by exploiting the linear connectivity between solutions obtained through multitask and continual learning in the form of Mode Connectivity SGD. Our work shares its fundamental motivation, i.e. driving the optimization process to parameter subspaces appropriate for the whole sequence of tasks. However, in contrast to Mirzadeh et al. (2021), we do not employ any kind of replay buffer, but we establish the solution boundaries based on interval arithmetic.

Interval arithmetic for neural networks In deep learning, interval arithmetic was used in three main areas. First of all, we can use interval arithmetic to deal with the situation where we have only uncertain information about the data. Chakraverty & Sahoo (2014) presented an interval artificial neural network (IANN) that can handle input and output data represented as intervals. This architecture was further adapted for viruses data with uncertainty modeled by intervals (Chakraverty & Sahoo, 2017; Sahoo & Chakraverty, 2020).

In Goyal et al. (2018); Morawiecki et al. (2019) interval arithmetic was used to produce neural networks that are robust against adversarial attacks. In such an approach authors used worst-case cross-entropy to train the classification model. In Proszewska et al. (2021), the architecture was employed for representing voxels in 3D shape modeling.

In contrast to these previous works, we apply interval arithmetic to weights instead of input data.

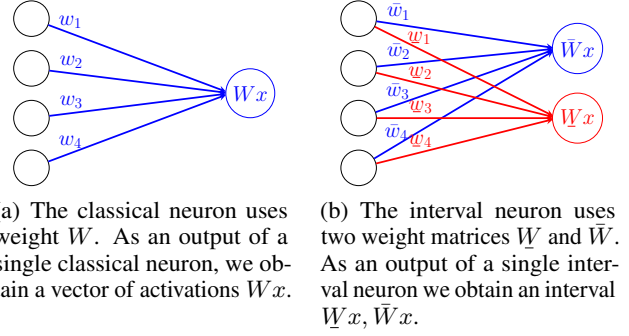


Figure 2. Comparison between the classical neuron and the interval neuron. In the interval neuron, we have two sets of weights describing the intervals.

3. Hyperrectangle Training

In this section, we will present Hyperrectangle Training for the continual learning problem. The main idea is to constrain the parameter search within the set of parameters for which any particular solution is valid for the previous tasks. In turn, we can guarantee that any parameter obtained this way will still be a valid solution for the previous task, thus putting bounds on forgetting. Although explicitly representing the region of valid solutions is not tractable, we show that using interval arithmetic we can efficiently compute the upper bound of the worst-case loss in this region to facilitate practical implementations. This general approach is presented in Figure 1.

The continual learning problem Assuming a sequence of tasks T_i for $i = 1, \dots, k$ we formulate the continual learning objective for task T_j following Chaudhry et al. (2019) as:

$$\arg \min_{\theta} \ell(T_j, \theta) \text{ satisfying } \ell(T_m, \theta) \leq \ell(T_m, \theta_m^*), \quad (1)$$

for all $m = 1, \dots, j-1$, where θ_m^* are the parameters obtained directly after learning task m and ℓ is the cross-entropy loss over the whole task dataset $\ell(T_j, \theta) = \frac{1}{|T_j|} \sum_{(x,y) \in T_j} \ell(\phi_\theta(x), y)$, where by $\phi_\theta(x)$ we denote the (pre-softmax) logits produced by the neural network ϕ with parameters θ . In other words, we would like to train our model on j -th task without reducing the performance of previous tasks $m = 1, \dots, j-1$.

Since this constraint is difficult to satisfy given the limited access to data from previous tasks, most continual learning methods transform it into a soft penalty. In particular, regularization-based continual learning methods such as EWC (Kirkpatrick et al., 2017) usually use a soft approximation of the forgetting constraint. While this approach reduces changes in weights which are considered important

for the previous task, it does not completely eliminate forgetting. Rephrasing this in the context of stability-plasticity trade-off, most methods allow for better plasticity at the cost of sometimes sacrificing stability. However, in many applications, such as medical imaging, autonomous vehicles, or robotics, even slight hints of forgetting are not acceptable. Our main goal is to create a CL method that does not allow any forgetting while maintaining good plasticity.

Hyperrectangle Training In order to fully realize the constraint in Eq. (1) we change the approach of thinking about the parameter space of the neural networks, switching the focus from finding particular points $\theta \in \mathbb{R}^D$ in the parameter space to reasoning about whole regions $\Theta \subset \mathbb{R}^D$. In other words, in the continual learning problem of optimizing over $T_1, \dots, T_n$, when training on task T_j , we are interested in finding:

$$\begin{aligned} \Theta_{j+1} = \arg \min_{\Theta} \max_{\theta \in \Theta_j} \ell(T_{j+1}, \theta) \\ \text{satisfying } \|\Theta\| > C_{j+1}, \Theta \subseteq \Theta_j \end{aligned} \quad (2)$$

where $\|\Theta\|$ denotes some notion of the size of the set, e.g. its volume, C_{j+1} is a hyperparameter denoting the minimal required size of that set, and we set $\Theta_0 = \mathbb{R}^D$ to be the whole domain of real vectors. With this formulation, since $\Theta_{j+1} \subseteq \Theta_j$ we can easily see that for any $\theta \in \Theta_{j+1}$, $\ell(T_j, \theta) \leq \max_{\theta' \in \Theta_j} \ell(T_j, \theta')$. Thus, this approach allows us to provide guarantees about the level of forgetting.

However, applying this idea directly to neural networks is intractable. In fact, as [Knoblauch et al. \(2020\)](#) noted, even finding the intersection between regions Θ_j and Θ_{j+1} of an arbitrary shape is an NP-hard problem. In order to bring the problem back to tractable regime, we try to instead find $\bar{\Theta} \subseteq \Theta$, where $\bar{\Theta}$ has a restricted shape that can be easily intersected. That is, we want to find:

$$\bar{\Theta}_j = \arg \max_{\bar{\Theta}} \|\bar{\Theta}\| \text{ satisfying } \bar{\Theta} \subseteq \Theta, \bar{\Theta} \in \mathcal{G}, \quad (3)$$

where $\mathcal{G}$ is a family of sets with required properties. For example, in order to solve the problem of NP-hard intersections we could use $\mathcal{G}$ as the set of convex polytopes. However, this restriction is not enough to make optimization of Eq. (2) tractable. Instead, we restrict this family even further, and we set $\mathcal{G}$ to be a set of hyperrectangles, which, as we will show, allows us to develop a viable algorithm for training neural networks in the continual learning setting. With this formulation, we lose some plasticity as we are not able to represent points in the region $\Theta_j \setminus \bar{\Theta}_j$, but in the experimental section we show that the final model still has enough expressiveness to solve difficult tasks.

Using this formulation, we will now show how to apply the Hyperrectangle Training to continual learning problem

with neural networks. In particular, we will demonstrate that for standard modern neural network architectures we can find a tractable and easily differentiable upper bound of $\max_{\theta \in \Theta} \ell(T_j, \theta)$ which will in turn allow us to perform optimization of Eq. (2).

Interval arithmetic for neural networks Let us consider the family of feed-forward neural networks trained for classification tasks. We assume that the neural network $\Phi(x; \theta)$ is defined as a sequence of transformations h_l for each of its L layers. The output $h_L(h_{L-1}(\dots h_1(x))) = z_L \in \mathbb{R}^N$ has N logits corresponding to N classes.

In Hyperrectangle Training we assume that weights of each transformation h_k are located arbitrarily in the hyperrectangle $[W, \bar{W}]$. For a particular task T_j , the Cartesian product of all intervals $[W_1, \bar{W}_1] \times [W_2, \bar{W}_2] \times \dots [W_L, \bar{W}_L]$ forms the hyperrectangle Θ_j . With this setting we aim to find an upper bound on $\max_{\theta \in \Theta} \ell(T_j, \theta)$.

Observe that for a given input x and particular interval layer $[W, \bar{W}]$ we can bound the possible outputs of that layer. As such, we can bound not only the weights of our model but also the activations. Thus, we will use interval arithmetic as our main tool to formalize this problem.

Interval arithmetic ([Dahlquist & Björck, 2008](#)) (Chapter 2.5.3) is based on the operations on segments. Let us assume A and B are numbers expressed as intervals. For all $\bar{a}, \underline{a}, \bar{b}, \underline{b} \in \mathbb{R}$ where $A = [\underline{a}, \bar{a}]$, $B = [\underline{b}, \bar{b}]$, we can define operations such as ([Lee, 2004](#)):

- addition: $[\underline{a}, \bar{a}] + [\underline{b}, \bar{b}] = [\underline{a} + \underline{b}, \bar{a} + \bar{b}]$
- multiplication: $[\underline{a}, \bar{a}] * [\underline{b}, \bar{b}] = [\min(\underline{a} * \underline{b}, \underline{a} * \bar{b}, \bar{a} * \underline{b}, \bar{a} * \bar{b}), \max(\underline{a} * \underline{b}, \underline{a} * \bar{b}, \bar{a} * \underline{b}, \bar{a} * \bar{b})]$

Thus, we can use interval arithmetic to perform affine transformations, which in turn allows us to implement fully-connected and convolutional layers of neural networks. Let us consider the interval version of the classical dense layer:

$$h(x) = Wx + b.$$

In Hyperrectangle Training a vector of weights is a vector of intervals and consequently the output of the dense layer is also a vector of intervals:

$$[h(x), \bar{h}(x)] = [W, \bar{W}][x, x].$$

InterContiNet is defined by a sequence of transformations h_l for each of its L layers. That is, for an input $z_0 = [x, x]$, we have

$$[z_l, \bar{z}_l] = h_l([z_{l-1}, \bar{z}_{l-1}]) \text{ for } l = 1, \dots, L.$$

The output $[z_L, \bar{z}_L]$ has N interval logits corresponding to N classes.

Propagating bounds through any element-wise monotonic activation function (e.g., ReLU, tanh, sigmoid) is trivial. Concretely, if h_l is an element-wise non-decreasing function, we have:

$$z_l = h_l(z_{l-1}), \quad \bar{z}_l = h_l(\bar{z}_{l-1}).$$

As such, we can use interval arithmetic to push any given input x through the whole network and obtain outputs of N logits intervals corresponding to N classes $[z_L, \bar{z}_L]$. In other words, we can find $z_L(x; \Theta_j) = [z_L, \bar{z}_L]$ such that $\phi(x; \theta) \in z_L(x; \Theta_j)$ for any $\theta \in \Theta_j$.

Upper bound of maximum loss over a region Thanks to the interval arithmetic, we can now find an upper-bound of $\max_{\theta \in \Theta} \ell(T_j, \theta)$ which is efficient to compute.

Theorem 3.1. *Let Φ be a neural network, Θ be a region of parameter space of that neural network, and $z_L(x, \Theta)$ be a function returning a hyperrectangle such that $\Phi(x, \theta) \in z_L(x, \Theta)$ for any $\theta \in \Theta$. Define worst-case interval-based loss as: $\hat{l}(x, y; \Theta) = l(\hat{z}_L, y)$, where $\hat{z}_L$ is a vector with each element defined as:*

$$\hat{z}_L^{(i)} = \begin{cases} \bar{z}_L^{(i)}, & \text{for } y \neq i, \\ z_L^{(y)}, & \text{otherwise.} \end{cases}$$

Then $\hat{l}(x, y; \Theta) \geq \max_{\theta \in \Theta} l(x, \theta)$.

Proof. First, we will show that that $\hat{z}_L$ is the maximum of the cross-entropy loss, i.e. $\hat{z}_L = \arg \max_{z \in z_L(x, \Theta)} l(z, y)$. From the definition of $\hat{l}$, for any $\tilde{z} \in z_L(x, \Theta)$, for the worst-case prediction of the true class (when $i = y$) we have $\tilde{z}^{(y)} \geq \hat{z}_L^{(y)}$. Similarly, for the incorrect class ($i \neq y$) we have $\tilde{z}^{(i)} \leq \hat{z}_L^{(i)}$. When we consider coordinate connected with correct label $y = i$ cross-entropy will be larger than any other elements from interval since we return a lower prediction on the correct class. Analogically, when we consider coordinate connected with an incorrect label $i \neq y$ any element larger than the minimum of the interval, we obtain lower cross-entropy since we increase the probability of false classes. From that, it is evident that $\hat{z}_L$ is the maximum argument in this hyperrectangle.

Now, denote the set of possible logits produced by the neural network over the region Θ as $\Phi(x) = \{\phi(x, \theta) \mid \theta \in \Theta\}$. Since we know that $\phi(x, \theta) \in z_L(x, \Theta)$, we see that $\Phi(x) \subseteq z_L(x, \Theta)$. Since $\hat{z}$ is the maximum over the set $z_L(x, \Theta)$, then it must also be the maximum over its subset $\Phi(x)$. $\square$

This gives us the worst-case scenario over a parameter region for a particular example. It is now straightforward to define $\hat{\ell}(T_j, \Theta) = \frac{1}{|T_j|} \sum_{(x,y) \in T_j} \hat{l}(x, y; \Theta)$ to be the worst-case scenario for the whole task. Now we can prove that

this $\hat{\ell}$ is an upper bound of the whole continual learning objective.

Theorem 3.2. *Assume a sequential learning problem described by (2). Then during task j for any $k < j$, we have $\hat{\ell}(T_k, \Theta_j) \geq \max_{\theta' \in \Theta_j} \ell(T_k, \theta') \geq \max_{\theta' \in \Theta_k} \ell(T_k, \theta')$*

Proof. The first inequality follows from Theorem 3.1:

$$\begin{aligned} \hat{\ell}(T_j, \Theta) &= \frac{1}{|T_j|} \sum_{(x,y) \in T_j} \hat{l}(x, y; \Theta) \\ &\geq \frac{1}{|T_j|} \sum_{(x,y) \in T_j} \max_{\theta \in \Theta} l(x, y; \theta) \\ &\geq \max_{\theta \in \Theta} \frac{1}{|T_j|} \sum_{(x,y) \in T_j} l(x, y; \theta) \\ &= \max_{\theta \in \Theta} \ell(T_j, \theta) \end{aligned} \quad (4)$$

The second inequality follows directly from the fact that $\Theta_j \subseteq \Theta_k$. $\square$

Analogously to $\hat{\ell}$, we can define $\hat{\text{acc}}(T_j, \Theta_j)$ to be the interval lower bound on minimum accuracy for task T_j over a region of parameter space Θ_j , i.e. $\hat{\text{acc}}(T_j, \Theta_j) \leq \min_{\theta \in \Theta_j} \text{acc}(T_j, \theta)$. Since for each $k > j$, $\Theta_k \subseteq \Theta_j$, we can guarantee that throughout the training the accuracy for j -th task will not fall below $\hat{\text{acc}}(T_j, \Theta_j)$.

To summarize, we have shown that by using Hyperrectangle Training we obtain guarantees on non-forgetting since the interval from task j is inside segments from the previous $j - 1$ -th task. Moreover, we can solve such a problem in polynomial time, since we restrict our consideration to hyperrectangles. Finally, in Theorem 3.2 we demonstrate that the bounds on logits provided by interval arithmetic on neural networks can be used to calculate an efficient upper bound on the whole continual learning objective.

The result presented here is relevant to [Knoblauch et al. \(2020\)](#) who have shown that classical continual learning problem is in general NP-hard because of the highly irregular shapes of the viable regions of parameters. In Section B we present a more rigorous treatment of the above derivation, using the setting and assumptions proposed in [Knoblauch et al. \(2020\)](#).

4. InterContiNet

In the previous section, we have shown that by considering the intervals of activations in a neural network we can derive an upper bound of the maximum loss over a hyperrectangle. Now, we will present a model which minimizes this upper

Algorithm 1 InterContiNet training procedure for a given task

```

Input: model trained on the previous task (weights  $W^*$ ,
radii  $\varepsilon^*$ ), current task  $T_j$ .
Reparameterize  $W, \varepsilon$  using Eq. (5).
for epoch in  $1 \dots \text{center\_epochs}$  do
    Update  $\mu$  by minimizing  $\ell(T_j, \Theta)$     {Train centers}
end for
Initialize  $\varepsilon$  as largest possible within the previous interval.
for epoch in  $1 \dots \text{radii\_epochs}$  do
    Update  $\nu$  by minimizing  $\hat{\ell}(T_j, \Theta)$     {Train radii}
    if  $\text{acc} \geq \text{acc} \cdot \text{acc\_thresh}$  then
        break
    end if
end for
return  $W, \varepsilon$ 
    
```

bound in order to approximate the optimization presented in Eq. (2).

Parametrization We start by taking a standard feed-forward neural network Φ with parameters θ , but instead of considering points in the parameter space, we consider regions Θ . In particular, we describe the k -th parameter of the network with the center W_k and the interval radius ε_k . Thus, we consider the region:

$$[W_k, \bar{W}_k] = [W_k - \varepsilon_k, W_k + \varepsilon_k],$$

With this formulation, we can still use this network as a standard non-interval model by only using the center weights W_k , which will, in turn, produce only the center activations for each layer. At the same time, we can also keep track of the interval $[\underline{z}_l, \bar{z}_l]$ by using the interval arithmetic. These operations are independent of each other, so as explained in [Gowal et al. \(2018\)](#) we can implement them efficiently for GPUs by calculating the center prediction and the intervals in parallel. In practice, we reimplement the basic blocks of neural networks (fully-connected layers, convolutional layers, activations, pooling, etc.) in order to handle interval inputs and interval weights.

With this formulation, we are able to implement most operations in contemporary standard neural network architectures. However, one important exception is batch normalization which uses running statistics of activations in inference mode. Since it is impossible to foresee how these statistics will change in the future, providing reliable forgetting guarantees with interval arithmetic is infeasible. As such, we restrict ourselves to simpler architectures without batch normalization.

During the training of the second and every subsequent task, we need to make sure that the intervals for the current task are contained within intervals from the previous task.

Starting from the second task, we keep the previous interval centers W_k^* and radii ε_k^* . Then, we reparameterize the training as:

$$\begin{aligned} W_k &= W_k^* + \tanh(\mu_k) \cdot \varepsilon_k^* \\ \varepsilon_k &= \sigma(\nu_k) \cdot \min((W_k^* + \varepsilon_k^*) - W_k, \\ &\quad W_k - (W_k^* - \varepsilon_k^*)). \end{aligned} \quad (5)$$

Observe that we cannot simply set the radius to be $\sigma(\nu_k) \cdot \varepsilon_k^*$ as depending on the position on the center W_k the interval $[W_k - \varepsilon_k, W_k + \varepsilon_k]$ might not be fully contained inside the previous task interval $[W_k^* - \varepsilon_k^*, W_k^* + \varepsilon_k^*]$. We train μ_k and ν_k instead of directly optimizing W_k and ε_k .

Training When training InterContiNet in a continual learning setting, we divide training of each task T_j into two phases. In the first phase, we focus on finding the interval centers W_k and keep the radii ε_k frozen. We do this by simply minimizing the cross-entropy loss, as done in standard classification neural network training. In the second phase, we freeze the centers W_k and initialize ε_k with the maximal value that still fits within the previous interval¹. Then, we minimize the upper-bound loss $\hat{\ell}(T_j, \Theta_j)$ by optimizing ε_k . Optimizing this upper bound indefinitely would lead to minimizing ε_k to zeroes and producing degenerated intervals. To prevent this, we introduce an interpretable hyperparameter `acc_thresh` which allows us to choose the fraction of examples in the current task that is guaranteed to be classified correctly for the rest of the training. For that purpose, we use the worst-case accuracy `acc` described in Section 3 and minimize the worst-case loss $\hat{\ell}$ until the condition $\text{acc} \geq \text{acc} \cdot \text{acc_thresh}$ is satisfied, with `acc` being the train accuracy on the current task. We use running averages on the last few batches to efficiently estimate `acc` and `acc`. As such, `acc_thresh` is the variable that controls the stability-plasticity trade-off in our model. A high value of `acc_thresh` will guarantee less forgetting in the end, but will also result in a smaller parameter region Θ_j , which will be used to constrain the parameter search in the next task. Algorithm 1 shows the whole training procedure for a single task.

Analysis We believe that realistic computational and memory constraints are an important part of the continual learning setting. In terms of the memory requirements, InterContiNet needs to remember the centers W_K^* and radii ε_k^* from the previous task. Thus, assuming a network with D parameters, we need to keep additional $2 \cdot D$ values in memory. This is the same memory constraint as for most of our baselines, e.g. online EWC, which remembers the weights and their importance for the last task.

¹For the first task the maximum radius is a hyperparameter to be set.

In terms of computational constraints, propagating both activation centers and intervals through the network requires approximately 3 times as many FLOPs as propagating just a single point. However, the main part of the training is the center optimization phase, that has no overhead as compared to the base network since we only need to push forward a single point (the centers). The interval propagation is only needed in the radii optimization phase that is a small part of the overall training. Thus, in practice it is on a similar order of complexity as, e.g. computing the Fisher information matrix in EWC. Finally, in practice even in the interval propagation phase the training time does not change significantly, as these operations are automatically parallelized on GPUs when using frameworks such as PyTorch.

5. Experiments

To verify the empirical usefulness of our method, we test it in three standard continual learning scenarios (Hsu et al., 2018; van de Ven & Tolias, 2019): incremental task, incremental domain, and incremental class. In the incremental task setting, we create a separate output head for each task and only train a single head per task along with the base network. During inference, the head appropriate for the given task is selected. The incremental domain and incremental class scenarios are more challenging, as they only use a single head and the task identity is not provided during inference. In our case, this single head is created with either 2 (incremental domain) or 10 output classes (incremental class).

We run the experiments on four datasets commonly used for continual learning: MNIST, FashionMNIST, CIFAR-10 and CIFAR-100 split into sequences of tasks. For MNIST and FashionMNIST, we use a standard MLP architecture with 2 hidden layers of size 400 as previously evaluated in (Hsu et al., 2018). For CIFAR-10 and CIFAR-100 we use the fairly simple AlexNet architecture (Krizhevsky et al., 2012). This choice is dictated by the infeasibility of implementing interval version of batch normalization which is present in most standard convolutional network architectures. In our experiments, we use the Avalanche (Lomonaco et al., 2021) library to facilitate easier and fairer comparisons. The code is available at <https://github.com/gmum/InterContiNet>.

We compare InterContiNet with a number of regularization-based approaches typically used in continual learning settings, i.e. EWC, Synaptic Intelligence, MAS, and LwF. As additional baselines, we present vanilla sequential training with SGD and Adam, and a simple L_2 regularization between new and old model parameters (i.e. all parameters have the same importance). Further training details are available in Appendix A.

Table 1. The average accuracy across all five tasks of the split MNIST protocol, evaluated after learning the whole sequence. Each value is the average of five runs (with standard deviations).

Method	Incremental task	Incremental domain	Incremental class
SGD	96.27 $\pm$ 0.38	64.58 $\pm$ 0.26	19.01 $\pm$ 0.04
Adam	95.53 $\pm$ 3.16	59.32 $\pm$ 1.08	19.74 $\pm$ 0.01
L2	96.31 $\pm$ 0.41	72.21 $\pm$ 0.17	18.88 $\pm$ 0.18
EWC	97.01 $\pm$ 0.13	76.90 $\pm$ 0.41	18.90 $\pm$ 0.06
oEWC	97.01 $\pm$ 0.13	77.02 $\pm$ 0.51	18.89 $\pm$ 0.07
SI	96.19 $\pm$ 0.63	80.62 $\pm$ 0.17	17.94 $\pm$ 0.57
MAS	96.52 $\pm$ 0.14	84.41 $\pm$ 0.42	17.38 $\pm$ 4.19
LwF	97.03 $\pm$ 0.05	82.76 $\pm$ 0.17	49.37 $\pm$ 0.68
InterContiNet	98.93 $\pm$ 0.05	77.77 $\pm$ 1.24	40.73 $\pm$ 3.26
Offline	99.74 $\pm$ 0.03	99.03 $\pm$ 0.04	98.49 $\pm$ 0.02

Table 2. The average accuracy across all five tasks of the split FashionMNIST protocol, evaluated after learning the whole sequence. Each value is the average of five runs (with standard deviations).

Method	Incremental task	Incremental domain	Incremental class
SGD	92.22 $\pm$ 3.06	82.77 $\pm$ 0.44	19.91 $\pm$ 0.01
Adam	88.13 $\pm$ 5.64	78.48 $\pm$ 0.47	19.96 $\pm$ 0.01
L2	97.36 $\pm$ 0.17	92.65 $\pm$ 0.09	26.91 $\pm$ 1.23
EWC	97.53 $\pm$ 0.15	92.12 $\pm$ 0.18	19.90 $\pm$ 0.01
oEWC	96.70 $\pm$ 0.57	88.83 $\pm$ 0.30	19.87 $\pm$ 0.02
SI	97.00 $\pm$ 0.25	91.45 $\pm$ 0.07	19.97 $\pm$ 0.34
MAS	97.43 $\pm$ 0.14	91.74 $\pm$ 0.19	10.00 $\pm$ 0.00
LwF	98.10 $\pm$ 0.07	88.63 $\pm$ 0.12	39.51 $\pm$ 1.45
InterContiNet	98.37 $\pm$ 0.06	92.65 $\pm$ 0.40	35.11 $\pm$ 0.02
Offline	97.98 $\pm$ 0.05	96.39 $\pm$ 0.06	82.54 $\pm$ 0.13

5.1. MNIST and CIFAR Benchmarks

Results presented in Tables 1 and 2 indicate that in the incremental task setup InterContiNet has better average accuracy across all five tasks than all the other continual learning baselines. We hypothesize that a multi-head setup facilitates our 2-phase training procedure, as the output heads in the last layer are independent and do not have to be constrained inside intervals. In the incremental domain setup, our method has a similar performance to EWC. However, the advantage of InterContiNet is that we can guarantee a maximal level of forgetting on the past tasks when training on new data.

The most significant difference is visible in the incremental class scenario, where LwF and InterContiNet perform much better than typical regularization-based approaches. We can partly explain this behavior with the particular setup, where the final number of output classes is known at the onset of the experiment. While such information induces an indirect regularization effect on the softmax layer in LwF and InterContiNet, it has smaller effect on other techniques. This

Table 3. The average accuracy across all five tasks of Split CIFAR-10, evaluated after training AlexNet on the whole sequence. Each value is the average of five runs (with standard deviations).

Method	Incremental task	Incremental domain	Incremental class
SGD	64.74 $\pm$ 8.53	69.22 $\pm$ 0.55	15.56 $\pm$ 5.07
EWC	67.33 $\pm$ 7.11	70.26 $\pm$ 0.66	11.83 $\pm$ 4.10
oEWC	64.59 $\pm$ 7.97	65.97 $\pm$ 8.97	15.49 $\pm$ 5.01
SI	67.26 $\pm$ 7.77	69.69 $\pm$ 0.73	17.38 $\pm$ 4.13
MAS	66.20 $\pm$ 9.29	72.54 $\pm$ 0.59	11.79 $\pm$ 4.01
LwF	93.03 $\pm$ 0.28	76.97 $\pm$ 0.91	13.89 $\pm$ 5.33
InterContiNet	72.64 $\pm$ 1.18	69.48 $\pm$ 1.36	19.07 $\pm$ 0.15
Offline	93.11 $\pm$ 0.18	90.54 $\pm$ 0.32	82.72 $\pm$ 0.09

dichotomy highlights the fundamental difference between LwF’s knowledge distillation approach and all the other baselines, which are prior-focused.

Finally, we extend our investigation to CIFAR-10 with the AlexNet. We choose this architecture as an example of a convolutional network without batch normalization, which is problematic in our setting, as we explain later in Section 6. As shown in Table 3, in this setting InterContiNet still performs comparably to other parameter regularization methods. At the same time, we observe that LwF tends to significantly outperform other methods, especially in the incremental task setting. We hypothesize that the regularization through functional penalties (KL divergence between outputs) rather than direct parameter regularization works much better for this particular problem (Oren & Wolf, 2021). In the end, we show that InterContiNet is able to perform similarly to other methods in the same family while providing guarantees.

5.2. Analysis of the Parameter Region Size

Although the stability-plasticity dilemma is a crucial problem in all CL methods, in InterContiNet its impact can be directly observed by investigating the region of valid parameters Θ . The bigger the region, the higher the plasticity, as we are able to use more parameter combinations and thus represent more functions. At the same time, in order to achieve stability and get guarantees on forgetting, we need to restrict the size of this region. Here, we investigate the question of whether the interval training will at some point lead to an area of zero volume, resulting in a state without any plasticity and effectively stopping the learning².

In practice, we control the pace at which the volume decreases with hyperparameters, e.g. by choosing the worst-case accuracy threshold – if we set this threshold high, the

²We note that this risk also occurs for other CL methods (e.g. accumulating high EWC penalties which prohibit learning or reducing the percentage of trainable parameters in PackNet to zero).

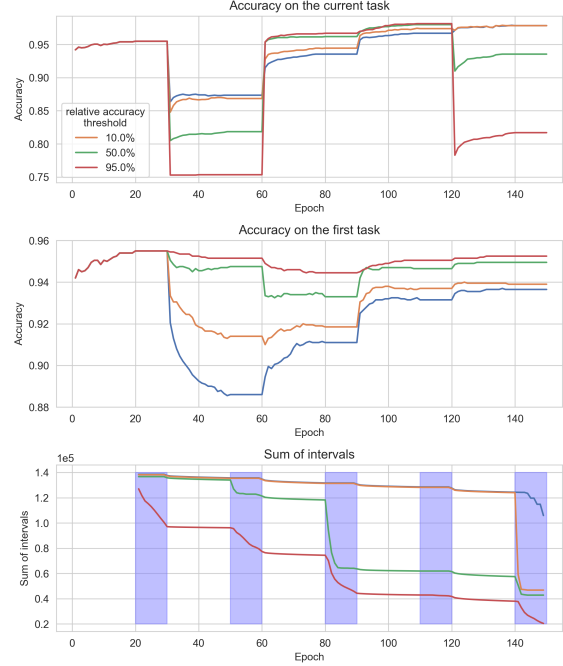


Figure 3. InterContiNet trained on split-Fashion MNIST in the incremental domain setting, with different values of the `acc_thresh` hyperparameter. We show accuracy on the current task (top, representing plasticity) and the first task (middle, representing stability) to show how the hyperparameter choices affect the final results. Additionally, the bottom plot shows how the valid parameter region changes throughout the training – highlighted areas denote the radii optimization phase, which shrinks this region most effectively.

region collapses quicker. Here, we conduct additional studies to better understand how the threshold hyperparameter impacts the results and the dynamics of the region of valid parameters. In Figure 3 we present the accuracy curves for training InterContiNet on FashionMNIST in the incremental domain setting, using different values of the accuracy threshold hyperparameter. This threshold controls the percentage of the correctly classified examples that we guarantee will not be forgotten throughout the training. For example, in Figure 3 the method obtains 95.3% accuracy at the end of the first task. If we set `acc_thresh` = 95%, then we can guarantee that at no point during training the accuracy on the first task will be lower than $95\% \cdot 95.3\% \approx 90.5\%$.

We see that this hyperparameter indeed controls the plasticity-stability trade-off, with larger values contributing to less forgetting, but also less plasticity on the subsequent tasks. Note that in practice the obtained accuracy is much higher than the one guaranteed by the worst-case accuracy. For example, even with `acc_thresh` = 0.05 we are able to maintain first-task accuracy above 90% through the whole training. However, this performance is not guaranteed and might deteriorate with further training.

In the bottom part of Figure 3, we plot how the valid pa-

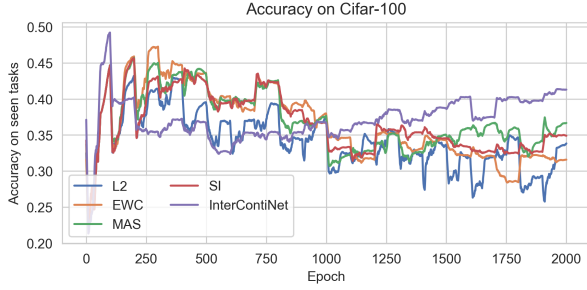


Figure 4. Results for parameter regularization methods for split-CIFAR-100 with 20 tasks, averaged over 3 seeds. InterContiNet is still able to learn new tasks at the end of the training.

parameter region changes throughout the training. Since the volume can get easily degenerated (e.g. if we prohibit even one parameter from changing), we summarize it by reporting the sum of all radii. Confirming the intuition, the valid region for models with higher `acc_thresh` shrinks much faster, while for lower thresholds the impact is not as noticeable. In the end, we observe that even in the most restrictive setting, there is still room for learning.

Finally, we extend the experiments on the collapse of the valid parameter region to a setting with a higher number of tasks. We use split-CIFAR-100 with 20 tasks of 5 classes and focus on the incremental task scenario. The results in Figure 4 show that InterContiNet performs on similar scale as other parameter regularization methods and is still able to learn new tasks at the end of the training. We consider this as additional evidence that interval constraints do not necessarily lead to faster plasticity collapse than soft regularization. Additionally, we ran LwF in this setting and found out that it significantly outperforms the parameter regularization methods by a very wide margin. We show the additional results in Figure 5 in Appendix A. This finding is consistent with the split-CIFAR-10 results and recent analysis (Oren & Wolf, 2021), and is a strong motivation to consider functional regularization in future work.

6. Limitations & Future Work

Although reformulating continual learning as a sequential contraction of the model’s parameter space creates a viable theoretical framework, InterContiNet has several limitations that would require further research to be addressed.

To begin with, interval arithmetic cannot be easily combined with batch normalization layers. As we cannot foresee the running statistics of activations at future inference, we cannot guarantee the bounds of the model’s output. This limitation is problematic due to the ubiquitousness of batch normalization in contemporary neural network models. We plan to evaluate InterContiNet with different normalization

schemes (e.g. layer (Ba et al., 2016) or group normalization (Wu & He, 2020)), where providing guarantees is still possible. Another option worth exploring is freezing the batch normalization statistics after the first task.

On the other hand, the no-forgetting guarantees are also limited in several ways. Firstly, we concentrate on the worst-case scenario, i.e. we maintain that the worst solution in the allowed parameter space will not get even worse. In practice, we evaluate the model at the interval centers, which can leave a lot of leeway for the accuracy to fluctuate above the defined threshold. Secondly, these guarantees are computed on the training set. Depending on the amount of training–test distribution shift, a non-negligible change in the model’s response can be unaccounted for in our worst-case scenario.

Another drawback of hard constraints is the risk of reducing the viable parameter space to a zero-volume region when exposed to very long sequences of tasks or very deep architectures. While it is a very reasonable concern, the goal of our empirical evaluation was to show that InterContiNet’s parameter space does not degenerate so rapidly in typical experimental setups. Nevertheless, based on Theorem 3.1, it is theoretically possible to devise a malicious dataset that would result in arbitrarily large forgetting or complete loss of plasticity. Lastly, our evaluation is limited to classification problems only. However, extensions to other types of problems (e.g. regression) could be made with simple adjustments to the loss function.

To summarize, in this work we proposed Hyperrectangle Training, a general approach to continual learning. We constrain the parameter search within the hyperrectangle representing the region of valid solutions for the previous task. We show that by applying interval arithmetic to operations present in standard neural networks in this setting we can derive a tractable and differentiable upper bound of the CL objective. Then we proposed and empirically evaluated InterContiNet, a CL algorithm that minimizes this upper bound to set hard limits on forgetting.

7. Acknowledgements

The work of M. Wołczyk, K. Piczak, J. Tabor, and T. Trzciński was supported by Foundation for Polish Science (grant no POIR.04.04.00-00-14DE/18-00) carried out within the Team-Net program co-financed by the European Union under the European Regional Development Fund. The work of P. Spurek was supported by the National Centre of Science (Poland) Grant No. 2019/33/B/ST6/00894. The work of T. Trzciński was supported by the National Centre of Science (Poland) Grant No. 2020/39/B/ST6/01511. The work of B. Wójcik was supported by the Priority Research Area DigiWorld under the program Excellence Initiative - Research University at the Jagiellonian University in Kraków.

References

- Aljundi, R., Babiloni, F., Elhoseiny, M., Rohrbach, M., and Tuytelaars, T. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.
- Ba, L. J., Kiros, J. R., and Hinton, G. E. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.
- Chakraverty, S. and Sahoo, D. M. Interval response data based system identification of multi storey shear buildings using interval neural network modelling. *Computer Assisted Methods in Engineering and Science*, 21(2):123–140, 2014.
- Chakraverty, S. and Sahoo, D. M. Novel transformation-based response prediction of shear building using interval neural network. *Journal of Earth System Science*, 126(3): 32, 2017.
- Chaudhry, A., Ranzato, M., Rohrbach, M., and Elhoseiny, M. Efficient lifelong learning with A-GEM. In *Proceedings of the 7th International Conference on Learning Representations (ICLR)*, 2019.
- Dahlquist, G. and Björck, Å. *Numerical methods in scientific computing, volume I*. SIAM, 2008.
- Delange, M., Aljundi, R., Masana, M., Parisot, S., Jia, X., Leonardis, A., Slabaugh, G., and Tuytelaars, T. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.
- French, R. M. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135, 1999.
- Goodfellow, I. J., Mirza, M., Da, X., Courville, A. C., and Bengio, Y. An empirical investigation of catastrophic forgetting in gradient-based neural networks. In Bengio, Y. and LeCun, Y. (eds.), *Proceedings of the 2nd International Conference on Learning Representations (ICLR)*, 2014.
- Gowal, S., Dvijotham, K., Stanforth, R., Bunel, R., Qin, C., Uesato, J., Arandjelovic, R., Mann, T., and Kohli, P. On the effectiveness of interval bound propagation for training verifiably robust models. *arXiv preprint arXiv:1810.12715*, 2018.
- Hsu, Y., Liu, Y., and Kira, Z. Re-evaluating continual learning scenarios: A categorization and case for strong baselines. *arXiv preprint arXiv:1810.12488*, 2018.
- Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017.
- Knoblauch, J., Husain, H., and Diethe, T. Optimal continual learning has perfect memory and is NP-hard. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, 2020.
- Krizhevsky, A., Sutskever, I., and Hinton, G. E. ImageNet classification with deep convolutional neural networks. *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2012.
- Lee, K. H. *First course on fuzzy theory and applications*, volume 27. Springer Science & Business Media, 2004.
- Lee, S., Kim, J., Jun, J., Ha, J., and Zhang, B. Overcoming catastrophic forgetting by incremental moment matching. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2017.
- Li, Z. and Hoiem, D. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947, 2017.
- Liu, X., Masana, M., Herranz, L., van de Weijer, J., López, A. M., and Bagdanov, A. D. Rotate your networks: Better weight consolidation and less catastrophic forgetting. *Proceedings of the 24th International Conference on Pattern Recognition (ICPR)*, 2018.
- Lomonaco, V., Pellegrini, L., Cossu, A., Carta, A., Graffieti, G., Hayes, T. L., Lange, M. D., Masana, M., Pomponi, J., van de Ven, G., Mundt, M., She, Q., Cooper, K., Forest, J., Belouadah, E., Calderara, S., Parisi, G. I., Cuzzolin, F., Tolias, A., Scardapane, S., Antiga, L., Amhad, S., Popescu, A., Kanan, C., van de Weijer, J., Tuytelaars, T., Bacciu, D., and Maltoni, D. Avalanche: an end-to-end library for continual learning. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2nd Continual Learning in Computer Vision Workshop, 2021.
- Lopez-Paz, D. and Ranzato, M. Gradient episodic memory for continual learning. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2017.
- Mallya, A. and Lazebnik, S. Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.
- Mallya, A., Davis, D., and Lazebnik, S. Piggyback: Adapting a single network to multiple tasks by learning to mask weights. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.
- Masse, N. Y., Grant, G. D., and Freedman, D. J. Alleviating catastrophic forgetting using context-dependent gating and synaptic stabilization. *Proceedings of the National Academy of Sciences*, 115:E10467 – E10475, 2018.

- McCloskey, M. and Cohen, N. J. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pp. 109–165. Elsevier, 1989.
- Mirzadeh, S., Farajtabar, M., Görür, D., Pascanu, R., and Ghasemzadeh, H. Linear mode connectivity in multitask and continual learning. In *Proceedings of the 9th International Conference on Learning Representations (ICLR)*, 2021.
- Morawiecki, P., Spurek, P., Śmieja, M., and Tabor, J. Fast and stable interval bounds propagation for training verifiably robust models. *arXiv preprint arXiv:1906.00628*, 2019.
- Oren, G. and Wolf, L. In defense of the learning without forgetting for task incremental learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, 2021.
- Parisi, G. I., Kemker, R., Part, J. L., Kanan, C., and Wermter, S. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, 2019.
- Proszewska, M., Mazur, M., Trzciński, T., and Spurek, P. HyperCube: Implicit field representations of voxelized 3D models. *arXiv preprint arXiv:2110.05770*, 2021.
- Ratcliff, R. Connectionist models of recognition memory: constraints imposed by learning and forgetting functions. *Psychological review*, 97(2):285, 1990.
- Rolnick, D., Ahuja, A., Schwarz, J., Lillicrap, T. P., and Wayne, G. Experience replay for continual learning. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2019.
- Rusu, A. A., Rabinowitz, N. C., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., Pascanu, R., and Hadsell, R. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.
- Sahoo, D. M. and Chakraverty, S. Structural parameter identification using interval functional link neural network. In *Recent Trends in Wave Mechanics and Vibrations*, pp. 139–150. Springer, 2020.
- Shin, H., Lee, J. K., Kim, J., and Kim, J. Continual learning with deep generative replay. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2017.
- van de Ven, G. M. and Tolias, A. S. Three scenarios for continual learning. *arXiv preprint arXiv:1904.07734*, 2019.
- van de Ven, G. M., Siegelmann, H. T., and Tolias, A. S. Brain-inspired replay for continual learning with artificial neural networks. *Nature Communications*, 11, 2020.
- Wortsman, M., Ramanujan, V., Liu, R., Kembhavi, A., Rastegari, M., Yosinski, J., and Farhadi, A. Supermasks in superposition. *arXiv preprint arXiv:2006.14769*, 2020.
- Wu, Y. and He, K. Group normalization. *International Journal of Computer Vision (IJCV)*, 128(3):742–755, 2020.
- Zenke, F., Poole, B., and Ganguli, S. Continual learning through synaptic intelligence. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 2017.

Zero Time Waste: Recycling Predictions in Early Exit Neural Networks

Maciej Wołczyk*[†]
Jagiellonian University

Bartosz Wójcik*
Jagiellonian University

Klaudia Bałazy
Jagiellonian University

Igor Podolak
Jagiellonian University

Jacek Tabor
Jagiellonian University

Marek Śmieja
Jagiellonian University

Tomasz Trzcíński
Jagiellonian University,
Warsaw University of Technology,
Tooploox

Abstract

The problem of reducing processing time of large deep learning models is a fundamental challenge in many real-world applications. Early exit methods strive towards this goal by attaching additional Internal Classifiers (ICs) to intermediate layers of a neural network. ICs can quickly return predictions for easy examples and, as a result, reduce the average inference time of the whole model. However, if a particular IC does not decide to return an answer early, its predictions are discarded, with its computations effectively being wasted. To solve this issue, we introduce Zero Time Waste (ZTW), a novel approach in which each IC reuses predictions returned by its predecessors by (1) adding direct connections between ICs and (2) combining previous outputs in an ensemble-like manner. We conduct extensive experiments across various datasets and architectures to demonstrate that ZTW achieves a significantly better accuracy vs. inference time trade-off than other recently proposed early exit methods.

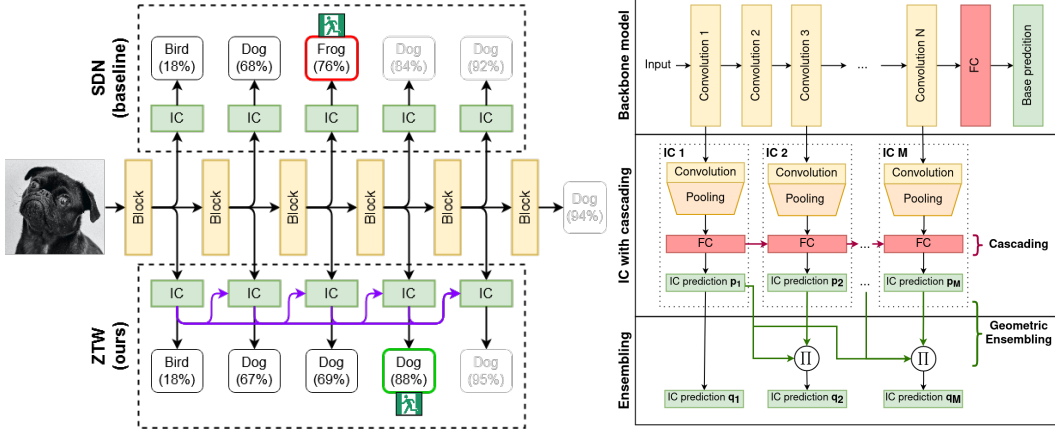
1 Introduction

Deep learning models achieve tremendous successes across a multitude of tasks, yet their training and inference often yield high computational costs and long processing times [11, 22]. For some applications, however, efficiency remains a critical challenge, *e.g.* to deploy a reinforcement learning (RL) system in production the policy inference must be done in real-time [7], while the robot performances suffer from the delay between measuring a system state and acting upon it [34]. Similarly, long inference latency in autonomous cars could impact its ability to control the speed [13] and lead to accidents [10, 17].

Typical approaches to reducing the processing complexity of neural networks in latency-critical applications include compressing the model [24, 26, 46] or approximating its responses [21]. For instance, Livne & Cohen [26] propose to compress a RL model by policy pruning, while Kouris et al. [21] approximate the responses of LSTM-based modules in self-driving cars to accelerate their inference time. While those methods improve processing efficiency, they still require samples to pass

*equal contribution

[†]Corresponding author: maciej.wolczyk@doctoral.uj.edu.pl



(a) Comparison of the proposed ZTW (bottom) with a conventional early-exit model, SDN (top). (b) Detailed scheme of the proposed ZTW model architecture.

Figure 1: (a) In both approaches, internal classifiers (ICs) attached to the intermediate hidden layers of the base network allow us to return predictions quickly for examples that are easy to process. While SDN discards predictions of uncertain ICs (*e.g.* below a threshold of 75%), ZTW reuses computations from all previous ICs, which prevents information loss and waste of computational resources. (b) Backbone network f_θ lends its hidden layer activations to ICs, which share inferred information using cascade connections (red horizontal arrows in the middle row) and give predictions p_m . The inferred predictions are combined using ensembling (bottom row) giving q_m .

through the entire model. In contrast, biological neural networks leverage simple heuristics to speed up decision making, *e.g.* by shortening the processing path even in case of complex tasks [1, 9, 18].

This observation led a way to the inception of the so-called *early exit* methods, such as Shallow-Deep Networks (SDN) [19] and Patience-based Early Exit (PBEE) [47], that attach simple classification heads, called internal classifiers (ICs), to selected hidden layers of neural models to shorten the processing time. If the prediction confidence of a given IC is sufficiently high, the response is returned, otherwise, the example is passed to the subsequent classifier. Although these models achieve promising results, they discard the response returned by early ICs in the evaluation of the next IC, disregarding potentially valuable information, *e.g.* decision confidence, and wasting computational effort already incurred.

Motivated by the above observation, we postulate to look at the problem of neural model processing efficiency from the *information recycling* perspective and introduce a new family of *zero waste models*. More specifically, we investigate how information available at different layers of neural models can contribute to the decision process of the entire model. To that end, we propose Zero Time Waste (ZTW), a method for an intelligent aggregation of the information from previous ICs. A high-level view of our model is given in Figure 1. Our approach relies on combining ideas from networks with skip connections [41], gradient boosting [3], and ensemble learning [8, 23]. Skip connections between subsequent ICs (which we call *cascade connections*) allow us to explicitly pass the information contained within low-level features to the deeper classifier, which forms a cascading structure of ICs. In consequence, each IC improves on the prediction of previous ICs, as in gradient boosting, instead of generating them from scratch. To give the opportunity for every IC to explicitly reuse predictions of all previous ICs, we additionally build an ensemble of shallow ICs.

We evaluate our approach on standard classification benchmarks, such as CIFAR-100 and ImageNet, as well as on the more latency-critical applications, such as reinforcement-learned models for interacting with sequential environments. To the best of our knowledge, we are the first to show that early exit methods can be used for cutting computational waste in a reinforcement learning setting.

Results show that ZTW is able to save much more computation while preserving accuracy than current state-of-the-art early exit methods. In order to better understand where the improvements come from, we introduce Hindsight Improvability, a metric for measuring how efficiently the model

reuses information from the past. We provide ablation studies and additional analysis of the proposed method in the Appendix.

To summarize, the contributions of our work are the following:

- We introduce a family of zero waste models that quantify neural network efficiency with the Hindsight Improvability metrics.
- We propose an instance of zero waste models dubbed Zero Time Waste (ZTW) method which uses cascade connections and ensembling to reuse the responses of previous ICs for the final decision.
- We show how the state-of-the-art performance of ZTW in the supervised learning scenario generalizes to reinforcement learning.

2 Related Work

The drive towards reducing computational waste in deep learning literature has so far focused on reducing the inference time. Numerous approaches for accelerating deep learning models focus on building more efficient architectures [15], reducing the number of parameters [12] or distilling knowledge to smaller networks [14]. Thus, they decrease inference time by reducing the overall complexity of the model instead of using the conditional computation framework of adapting computational effort to each example. As such we find them orthogonal to the main ideas of our work, e.g. we show that applying our method to architectures designed for efficiency, such as MobileNet [15], leads to even further acceleration. Hence, we focus here on methods that adaptively set the inference time for each example.

Conditional Computation Conditional computation was first proposed for deep neural networks in Bengio et al. [2] and Davis & Arel [5], and since then many sophisticated methods have been proposed in this field, including dynamic routing [27], cascading with multiple networks [40] and skipping intermediate layers [41] or channels [42]. In this work, we focus on the family of early exit approaches, as they usually do not require special assumptions about the underlying architecture of the network and the training paradigm, and because of that can be easily applied to many commonly used architectures. In BranchyNet [38] a loss function consisting of a weighted sum of individual head losses is utilized in training, and entropy of the head prediction is used for the early exit criterion. Berestizshevsky & Guy [4] propose to use confidence (maximum of the softmax output) instead. A broader overview of early exit methods is available in Scardapane et al. [32].

Several works proposed specialized architectures for conditional computation which allow for multi-scale feature processing [16, 44, 43], and developed techniques to train them more efficiently by passing information through the network [29, 25]. However, in this paper, we consider the case of increasing inference speed of a pre-trained network based on an architecture which was not built with conditional computation or even efficiency in mind. We argue that this is a practical use case, as this approach can be used to a wider array of models. As such, we do not compare with these methods directly.

Shallow-Deep Networks (SDN) [19] is a conceptually simple yet effective method, where the comparison of confidence with a fixed threshold is used as the exit criterion. The authors attach internal classifiers to layers selected based on the number of compute operations needed to reach them. The answer of each head is independent of the answers of the previous heads, although in a separate experiment the authors analyze the measure of disagreement between the predictions of final and intermediate heads.

Zhou et al. [47] propose Patience-based Early Exit (PBEE) method, which terminates inference after t consecutive unchanged answers, and show that it outperforms SDN on a range of NLP tasks. The idea of checking for agreement in preceding ICs is connected to our approach of reusing information from the past. However, we find that applying PBEE in our setting does not always work better than SDN. Additionally, in the experiments from the original work, PBEE was trained simultaneously along with the base network, thus making it impossible to preserve the original pre-trained model.

Ensembles Ensembling is typically used to improve the accuracy of machine learning models [6]. Lakshminarayanan et al. [23] showed that it also greatly improves calibration of deep neural networks. There were several attempts to create an ensemble from different layers of a network. Scardapane et

al. [31] adaptively exploit outputs of all internal classifiers, albeit not in a conditional computation context. Phuong & Lampert [29] used averaged answers of heads up to the current head for anytime-prediction, where the computational budget is unknown. Besides the method being much more basic, their setup is notably different from ours, as it assumes the same computational budget for all samples no matter how difficult the example is. Finally, none of the ensemble methods mentioned above were designed to work with pre-trained models.

3 Zero Time Waste

Our goal is to reduce computational costs of neural networks by minimizing redundant operations and information loss. To achieve it, we use the conditional computation setting, in which we dynamically select the route of an input example in a neural network. By controlling the computational route, we can decide how the information is stored and utilized within the model for each particular example. Intuitively, difficult examples require more resources to process, but using the same amount of compute for easy examples is wasteful. Below we describe our Zero Time Waste method in detail.

In order to adapt already trained models to conditional computation setting, we attach and train early exit classifier heads on top of several selected layers, without changing the parameters of the base network. During inference, the whole model exits through one of them when the response is likely enough, thus saving computational resources.

Formally, we consider a multi-class classification problem, where $x \in \mathbb{R}^D$ denotes an input example and $y \in \{1, \dots, K\}$ is its target class. Let $f_\theta : \mathbb{R}^D \rightarrow \mathbb{R}^K$ be a pre-trained neural network with logit output designed for solving the above classification task. The weights θ will not be modified.

Model overview Following typical early exit frameworks, we add M shallow Internal Classifiers, $IC_1, \dots, IC_M$, on intermediate layers of f_θ . Namely, let g_{ϕ_m} , for $m \in \{1, \dots, M\}$, be the m -th IC network returning K logits, which is attached to hidden layer f_{θ_m} of the base network f_θ . The index m is independent of f_θ layer numbering. In general, M is lower than the overall number of f_θ hidden layers since we do not add ICs after every layer (see more details in Appendix A.1).

Although using ICs to return an answer early can reduce overall computation time [19], in a standard setting each IC makes its decision independently, ignoring the responses returned by previous ICs. As we show in Section 4.2, early layers often give correct answers for examples that are misclassified by later classifiers, and hence discarding their information leads to waste and performance drops. To address this issue, we need mechanisms that collect the information from the first $(m - 1)$ ICs to inform the decision of IC_m . For this purpose, we introduce two complementary techniques: *cascade connections* and *ensembling*, and show how they help reduce information waste and, in turn, accelerate the model.

Cascade connections directly transfer the already inferred information between consecutive ICs instead of re-computing it again. Thus, they improve the performance of initial ICs that lack enough predictive power to classify correctly based on low-level features. Ensembling of individual ICs improves performance as the number of members increases, thus showing greatest improvements in the deeper part of the network. This is visualized in Figure 1 where cascade connections are used first to pass already inferred information to later ICs, while ensembling is utilized to conclude the IC prediction. The details on these two techniques are presented in the following paragraphs.

Cascade connections Inspired by the gradient boosting algorithm and literature on cascading classifiers [39], we allow each IC to improve on the predictions of previous ICs instead of inferring them from scratch. The idea of cascade connections is implemented by adding skip connections that combine the output of the base model hidden layer f_{θ_m} with the logits of IC_{m-1} and pass it to IC_m . The prediction is realized by the softmax function applied to g_{ϕ_m} (the m -th IC network):

$$p_m = \text{softmax}(g_{\phi_m}(f_{\theta_m}(x), g_{\phi_{m-1}} \circ f_{\theta_{m-1}}(x))), \text{ for } m > 1, \quad (1)$$

where $g \circ f(x) = g(f(x))$ denotes the composition of functions. Formally, $p_m = p_m(x; \phi_m)$, where ϕ_m are trainable parameters of IC_m , but we drop these parameters in notation for brevity. IC_1 uses only the information coming from the layer f_{θ_1} which does not need to be the first hidden layer of f_θ . Figure 1 shows the skip connections as red horizontal arrows.

Each IC_m is trained in parallel (with respect to ϕ_m) to optimize the prediction of all output classes using an appropriate loss function $\mathcal{L}(p_m)$, e.g. cross-entropy for classification. However, during the backward step it is crucial to stop the gradient of a loss function from passing to the previous classifier. Allowing the gradients of loss $\mathcal{L}(p_m)$ to affect ϕ_j for $j \in 1, \dots, m-1$ leads to a significant performance degradation of earlier layers due to increased focus on the features important for IC_m , as we show in Appendix C.3.

Ensembling Ensembling in machine learning models reliably increases the performance of a model while improving robustness and uncertainty estimation [8, 23]. The main drawback of this approach is its wastefulness, as it requires to train multiple models and use them to process the same examples. However, in our setup we can adopt this idea to combine predictions which were already pre-computed in previous ICs, with near-zero additional computational cost.

To obtain a reliable zero-waste system, we build ensembles that combine outputs from groups of ICs to provide the final answer of the m -th classifier. Since the classifiers we are using vary significantly in predictive strength (later ICs achieve better performance than early ICs) and their predictions are correlated, the standard approach to deep model ensembling does not work in our case. Thus, we introduce weighted geometric mean with class balancing, which allows us to reliably find a combination of pre-computed responses that maximizes the expected result.

Let $p_1, p_2, \dots, p_m$ be the outputs of m consecutive IC predictions (after cascade connections stage) for a given x (Figure 1). We define the probability of the i -th class in the m -th ensemble to be:

$$q_m^i(x) = \frac{1}{Z_m} b_m^i \prod_{j \leq m} (p_j^i(x))^{w_m^j}, \quad (2)$$

where $b_m^i > 0$ and $w_m^j > 0$, for $j = 1, \dots, m$, are trainable parameters, and Z_m is a normalization factor, such that $\sum_i q_m^i(x) = 1$. Observe that w_m^j can be interpreted as our prior belief in predictions of IC_j , i.e. large weight w_m^j indicates less confidence in the predictions of IC_j . On the other hand, b_m^i represents the prior of i -th class for IC_m . The m indices in w_m and b_m are needed as the weights are trained independently for each subset $\{IC_j : j \leq m\}$. Although there are viable potential approaches to setting these parameters by hand, we verified that optimizing them directly by minimizing the cross-entropy loss on the training dataset works best.

Out of additive and geometric ensemble settings we found the latter to be preferable. In this formulation, a low class confidence of a single IC would significantly reduce the probability of that class in the whole ensemble. In consequence, in order for the confidence of the given class to be high, we require all ICs to be confident in that class. Thus, in geometric ensembling, an incorrect although confident IC answer has less chance of ending calculations prematurely. In the additive setting, the negative impact of a single confident but incorrect IC is much higher, as we show in Appendix C.2. Hence our choice of geometric ensembling.

Direct calculation of the product in (2) might lead to numerical instabilities whenever the probabilities are close to zero. To avoid this problem we note that

$$b_m^i \prod_{j \leq m} (p_j^i(x))^{w_m^j} = b_m^i \exp \left(\sum_{j \leq m} w_m^j \ln p_j^i(x) \right),$$

and that log-probabilities $\ln p_j^i$ can be obtained by running the numerically stable log softmax function on the logits g_{ϕ_m} of the classifier.

Both cascade connections and ensembling have different impact on the model. Cascade connections primarily boost the accuracy of early ICs. Ensembling, on the other hand, improves primarily the performance of later ICs, which combine the information from many previous classifiers.

This is not surprising, given that the power of the ensemble increases with the number of members, provided they are at least weak in the sense of boosting theory [33]. As such, the two techniques introduced above are complementary, which we also show empirically via ablation studies in Appendix C. The whole training procedure is presented in Algorithm 1.

Conditional inference Once a ZTW model is trained, the following question appears: how to use the constructed system at test time? More precisely, we need to dynamically find the shortest processing path for a given input example. For this purpose, we use one of the standard confidence

Algorithm 1 Zero Time Waste

Input: pre-trained model f_θ , cross-entropy loss function $\mathcal{L}$, training set $\mathcal{T}$.

Initialize M shallow models g_{ϕ_m} at selected layers f_{θ_m} .

For $m = 1, \dots, M$ **do in parallel** ▷ Cascade connection ICs

Set p_m according to (1).
minimize $\mathbb{E}_{(x,y) \in \mathcal{T}} [\mathcal{L}(p_m(x), y)]$ wrt. ϕ_m by gradient descent

For $m = 1, \dots, M$ **do** ▷ Geometric Ensembling

Initialize w_m, b_m and define $q_m(x)$ according to (2).
minimize $\mathbb{E}_{(x,y) \in \mathcal{T}} [\mathcal{L}(q_m(x), y)]$ wrt. w_m, b_m by gradient descent

scores given by the probability of the most confident class. If the m -th classifier is confident enough about its prediction, i.e. if

$$\max_i q_m^i > \tau, \text{ for a fixed } \tau > 0, \quad (3)$$

where i is the class index, then we terminate the computation and return the response given by this IC. If this condition is not satisfied, we continue processing x and go to the next IC.

Threshold τ in (3) is a manually selected value, which controls the acceleration-performance trade-off of the model. A lower threshold leads to a significant speed-up at the cost of a possible drop in accuracy. Observe that for $\tau > 1$, we recover the original model f_θ , since none of the ICs is confident enough to answer earlier. In practice, to select its appropriate value, we advise using a held-out set to evaluate a range of possible values of τ .

4 Experiments

In this section we examine the performance of Zero Time Waste and analyze its impact on waste reduction in comparison to two recently proposed early-exit methods: (1) Shallow-Deep Networks (SDN) [19] and (2) Patience-Based Early Exit (PBEE) [47]. In contrast to SDN and PBEE, which train ICs independently, ZTW reuses information from past classifiers to improve the performance. SDN and ZTW use maximum class probability as the confidence estimator, while PBEE checks the number of classifiers in sequence that gave the same prediction. For example, for PBEE $\tau = 2$ means that if the answer of the current IC is the same as the answers of the 2 preceding ICs, we can return that answer, otherwise we continue the computation.

In our experiments, we measure how much computation we can save by re-using responses of ICs while keeping good performance, hence obeying the zero waste paradigm. To evaluate the efficiency of the model, we compute the average number of floating-point operations required to perform the forward pass for a single sample. We use it as a hardware-agnostic measure of inference cost and refer to it simply as the "inference time" in all subsequent references. For the evaluation in supervised learning, we use three datasets: CIFAR-10, CIFAR-100, and Tiny ImageNet, and four commonly used architectures: ResNet-56 [11], MobileNet [15], WideResNet [45], and VGG-16BN [37] as base networks. We check all combinations of methods, datasets, and architectures, giving $3 \cdot 3 \cdot 4 = 36$ models in total, and we additionally evaluate a single architecture on the ImageNet dataset to show that the approach is scalable. Additionally, we examine how Zero Time Waste performs at reducing waste in a reinforcement learning setting of Atari 2600 environments. To the best of our knowledge, we are the first to apply early exit methods to reinforcement learning.

Appendix A.1 describes the details about the network architecture, hyperparameters, and training process. Appendix B contains extended plots and tables, and results of an additional transfer learning experiment. In Appendix C we provide ablation studies, focusing in particular on analyzing how each of the proposed improvements affects the performance, and empirically justifying some of the design choices (e.g. geometric ensembles vs. additive ensembles). We provide the source code for our experiments at <https://github.com/gmum/Zero-Time-Waste>.

4.1 Time Savings in Supervised Learning

We check what percentage of computation of the base network can be saved by reusing the information from previous layers in a supervised learning setting. To do this, we evaluate how each method behaves at a particular fraction of the computational power (measured in floating point operations) of

Table 1: Results on four different architectures and three datasets: Cifar-10, Cifar-100 and Tiny ImageNet. Test accuracy (in percentages) for time budgets: 25%, 50%, 75%, 100% of the base network, and Max without any time limits. The first column shows the test accuracy of the base network. The results represent a mean of three runs and standard deviations are provided in Appendix B. We bold results within two standard deviations of the best model.

ResNet-56							MobileNet						
Data	Algo	25%	50%	75%	100%	Max	Data	Algo	25%	50%	75%	100%	Max
C10 (92.0)	SDN	77.7	87.3	91.1	92.0	92.1	C10 (90.6)	SDN	86.1	90.5	90.8	90.7	90.9
	PBEE	69.8	81.8	87.5	91.0	92.1		PBEE	76.3	85.9	89.7	90.9	91.1
	ZTW	80.3	88.7	91.5	92.1	92.1		ZTW	86.7	90.9	91.4	91.4	91.5
C100 (68.4)	SDN	47.1	57.2	64.7	69.0	69.7	C100 (65.1)	SDN	54.3	63.5	66.8	67.8	67.9
	PBEE	45.2	53.5	60.1	67.0	69.0		PBEE	47.1	61.6	61.6	67.0	68.0
	ZTW	51.3	62.1	68.4	70.7	70.9		ZTW	54.5	65.2	68.4	69.0	69.1
T-IM (53.9)	SDN	31.2	41.2	49.9	54.5	54.7	T-IM (59.3)	SDN	35.6	47.1	55.3	58.9	59.7
	PBEE	29.0	37.6	48.2	53.4	54.3		PBEE	26.7	38.4	50.3	55.6	59.7
	ZTW	35.2	46.2	53.7	56.3	56.4		ZTW	37.3	49.5	56.7	59.7	60.2
WideResNet							VGG						
Data	Algo	25%	50%	75%	100%	Max	Data	Algo	25%	50%	75%	100%	Max
C10 (94.4)	SDN	83.8	91.7	94.1	94.4	94.4	C10 (93.0)	SDN	86.0	92.1	93.0	93.0	93.0
	PBEE	78.0	84.0	90.3	93.8	94.4		PBEE	75.0	86.0	91.0	92.9	93.1
	ZTW	86.7	92.9	94.5	94.7	94.7		ZTW	87.1	92.5	93.2	93.2	93.2
C100 (75.1)	SDN	55.9	65.1	71.6	75.0	75.4	C100 (70.4)	SDN	58.5	67.2	70.6	71.4	71.5
	PBEE	46.7	57.2	66.0	73.2	75.4		PBEE	51.2	65.3	65.3	70.9	72.0
	ZTW	59.5	69.1	74.5	76.2	76.4		ZTW	60.2	69.3	72.6	73.5	73.6
T-IM (59.6)	SDN	36.8	46.0	54.6	59.4	59.7	T-IM (59.0)	SDN	40.0	50.5	57.4	59.6	59.7
	PBEE	29.9	37.8	52.7	58.5	59.7		PBEE	31.0	45.2	55.2	60.1	60.2
	ZTW	40.0	50.1	57.5	60.2	60.3		ZTW	41.4	52.3	59.3	60.1	60.5

the base network. We select the highest threshold τ such that the average inference time is smaller than, for example, 25% of the original time. Then we calculate accuracy for that threshold. Table 1 contains summary of this analysis, averaged over three seeds, with further details (plots for all thresholds, standard deviations) shown in Appendix B.1.

Looking at the results, we highlight the fact that methods which do not reuse information between ICs do not always achieve the goal of reducing computational waste. For example, SDN and PBEE cannot maintain the accuracy of the base network for MobileNet on Tiny ImageNet when using the same computational power, scoring respectively 0.4 and 3.7 percentage points lower than the baseline. Adding ICs to the network and then discarding their predictions when they are not confident enough to return the final answer introduces computational overhead without any gains. By reusing the information from previous ICs ZTW overcomes this issue and maintains the accuracy of the base network for all considered settings. In particular cases, such as ResNet-56 on Tiny ImageNet or MobileNet on Cifar-100, Zero Time Waste even significantly outperforms the core network.

Similar observation can be made for other inference time limits as well. ZTW consistently maintains high accuracy using less computational resources than the other approaches, for all combinations of datasets and architectures. Although PBEE reuses information from previous layers to decide whether to stop computation or not, this is not sufficient to reduce the waste in the network. While PBEE outperforms SDN when given higher inference time limits, it often fails for smaller limits (25%, 50%). We hypothesize that this is result of the fact that PBEE has smaller flexibility with respect to τ . While for SDN and ZTW values of τ are continuous, for PBEE they represent a discrete number of ICs that must sequentially agree before returning an answer.

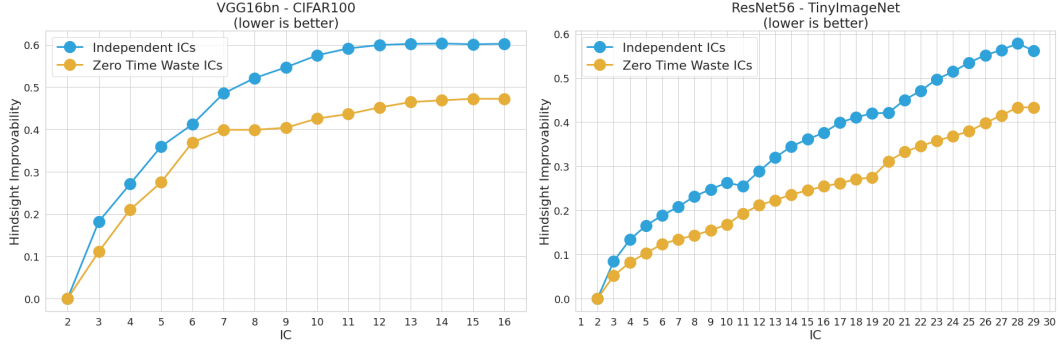


Figure 2: Hindsight Improvability. For each IC (horizontal axis) we look at examples it misclassified and we check how many of them were classified correctly by any of the previous ICs. The lower the number, the better the IC is at reusing previous information.

Finally, we check whether our observations scale up to larger datasets by running experiments on ImageNet using a pre-trained ResNet-50 from the torchvision package³. The results presented in Table 2 show that Zero Time Waste is able to gain significant improvements over the two tested baselines even in this more challenging setting. Additional details of this experiments are presented in Appendix B.2.

Given the performance of ZTW, the results show that paying attention to the minimization of computational waste leads to tangible, practical improvements of the inference time of the network. Therefore, we devote next section to explaining where the empirical gains come from and how to measure information loss in the models.

4.2 Information Loss in Early Exit Models

Since ICs in a given model are heavily correlated, it is not immediately obvious why reusing past predictions should improve performance. Later ICs operate on high-level features for which class separation is much easier than for early ICs, and hence get better accuracy. Thus, we ask a question — is there something that early ICs know that the later ICs do not?

For that purpose, we introduce a metric to evaluate how much a given IC could improve performance by reusing information from all previous ICs. We measure it by checking how many examples incorrectly classified by IC_m were classified correctly by any of the previous ICs. An IC which reuses predictions from the past perfectly would achieve a low score on this metric since it would remember all the correct answers of the previous ICs. On the other hand, an IC in a model which trains each classifier independently would have a higher score on this metric, since it does not use past information at all. We call this metric Hindsight Improvability (HI) since it measures how many mistakes we would be able to avoid if we used information from the past efficiently.

Let $\mathcal{C}_m$ denote the set of examples correctly classified by IC_m , with its complement $\bar{\mathcal{C}}_m$ being the set of examples classified incorrectly. To measure the Hindsight Improvability of IC_m we calculate:

$$HI_m = \frac{|\bar{\mathcal{C}}_m \cap (\bigcup_{n < m} \mathcal{C}_n)|}{|\bar{\mathcal{C}}_m|}$$

Figure 2 compares the values of HI for a method with independent ICs (SDN in this case) and ZTW which explicitly recycles computations. In the case of VGG16 trained with independent ICs, over 60% of the mistakes could be avoided if we properly used information from the past, which

³<https://pytorch.org/vision/stable/index.html>

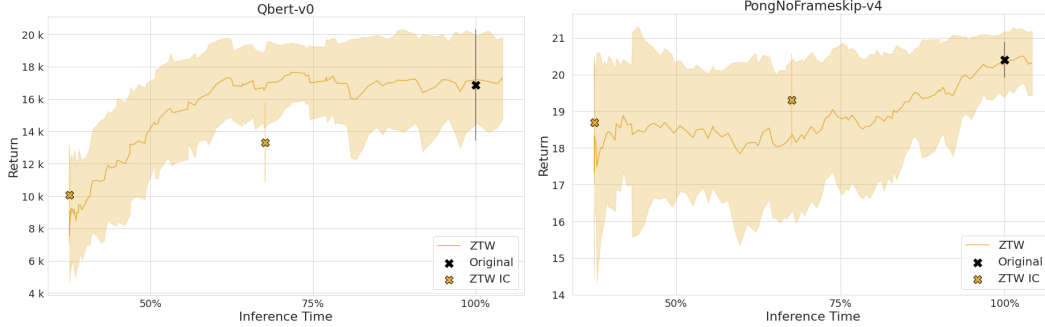


Figure 3: Inference time vs. average return of the ZTW policy in an RL setting on Q*bert and Pong Atari 2600 environments. The plot was generated by using different values of the confidence threshold τ hyperparameter. Since the RL environments are stochastic, we plot the return with a standard deviation calculated on 10 runs. ZTW saves a significant amount of computation while preserving the original performance, showcasing that waste can be minimized also in the reinforcement learning domain.

would translate to improvement from 71.5% to 82.9% accuracy. Similarly, for ResNet-56 trained on TinyImageNet, the number of errors could be cut by around 57%.

ZTW consistently outperforms the baseline, with the largest differences visible at the later ICs, which can in principle gain the most from reusing previous predictions. Thus, Zero Time Waste is able to efficiently recycle information from the past. At the same time, there is still a room for significant improvements, which shows that future zero waste approaches could offer additional enhancements.

4.3 Time Savings in Reinforcement Learning

Although supervised learning is an important testbed for deep learning, it does not properly reflect the challenges encountered in the real world. In order to examine the impact of waste-minimization methods in a setting that reflects the sequential nature of interacting with the world, we evaluate it in a Reinforcement Learning (RL) setting. In particular, we use the environments from the suite of Atari 2600 games [28].

Similarly as in the supervised setting, we start with a pre-trained network, which in this case represents a policy trained with the Proximal Policy Optimization (PPO) algorithm [36]. We attach the ICs to the network and train it by distilling the knowledge from the core network to the ICs. We use a behavioral cloning approach, where the states are sampled from the policy defined by the ICs and the labels are provided by the expert model. Since actions in Atari 2600 are discrete, we can then use the same confidence threshold-based approach to early exit inference as in the case of classification. More details about the training process are provided in the Appendix A.2.

In order to investigate the relationship between computation waste reduction and performance, we evaluate Zero Time Waste for different values of confidence threshold τ . By setting a very high τ value, we retrieve the performance of the original model (none of the ICs respond) and by slowly decreasing its value we can reduce the computational cost (ICs begin to return answers earlier). In Figure 3 we check values of τ in the interval $[0.1, 1.0]$ to show how ZTW is able to control the acceleration-performance balance for Q*Bert and Pong, two popular Atari 2600 environments. By setting lower τ thresholds for Q*Bert we can save around 45% of computations without score degradation. Similarly, for Pong we can get 60% reduction with minor impact on performance (note that average human score is 9.3 points). This shows that even the small four-layered convolutional architecture commonly used for Atari [28] introduces a noticeable waste of computation which can be mitigated within a zero-waste paradigm. We highlight this fact as the field of reinforcement learning has largely focused on efficiency in terms of number of samples and training time, while paying less attention to the issue of efficient inference.

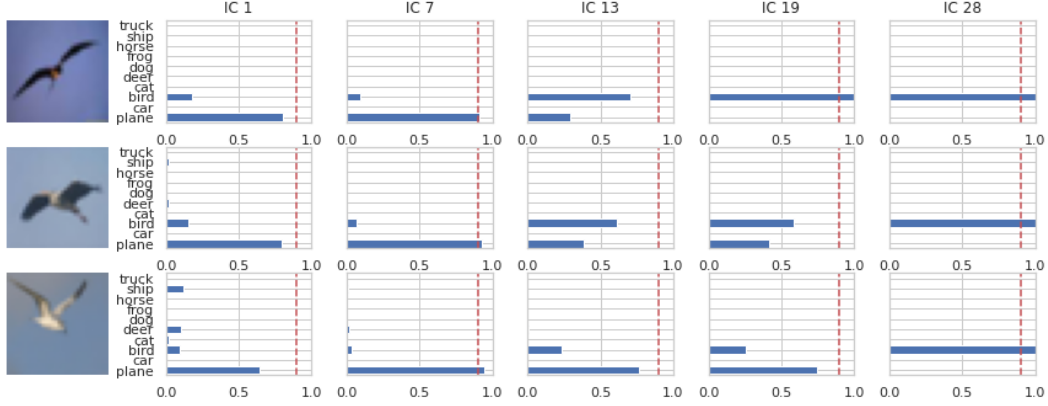


Figure 4: Examples of bird images which were incorrectly classified as airplanes by ZTW. The early ICs are misled by the low-level features (blue sky, sharp edges, grayscale silhouette) and return a prediction before the later ICs can detect more subtle high-level features.

4.4 Impact & Limitations

Our framework is the cornerstone of an environmental-aware computation where information recycling within a model is cautiously studied to avoid wasting resources. The focus on computational efficiency, however, introduces a natural trade-off between model accuracy and its computational cost. Although in most cases we can carefully adjust the appropriate method hyperparameters to avoid a significant accuracy drop, some testing samples remain surprisingly challenging for ZTW, which indicates a need for further investigation of the accuracy vs. computation cost trade-off offered by our method.

Figure 4 contains examples of images for which low-level features in a given image consistently point at a wrong class, while high-level features would allow us to deduce the correct class. Images of birds which contain sharp lines and grayscale silhouettes are interpreted as airplanes by early ICs which operate on low-level features. If the confidence of these classifiers gets high enough, the answer might be returned before later classifiers can correct this decision. We highlight the problem of dealing with examples which are seemingly easy but turn out difficult as an important future direction for conditional computation methods.

5 Conclusion

In this work, we show that discarding predictions of the previous ICs in early exit models leads to waste of computation resources and a significant loss of information. This result is supported by the introduced Hindsight Improvability metric, as well as empirical result for reducing computations in existing networks. The proposed Zero Time Waste method attempts to solve these issues by incorporating outputs from the past heads by using cascade connections and geometric ensembling. We show that ZTW outperforms other approaches on multiple standard datasets and architectures for supervised learning, as well as in Atari 2600 reinforcement learning suite. At the same time we postulate that focusing on reducing the computational waste in a safe and stable way is an important direction for future research in deep learning.

Acknowledgments & Funding Disclosure

This research was funded by Foundation for Polish Science (grant no POIR.04.04.00-00-14DE/18-00 carried out within the Team-Net program co-financed by the European Union under the European Regional Development Fund) and National Science Centre, Poland (grant no 2018/31/B/ST6/00993 and grant no 2020/39/B/ST6/01511). The authors have applied a CC BY license to any Author Accepted Manuscript (AAM) version arising from this submission, in accordance with the grants' open access conditions.

References

- [1] Dan Ariely and Michael I. Norton. From thinking too little to thinking too much: a continuum of decision making. *WIREs Cognitive Science*, 2(1):39–46, 2011.
- [2] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv:1308.3432*, 2013.
- [3] Candice Bentéjac, Anna Csörgő, and Gonzalo Martínez-Muñoz. A comparative analysis of gradient boosting algorithms. *Artificial Intelligence Review*, pages 1–31, 2020.
- [4] Konstantin Berestizshevsky and Guy Even. Dynamically sacrificing accuracy for reduced computation: cascaded inference based on softmax confidence. In *Proceedings of the International Conference on Artificial Neural Networks*, ICANN, pages 306–320. Springer, 2019.
- [5] Andrew Davis and Itamar Arel. Low-rank approximations for conditional feedforward computation in deep neural networks. *arXiv:1312.4461*, 2013.
- [6] Thomas G Dietterich. Ensemble methods in machine learning. In *Proceedings of the International Workshop on Multiple Classifier Systems*, page 15. Springer, 2000.
- [7] Gabriel Dulac-Arnold, Daniel J. Mankowitz, and Todd Hester. Challenges of real-world reinforcement learning. *CoRR*, abs/1904.12901, 2019.
- [8] Stanislav Fort, Huiyi Hu, and Balaji Lakshminarayanan. Deep ensembles: a loss landscape perspective. *arXiv:1912.02757*, 2019.
- [9] Gerd Gigerenzer and Wolfgang Gaissmaier. Heuristic decision making. *Annual Review of Psychology*, 62:451–82, 2011.
- [10] Sorin Grigorescu, Bogdan Trasnea, Tiberiu Cocias, and Gigel Macesanu. A survey of deep learning techniques for autonomous driving. *Journal of Field Robotics*, 37(3):362–386, 2020.
- [11] 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*, CVPR, pages 770–778, 2016.
- [12] Yihui He, Xiangyu Zhang, and Jian Sun. Channel pruning for accelerating very deep neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1389–1397, 2017.
- [13] Todd Hester and Peter Stone. TEXPLORE: Real-time sample-efficient reinforcement learning for robots. *Machine Learning*, 90(3):385–429, 2013.
- [14] Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. In *Proceedings of the NIPS Workshop on Deep Learning and Representation Learning*, 2015.
- [15] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv:1704.04861*, 2017.
- [16] Gao Huang, Danlu Chen, Tianhong Li, Felix Wu, Laurens van der Maaten, and Kilian Q. Weinberger. Multi-scale dense networks for resource efficient image classification. In *International Conference on Learning Representations*, 2018.
- [17] Sunggoo Jung, Sunyou Hwang, Heemin Shin, and David Hyunchul Shim. Perception, guidance, and navigation for indoor autonomous drone racing using deep learning. *IEEE Robotics and Automation Letters*, 3(3):2539–2544, 2018.
- [18] Daniel Kahneman. *Thinking, fast and slow*. Farrar, Straus and Giroux, 2017.
- [19] Yigitcan Kaya, Sanghyun Hong, and Tudor Dumitras. Shallow-deep networks: Understanding and mitigating network overthinking. In *Proceedings of the International Conference on Machine Learning*, ICML, pages 3301–3310, 2019.
- [20] Daniel S Kermany, Michael Goldbaum, Wenjia Cai, Carolina CS Valentim, Huiying Liang, Sally L Baxter, Alex McKeown, Ge Yang, Xiaokang Wu, Fangbing Yan, et al. Identifying medical diagnoses and treatable diseases by image-based deep learning. *Cell*, 172(5):1122–1131, 2018.

- [21] Alexandros Kouris, Stylianos I. Venieris, Michail Rizakis, and Christos-Savvas Bouganis. Approximate LSTMs for time-constrained inference: Enabling fast reaction in self-driving cars, 2019.
- [22] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.
- [23] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, NIPS, pages 6402–6413, 2017.
- [24] Juhyoung Lee, Sangyeob Kim, Sangjin Kim, Wooyoung Jo, and Hoi-Jun Yoo. Gst: Group-sparse training for accelerating deep reinforcement learning, 2021.
- [25] Hao Li, Hong Zhang, Xiaojuan Qi, Yang Ruigang, and Gao Huang. Improved techniques for training adaptive deep networks. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1891–1900, 2019.
- [26] Dor Livne and Kobi Cohen. Pops: Policy pruning and shrinking for deep reinforcement learning, 2020.
- [27] Mason McGill and Pietro Perona. Deciding how to decide: dynamic routing in artificial neural networks. In *ICML’17 Proceedings of the 34th International Conference on Machine Learning - Volume 70*, pages 2363–2372, 2017.
- [28] 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):529–533, 2015.
- [29] Mary Phuong and Christoph H Lampert. Distillation-based training for multi-exit architectures. In *Proceedings of the IEEE International Conference on Computer Vision, ICCV*, pages 1355–1364, 2019.
- [30] Antonin Raffin, Ashley Hill, Maximilian Ernestus, Adam Gleave, Anssi Kanervisto, and Noah Dormann. Stable baselines3. <https://github.com/DLR-RM/stable-baselines3>, 2019.
- [31] Simone Scardapane, Danilo Comminiello, Michele Scarpiniti, Enzo Baccarelli, and Aurelio Uncini. Differentiable branching in deep networks for fast inference. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP*, pages 4167–4171, 2020.
- [32] Simone Scardapane, Michele Scarpiniti, Enzo Baccarelli, and Aurelio Uncini. Why should we add early exits to neural networks? *arXiv:2004.12814*, 2020.
- [33] Robert E. Schapire. The strength of weak learnability. *Machine Learning*, 5(2):197–227, 1990.
- [34] E. Schuitema, L. Buşoniu, Robert Babuška, and P. Jonker. Control delay in reinforcement learning for real-time dynamic systems: A memoryless approach. *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3226–3231, 2010.
- [35] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv:1506.02438*, 2015.
- [36] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.
- [37] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proceedings of the International Conference on Learning Representations, ICLR*, 2015.
- [38] Surat Teerapittayanon, Bradley McDanel, and Hsiang-Tsung Kung. Branchynet: Fast inference via early exiting from deep neural networks. In *Proceedings of the International Conference on Pattern Recognition, ICPR*, pages 2464–2469, 2016.
- [39] Paul Viola and Michael J. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154, 2004.
- [40] Xin Wang, Yujia Luo, Daniel Crankshaw, Alexey Tumanov, Fisher Yu, and Joseph E. Gonzalez. Idk cascades: Fast deep learning by learning not to overthink. In *UAI*, pages 580–590, 2017.
- [41] Xin Wang, Fisher Yu, Zi-Yi Dou, Trevor Darrell, and Joseph E Gonzalez. Skipnet: learning dynamic routing in convolutional networks. In *Proceedings of the European Conference on Computer Vision, ECCV*, pages 409–424, 2018.

- [42] Xin Wang, Fisher Yu, Lisa Dunlap, Yi-An Ma, Ruth Wang, Azalia Mirhoseini, Trevor Darrell, and Joseph E Gonzalez. Deep mixture of experts via shallow embedding. In *Proceedings of the Uncertainty in Artificial Intelligence*, UAI, pages 552–562, 2020.
- [43] Yulin Wang, Kangchen Lv, Rui Huang, Shiji Song, Le Yang, and Gao Huang. Glance and focus: a dynamic approach to reducing spatial redundancy in image classification. In *Advances in Neural Information Processing Systems*, volume 33, pages 2432–2444, 2020.
- [44] Le Yang, Yizeng Han, Xi Chen, Shiji Song, Jifeng Dai, and Gao Huang. Resolution adaptive networks for efficient inference. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2369–2378, 2020.
- [45] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv:1605.07146*, 2016.
- [46] Hongjie Zhang, Zhuocheng He, and Jing Li. Accelerating the deep reinforcement learning with neural network compression. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2019.
- [47] Wangchunshu Zhou, Canwen Xu, Tao Ge, Julian McAuley, Ke Xu, and Furu Wei. BERT loses patience: fast and robust inference with early exit. *arXiv:2006.04152*, 2020.

PluGeN: Multi-Label Conditional Generation from Pre-trained Models

Maciej Wołczyk^{1*}, Magdalena Proszewska^{1*}, Łukasz Maziarka¹,
Maciej Zieba^{2,4}, Patryk Wielopolski², Rafał Kurczab³, Marek Śmieja^{1†}

¹Jagiellonian University

²Wrocław University of Science and Technology

³Institute of Pharmacology PAS

⁴Tooploox

Abstract

Modern generative models achieve excellent quality in a variety of tasks including image or text generation and chemical molecule modeling. However, existing methods often lack the essential ability to generate examples with requested properties, such as the age of the person in the photo or the weight of the generated molecule. Incorporating such additional conditioning factors would require rebuilding the entire architecture and optimizing the parameters from scratch. Moreover, it is difficult to disentangle selected attributes so that to perform edits of only one attribute while leaving the others unchanged. To overcome these limitations we propose PluGeN (Plugin Generative Network), a simple yet effective generative technique that can be used as a plugin to pre-trained generative models. The idea behind our approach is to transform the entangled latent representation using a flow-based module into a multi-dimensional space where the values of each attribute are modeled as an independent one-dimensional distribution. In consequence, PluGeN can generate new samples with desired attributes as well as manipulate labeled attributes of existing examples. Due to the disentangling of the latent representation, we are even able to generate samples with rare or unseen combinations of attributes in the dataset, such as a young person with gray hair, men with make-up, or women with beards. We combined PluGeN with GAN and VAE models and applied it to conditional generation and manipulation of images and chemical molecule modeling. Experiments demonstrate that PluGeN preserves the quality of backbone models while adding the ability to control the values of labeled attributes. Implementation is available at <https://github.com/gmum/plugin>.

Introduction

Generative models such as GANs and variational autoencoders have achieved great results in recent years, especially in the domains of images (Brock, Donahue, and Simonyan 2018; Brown et al. 2020) and cheminformatics (Gómez-Bombarelli et al. 2018; Jin, Barzilay, and Jaakkola 2018). However, in many practical applications, we need to control the process of creating samples by enforcing particular features of generated objects. This would be required to regulate the biases present in the data, e.g. to assure that people

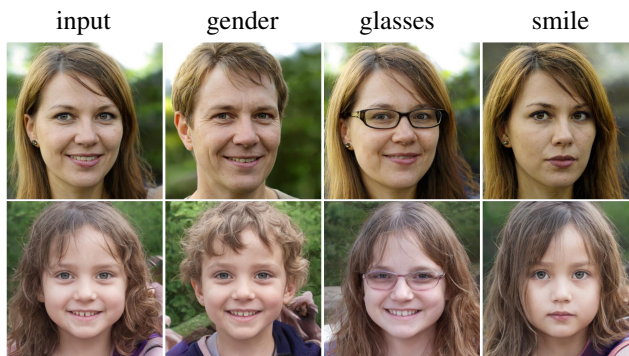


Figure 1: Attributes manipulation performed by PluGeN using the StyleGAN backbone.

of each ethnicity are properly represented in the generated set of face images. In numerous realistic problems, such as drug discovery, we want to find objects with desired properties, like molecules with a particular activity, non-toxicity, and solubility.

Designing the conditional variants of generative models that operate on multiple labels is a challenging problem due to intricate relations among the attributes. Practically, it means that some combinations of attributes (e.g. a woman with a beard) might be unobserved or rarely observed in the training data. In essence, the model should be able to go beyond the distribution of seen data and generate examples with combinations of attributes not encountered previously. One might approach this problem by building a new conditional generative model from the ground up or design a solution tailored for a specific existing unsupervised generative model. However, this introduces an additional effort when one wants to adapt it to a newly invented approach.

To tackle this problem while leveraging the power of existing techniques, we propose PluGeN (Plugin Generative Network), a simple yet effective generative technique that can be used as a plugin to various pre-trained generative models such as VAEs or GANs, see Figure 1 for demonstration. Making use of PluGeN, we can manipulate the attributes of input examples as well as generate new samples with desired features. When training the proposed module, we do not change the parameters of the base model and thus

*Equal contribution

†Corresponding author: marek.smieja@uj.edu.pl

Copyright © 2022, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

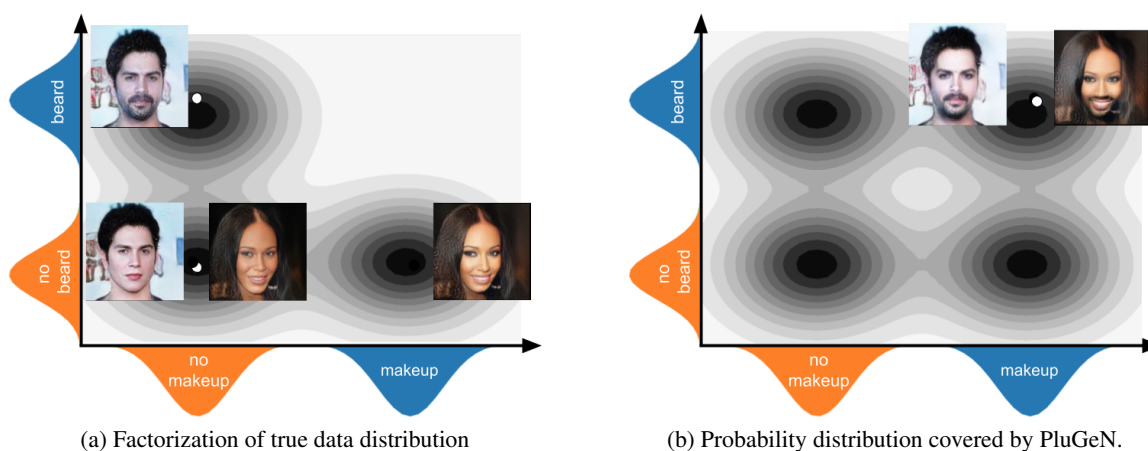


Figure 2: PluGeN factorizes true data distribution into components (marginal distributions) related to labeled attributes, see (a), and allows for describing unexplored regions of data (uncommon combinations of labels) by sampling from independent components, see (b). In the case illustrated here, PluGeN constructs pictures of men with make-up or women with beards, although such examples rarely (or never) appear in the training set.

we retain its generative and reconstructive abilities, which places our work in the emerging family of non-invasive network adaptation methods (Wołczyk et al. 2021; Rebuffi, Bilen, and Vedaldi 2017; Koperski et al. 2020; Kaya, Hong, and Dumitras 2019; Zhou et al. 2020).

Our idea is to find a mapping between the entangled latent representation of the backbone model and a disentangled space, where each dimension corresponds to a single, interpretable attribute of the image. By factorizing the true data distribution into independent components, we can sample from each component independently, which results in creating samples with arbitrary combinations of attributes, see Figure 2. In contrast to many previous works, which are constrained to the attributes combinations visible in the training set, PluGeN gives us full control of the generation process, being able to create uncommon combinations of attributes, such as a woman with a beard or a man with heavy make-up. Generating samples with unseen combinations of attributes can be viewed as extending the distribution of generative models to unexplored although reasonable regions of data space, which distinguishes our approach from existing solutions.

Extensive experiments performed on the domain of images and a dataset of chemical compounds demonstrate that PluGeN is a reusable plugin that can be applied to various architectures including GANs and VAEs. In contrast to the baselines, PluGeN can generate new samples as well as manipulate the properties of existing examples, being capable of creating uncommon combinations of attributes.

Our contributions are as follow:

- We propose a universal and reusable plugin for multi-label generation and manipulation that can be attached to various generative models and applied it to diverse domains, such as chemical molecule modeling.
- We introduce a novel way of modeling conditional distributions using invertible normalizing flows based on the latent space factorization.

- We experimentally demonstrate that PluGeN can produce samples with uncommon combinations of attributes going beyond the distribution of training data.

Related work

Conditional VAE (cVAE) is one of the first methods which includes additional information about the labeled attributes in a generative model (Kingma et al. 2014). Although this approach has been widely used in various areas ranging from image generation (Sohn, Lee, and Yan 2015; Yan et al. 2016; Klys, Snell, and Zemel 2018) to molecular design (Kang and Cho 2018), the independence of the latent vector from the attribute data is not assured, which negatively influences the generation quality. Conditional GAN (cGAN) is an alternative approach that gives results of significantly better quality (Mirza and Osindero 2014; Perarnau et al. 2016; He et al. 2019), but the model is more difficult to train (Kodali et al. 2017). cGAN works very well for generating new images and conditioning factors may take various forms (images, sketches, labels) (Park et al. 2019; Jo and Park 2019; Choi et al. 2020), but manipulating existing examples is more problematic because GAN models lack the encoder network (Tov et al. 2021). Fader Networks (Lample et al. 2017) combine features of both cVAE and cGAN, as they use encoder-decoder architecture, together with the discriminator, which predicts the image attributes from its latent vector returned from the encoder. As discussed in (Li et al. 2020), the training of Fader Networks is even more difficult than standard GANs, and disentanglement of attributes is not preserved. MSP (Li et al. 2020) is a recent auto-encoder based architecture with an additional projection matrix, which is responsible for disentangling the latent space and separating the attribute information from other characteristic information. In contrast to PluGeN, MSP cannot be used with pre-trained GANs and performs poorly at generating new images (it was designed for manipulating existing examples). CAGlow (Liu et al. 2019) is an adaptation of Glow (Kingma and Dhariwal

2018) to conditional image generation based on modeling a joint probabilistic density of an image and its conditions. Since CAGlow does not reduce data dimension, applying it to more complex data might be problematic.

While the above approaches focus on building conditional generative models from scratch, recent works often focus on manipulating the latent codes of pre-trained models. StyleFlow (Abdal et al. 2021) operates on the latent space of StyleGAN (Karras, Laine, and Aila 2019) using a conditional continuous flow module. Although the quality of generated images is impressive, the model has not been applied to other generative models than StyleGAN and domains other than images. Moreover, StyleFlow needs an additional classifier to compute the conditioning factor (labels) for images at test time. Competitive approaches to StyleGAN appear in (Gao et al. 2021; Tewari et al. 2020; Härkönen et al. 2020; Nitzan et al. 2020). InterFaceGAN (Shen et al. 2020) postulates that various properties of the facial semantics can be manipulated via linear models applied to the latent space of GANs. Hijack-GAN (Wang, Yu, and Fritz 2021) goes beyond linear models and designs a proxy model to traverse the latent space of GANs.

In disentanglement learning, we assume that the data has been generated from a fixed number of independent factors of underlying variation. The goal is then to find a transformation that unravels these factors so that a change in one dimension of the latent space corresponds to a change in one factor of variation while being relatively invariant to changes in other factors (Bengio, Courville, and Vincent 2013; Kim and Mnih 2018; Higgins et al. 2017; Brakel and Bengio 2017; Kumar, Sattigeri, and Balakrishnan 2017; Chen et al. 2019; Spurek et al. 2020; Dinh, Krueger, and Bengio 2014; Sorrenson, Rother, and Köthe 2020; Chen et al. 2016). As theoretically shown in (Locatello et al. 2019), the unsupervised learning of disentangled representations is fundamentally impossible without inductive biases on both the models and the data. In this paper, we solve a slightly different problem than typical disentanglement, as we aim to deliver an efficient plug-in model to a large variety of existing models in order to manipulate attributes without training the entire system. Creating compact add-ons for large models saves training time and energy consumption.

Plugin Generative Network

We propose a plugin generative network (PluGeN), which can be attached to pre-trained generative models and allows for direct manipulation of labeled attributes, see Figure 3 for the basic scheme of PluGeN. Making use of PluGeN we preserve all properties of the base model, such as generation quality and reconstruction in the case of auto-encoders, while adding new functionalities. In particular, we can:

- modify selected attributes of existing examples,
- generate new samples with desired labels.

In contrast to typical conditional generative models, PluGeN is capable of creating examples with rare or even unseen combinations of attributes, e.g. man with makeup.

Probabilistic model. PluGeN works in a multi-label setting, where every example $\mathbf{x} \in \mathcal{X}$ is associated with a K -

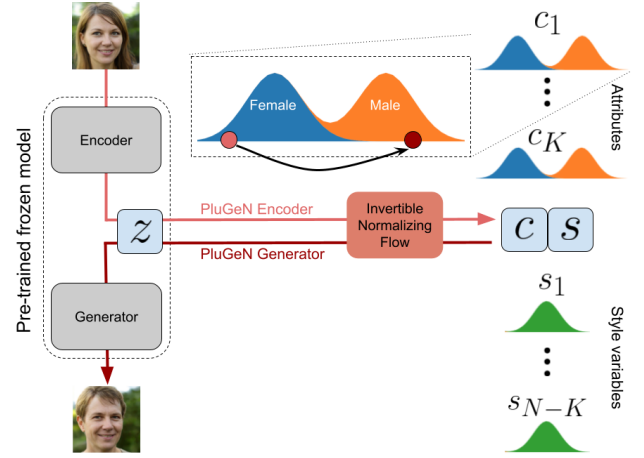


Figure 3: PluGeN maps the entangled latent space $\mathcal{Z}$ of pre-trained generative models using invertible normalizing flow into a separate space, where labeled attributes are modeled using independent 1-dimensional distributions. By manipulating label variables in this space, we fully control the generation process.

dimensional vector of binary labels¹ $\mathbf{y} = (y_1, \dots, y_K) \in \{0, 1\}^K$. We assume that there is a pre-trained generative model $\mathcal{G} : \mathcal{Z} \rightarrow \mathbb{R}^D$, where $\mathcal{Z} \subset \mathbb{R}^N$ is the latent space, which is usually heavily entangled. That is, although each latent code $\mathbf{z} \in \mathcal{Z}$ contains the information about the labels $\mathbf{y}$, there is no direct way to extract or modify it.

We want to map this entangled latent space $\mathcal{Z}$ into a separate latent space $\mathcal{D} \subset \mathbb{R}^N$ which encodes the values of each label y_k as a separate random variable C_k living in a single dimension of this space. Thus, by changing the value of C_k , going back to the entangled space $\mathcal{Z}$ and generating a sample, we can control the values of y_k . Since labeled attributes usually do not fully describe a given example, we consider additional $N - K$ random variables S_k , which are supposed to encode the information not included in the labels. We call $\mathbf{C} = (C_1, \dots, C_K)$ the label variables (or attributes) and $\mathbf{S} = (S_1, \dots, S_{N-K})$ the style variables.

Since we want to control the value of each attribute independently of any other factors, we assume the factorized form of the probability distribution of the random vector $(\mathbf{C}, \mathbf{S})$. More precisely, the conditional probability distribution of $(\mathbf{C}, \mathbf{S})$ given any condition $\mathbf{Y} = \mathbf{y}$ imposed on labeled attributes is of the form:

$$p_{\mathbf{C}, \mathbf{S} | \mathbf{Y} = \mathbf{y}}(\mathbf{c}, \mathbf{s}) = \prod_{i=1}^K p_{C_i | Y_i = y_i}(c_i) \cdot p_{\mathbf{S}}(\mathbf{s}), \quad (1)$$

for all $(\mathbf{c}, \mathbf{s}) = (c_1, \dots, c_K, s_1, \dots, s_{N-K}) \in \mathbb{R}^N$. In other words, modifying $Y_i = y_i$ influences only the i -th factor C_i leaving other features unchanged.

Parametrization. To instantiate the above probabilistic model (1), we need to parametrize the conditional distribu-

¹Our model can be extended to continuous values, which we describe in the supplementary materials due to page limit.

tion of C_i given $Y_i = y_i$ and the distribution of $\mathbf{S}$. Since we do not impose any constraints on style variables, we use standard Gaussian distribution for modeling density of $\mathbf{S}$:

$$p_{\mathbf{S}} = \mathcal{N}(0, I_{N-K}).$$

To provide the consistency with $p_{\mathbf{S}}$ and avoid potential problems with training our deep learning model using discrete distributions, we use the mixture of two Gaussians for modeling the presence of labels – each component corresponds to a potential value of the label (0 or 1). More precisely, the conditional distribution of C_i given $Y_i = y_i$ is parametrized by:

$$p_{C_i|Y_i=y_i} = \mathcal{N}(m_0, \sigma_0)^{(1-y_i)} \cdot \mathcal{N}(m_1, \sigma_1)^{y_i}, \quad (2)$$

where $m_0, m_1, \sigma_0, \sigma_1$ are the user-defined parameters. If $y_i = 0$, then the latent factor C_i takes values close to m_0 ; otherwise we get values around m_1 (depending on the value of σ_0 and σ_1). To provide good separation between components, we put $m_0 = -1, m_1 = 1$; the selection of σ_0, σ_1 will be discussed in the supplementary materials.

Thanks to this continuous parametrization, we can smoothly interpolate between different labels, which would not be so easy using e.g. Gumbel softmax parametrization (Jang, Gu, and Poole 2016). In consequence, we can gradually change the intensity of certain labels, like smile or beard, even though such information was not available in a training set (see Figure 4 in the experimental section).

Training the model To establish a two-way mapping between entangled space $\mathcal{Z}$ and the disentangled space $\mathcal{D}$, we use an invertible normalizing flow (INF), $\mathcal{F} : \mathbb{R}^N \rightarrow \mathcal{Z}$. Let us recall that INF is a neural network, where the inverse mapping is given explicitly and the Jacobian determinant can be easily calculated (Dinh, Krueger, and Bengio 2014). Due to the invertibility of INF, we can transform latent codes $\mathbf{z} \in \mathcal{Z}$ to the prior distribution of INF, modify selected attributes, and map the resulting vector back to $\mathcal{Z}$. Moreover, INFs can be trained using log-likelihood loss, which is very appealing in generative modeling.

Summarizing, given a latent representation $\mathbf{z} \in \mathcal{Z}$ of a sample $\mathbf{x}$ with label $\mathbf{y}$, the loss function of PluGeN equals:

$$\begin{aligned} & -\log p_{\mathbf{Z}|\mathbf{Y}=\mathbf{y}}(\mathbf{z}) = \\ & -\log \left(p_{\mathbf{C},\mathbf{S}|\mathbf{Y}=\mathbf{y}}(\mathbf{c}, \mathbf{s}) \cdot \left| \det \frac{\partial \mathcal{F}^{-1}(\mathbf{z})}{\partial \mathbf{z}} \right| \right) = \\ & -\log \left(\prod_{i=1}^K p_{C_i|Y_i=y_i}(c_i) \cdot p_{\mathbf{S}}(\mathbf{s}) \right) - \log \left| \det \frac{\partial \mathcal{F}^{-1}(\mathbf{z})}{\partial \mathbf{z}} \right| = \\ & -\sum_{i=1}^K \log p_{C_i|Y_i=y_i}(c_i) - \log p_{\mathbf{S}}(\mathbf{s}) - \log \left| \det \frac{\partial \mathcal{F}^{-1}(\mathbf{z})}{\partial \mathbf{z}} \right|, \end{aligned} \quad (3)$$

where $(\mathbf{c}, \mathbf{s}) = \mathcal{F}^{-1}(\mathbf{z})$. In the training phase, we collect latent representations $\mathbf{z}$ of data points $\mathbf{x}$. Making use of labeled attributes $\mathbf{y}$ associated with every $\mathbf{x}$, we modify the weights of $\mathcal{F}$ so that to minimize the negative log-likelihood (3) using gradient descent. The weights of the base model $\mathcal{G}$ are kept frozen.

In contrast to many previous works (Abdal et al. 2021), PluGeN can be trained in a semi-supervised setting, where only partial information about labeled attributes is available (see supplementary materials for details).

Inference. We may use PluGeN to generate new samples with desired attributes as well as to manipulate attributes of input examples. In the first case, we generate a vector $(\mathbf{c}, \mathbf{s})$ from the conditional distribution $p_{\mathbf{C},\mathbf{S}|\mathbf{Y}=\mathbf{y}}$ with selected condition $\mathbf{y}$. To get the output sample, the vector $(\mathbf{c}, \mathbf{s})$ is transformed by the INF and the base generative network $\mathcal{G}$, which gives us the final output $\mathbf{x} = \mathcal{G}(\mathcal{F}(\mathbf{c}, \mathbf{s}))$.

In the second case, to manipulate the attributes of an existing example $\mathbf{x}$, we need to find its latent representation $\mathbf{z}$. If $\mathcal{G}$ is a decoder network of an autoencoder model, then $\mathbf{x}$ should be passed through the encoder network to obtain $\mathbf{z}$ (Li et al. 2020). If $\mathcal{G}$ is a GAN, then $\mathbf{z}$ can be found by minimizing the reconstruction error between $\mathbf{x}' = \mathcal{G}(\mathbf{z})$ and $\mathbf{x}$ using gradient descent for a frozen $\mathcal{G}$ (Abdal et al. 2021). In both cases, $\mathbf{z}$ is next processed by INF, which gives us its factorized representation $(\mathbf{c}, \mathbf{s}) = \mathcal{F}^{-1}(\mathbf{z})$. In this representation, we can modify any labeled variable c_i and map the resulting vector back through $\mathcal{F}$ and $\mathcal{G}$ as in the generative case.

Observe that PluGeN does not need to know what are the values of labeled attributes when it modifies attributes of existing examples. Given a latent representation $\mathbf{z}$, PluGeN maps it through $\mathcal{G}^{-1}$, which gives us the factorization into labeled and unlabeled attributes. In contrast, existing solutions based on conditional INF, e.g. StyleFlow (Abdal et al. 2021), have to determine all labels before passing $\mathbf{z}$ through INF as they represent the conditioning factors. In consequence, these models involve additional classifiers for labeled attributes.

Experiments

To empirically evaluate the properties of PluGeN, we combine it with GAN and VAE architectures to manipulate attributes of image data. Moreover, we present a practical use-case of chemical molecule modeling using CharVAE. Due to the page limit, we included architecture details and additional results in the supplementary materials.

GAN backbone First, we consider the state-of-the-art StyleGAN architecture (Karras, Laine, and Aila 2019), which was trained on Flickr-Faces-HQ (FFHQ) containing 70 000 high-quality images of resolution 1024×1024 . The Microsoft Face API was used to label 8 attributes in each image (gender, pitch, yaw, eyeglasses, age, facial hair, expression, and baldness).

PluGeN is instantiated using NICE flow model (Dinh, Krueger, and Bengio 2014) that operates on the latent vectors $\mathbf{w} \in \mathbb{R}^{512}$ sampled from the $\mathbf{W}$ space of the StyleGAN. As a baseline, we select StyleFlow (Abdal et al. 2021), which is currently one of the state-of-the-art models for controlling the generation process of StyleGAN. In contrast to PluGeN, StyleFlow uses the conditional continuous INF to operate on the latent codes of StyleGAN, where the conditioning factor corresponds to the labeled attributes. For eval-

uation, we modify one of 5 attributes² and verify the success of this operation using the prediction accuracy returned by Microsoft Face API. The quality of images is additionally assessed by calculating the standard Fréchet Inception Distance (FID) (Heusel et al. 2017).

Figure 1 (first page) and 4 present the effects of how PluGeN and StyleFlow manipulate images sampled by StyleGAN. It is evident that PluGeN can switch the labels to opposite values as well as gradually change their intensities. At the same time, the requested modifications do not influence the remaining attributes leaving them unchanged. One can observe that the results produced by StyleFlow are also acceptable, but the modification of the requested attribute implies the change of other attributes. For example, increasing the intensity of "baldness" changes the type of glasses, or turning the head into right makes the woman look straight.

The above qualitative evaluation is supported by the quantitative assessment presented in Table 1. As can be seen, StyleFlow obtains a better FID score, while PluGeN outperforms StyleFlow in terms of accuracy. Since FID compares the distribution of generated and real images, creating images with uncommon combinations of attributes that do not appear in a training set may be scored lower, which can explain the relation between accuracy and FID obtained by PluGeN and StyleFlow. In consequence, FID is not an adequate metric for measuring the quality of arbitrary image manipulations considered here, because it is too closely tied to the distribution of input images.

It is worth mentioning that PluGeN obtains these very good results using NICE model, which is the simplest type of INFs. In contrast, StyleFlow uses continuous INF, which is significantly more complex and requires using an ODE solver leading to unstable training. Moreover, to modify even a single attribute, StyleFlow needs to determine the values of all labels, since they represent the conditioning factors of INF. In consequence, every modification requires applying an auxiliary classifier to predict all image labels. The usage of PluGeN is significantly simpler, as subsequent coordinates in the latent space of INF correspond to the labeled attributes and they are automatically determined by PluGeN. Finally, our approach is less computationally expensive as we verified that, using the same hardware, PluGeN can be trained 3 times faster than StyleFlow and is around 100 times faster in inference.

Image manipulation on VAE backbone In the following experiment, we show that PluGeN can be combined with autoencoder models to effectively manipulate image attributes. We use CelebA database, where every image of the size 256×256 is annotated with 40 binary labels.

We compare PluGeN to MSP (Li et al. 2020), a strong baseline, which uses a specific loss for disentangling the latent space of VAE. Following the idea of StyleFlow, we also consider a conditional INF attached to the latent space of pre-trained VAE (referred to as cFlow), where conditioning factors correspond to the labeled attributes. The architecture of the base VAE and the evaluation protocol were taken from

²The remaining 3 attributes (age, pitch, yaw) are continuous and it is more difficult to assess their modifications.

Requested value	PluGeN	StyleFlow
female	0.95	0.95
male	0.92	0.87
no-glasses	1.00	0.99
glasses	0.90	0.70
not-bald	1.00	1.00
bald	0.53	0.54
no-facial-hair	1.00	1.00
facial-hair	0.72	0.65
no-smile	0.99	0.92
smile	0.96	0.99
Average Acc	0.90	0.86
Average FID	46.51	32.59

Table 1: Accuracy and FID scores of attributes modification using StyleGAN backbone.

the original MSP paper. More precisely, for every input image, we manipulate the values of two attributes (we inspect 20 combinations in total). The success of the requested manipulation is verified using a multi-label ResNet-56 classifier trained on the original CelebA dataset.

The sample results presented in Figure 5 demonstrate that PluGeN attached to VAE produces high-quality images satisfying the constraints imposed on the labeled attributes. The quantitative comparison shown in Table 2 confirms that PluGeN is extremely efficient in creating uncommon combinations of attributes, while cFlow performs well only for the usual combinations. At the same time, the quality of images produced by PluGeN and MSP is better than in the case of cFlow. Although both PluGeN and MSP focus on disentangling the latent space of the base model, MSP has to be trained jointly with the base VAE model and it was designed only to autoencoder models. In contrast, PluGeN is a separate module, which can be attached to arbitrary pre-trained models. Due to the use of invertible neural networks, it preserves the reconstruction quality of the base model, while adding manipulation functionalities. In the following experiment, we show that PluGeN also performs well at generating entirely new images, which is not possible using MSP.

Image generation with VAE backbone In addition to manipulating the labeled attributes of existing images, PluGeN generates new examples with desired attributes. To verify this property, we use the same VAE architecture as before trained on CelebA dataset. The baselines include cFlow and two previously introduced methods for multi-label conditional generation³: cVAE (Yan et al. 2016) and Δ -GAN (Gan et al. 2017). We exclude MSP from the comparison because it cannot generate new images, but only manipulate the attributes of existing ones (see supplementary materials for a detailed explanation).

Figure 6 presents sample results of image generation with the specific conditions. In each row, we fix the style variables s and vary the label variables c in each column, generating

³For cVAE and Δ -GAN we use images of the size 64×64 following their implementations.

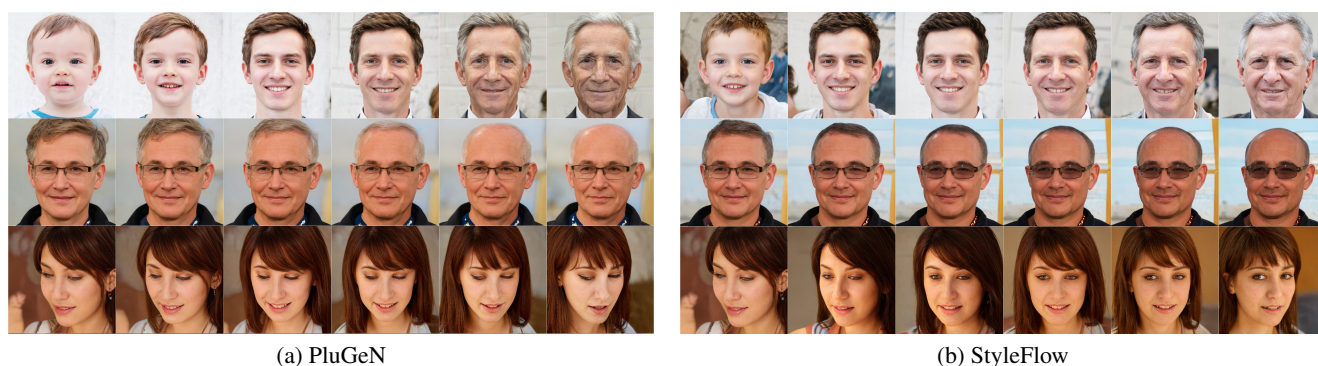


Figure 4: Gradual modification of attributes (age, baldness, and yaw, respectively) performed on the StyleGAN latent codes.

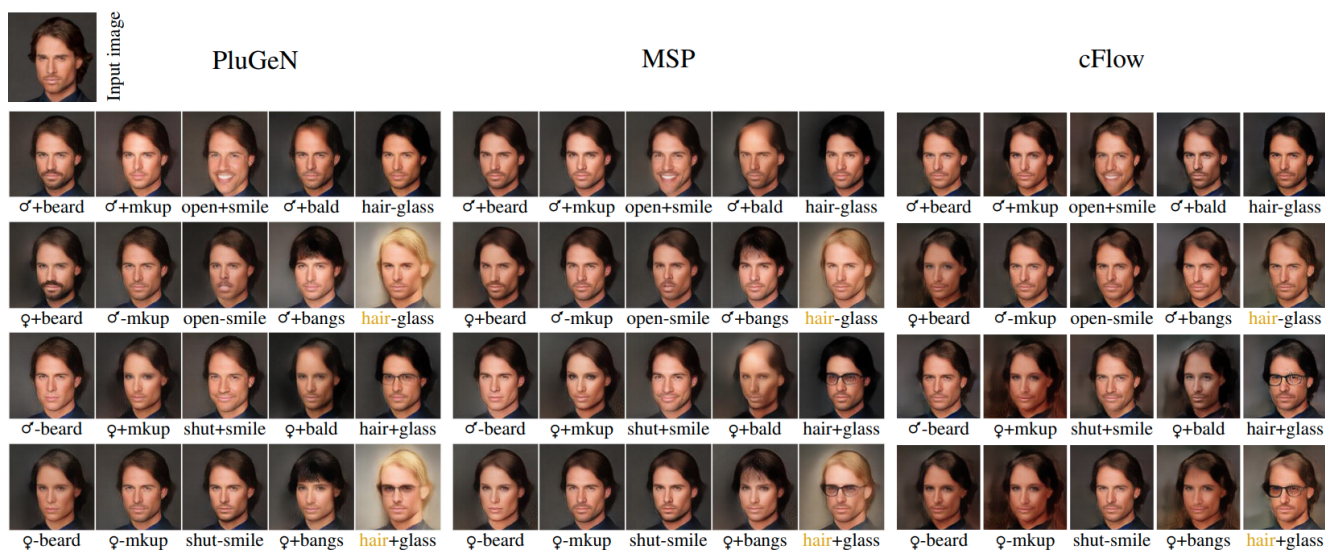


Figure 5: Examples of image attribute manipulation using VAE backbone.

the same person but with different characteristics such as hair color, eyeglasses, etc. Although cVAE manages to modify the attributes, the quality of obtained samples is poor, while Δ -GAN falls completely out of distribution. PluGeN and cFlow generate images of similar quality, but only PluGeN is able to correctly manipulate the labeled attributes. The lower quality of generated images is caused by the poor generation abilities of VAE backbone, which does not work well with high dimensional images (see supplementary materials for a discussion). For this reason, it is especially notable that PluGeN can improve the generation performance of the backbone model in contrast to MSP.

Disentangling the attributes The attributes in the CelebA dataset are strongly correlated and at times even contradictory, e.g. attributes 'bald' and 'blond hair' cannot both be present at the same time. In this challenging task, we aim to disentangle the attribute space as much as it is possible to allow for generating examples with arbitrary combinations of attributes. For this purpose, we sample the conditional variables c_i independently, effectively ignoring the underlying correlations of attributes, and use them to generate im-

ages. Since the attributes in the CelebA dataset are often imbalanced (e.g. only in 6.5% of examples the person wears glasses), we calculate F1 and AUC scores for each attribute.

The quantitative analysis of the generated images presented in Table 3 confirms that PluGeN outperforms the rest of the methods with respect to classification scores. The overall metrics are quite low for all approaches, which is due to the difficulty of disentanglement mentioned above, as well as the inaccuracy of the ResNet attribute classifier. Deep learning models often fail when the correlations in the training data are broken, e.g. the classifier might use the presence of a beard to predict gender, thus introducing noise in the evaluation (Beery, Horn, and Perona 2018).

Chemical molecules modeling Finally, we present a practical use-case, in which we apply PluGeN to generate chemical molecules with the requested properties. As a backbone model, we use CharVAE (Gómez-Bombarelli et al. 2018), which is a type of recurrent network used for processing SMILES (Weininger 1988), a textual representation of molecules. It was trained on ZINC 250k database (Sterling and Irwin 2015) of commercially available chemical com-

Requested value	PluGeN	MSP	cFlow
male x beard	0.80	0.79	0.85
female x beard	0.59	0.33	0.31
male x no-beard	0.88	0.92	0.91
female x no-beard	0.85	0.82	0.95
male x makeup	0.44	0.43	0.29
male x no-makeup	0.72	0.92	0.96
female x makeup	0.42	0.41	0.58
female x no-makeup	0.55	0.40	0.85
smile x open-mouth	0.97	0.99	0.79
no-smile x open-mouth	0.79	0.82	0.77
smile x calm-mouth	0.84	0.91	0.72
no-smile x calm-mouth	0.96	0.97	0.99
male x bald	0.26	0.41	0.34
male x bangs	0.58	0.74	0.45
female x bald	0.19	0.13	0.39
female x bangs	0.52	0.49	0.60
no-glasses x black-hair	0.92	0.93	0.74
no-glasses x golden-hair	0.92	0.91	0.81
glasses x black-hair	0.76	0.90	0.58
glasses x golden-hair	0.75	0.85	0.61
Average Acc	0.69	0.70	0.67
Average FID	28.07	30.67	39.68

Table 2: Accuracy and FID scores of image manipulation performed on the VAE backbone.

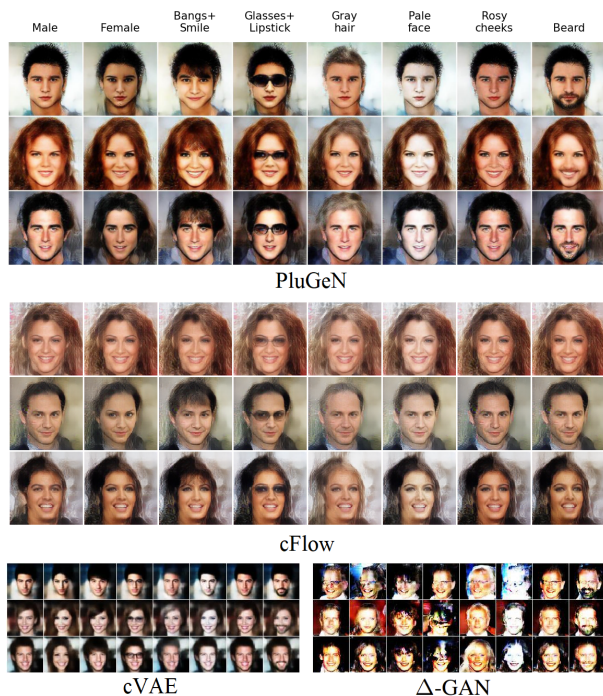


Figure 6: Examples of conditional generation using VAE backbone. Each row contains the same person (style variables) with modified attributes (label variables).

	PluGeN	cFlow	Δ -GAN	cVAE
F1	0.44	0.29	0.39	0.39
AUC	0.76	0.68	0.70	0.73

Table 3: Results of the independent conditional generation using VAE backbone.

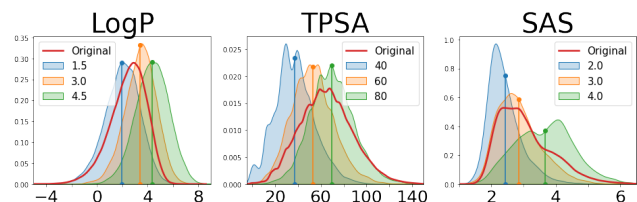


Figure 7: Distribution of attributes of generated molecules, together with distribution for the training dataset. Each color shows the value of a labeled attribute that was used for generation. PluGeN is capable of moving the density of generated molecules’ attributes towards the desired value. The average of every distribution is marked with a vertical line.

pounds. For every molecule, we model 3 physio-chemical continuous (not binary) labels: logP, SAS, TPSA, which values were calculated using RDKit package⁴. Additional explanations and more examples are given in the supplementary materials.

First, we imitate a practical task of de novo design (Olivecrona et al. 2017; Popova, Isayev, and Tropsha 2018), where we force the model to generate new compounds with desirable properties. For every attribute, we generate 25k molecules with 3 different values: for logP we set the label of generated molecules to: 1.5, 3.0, 4.5; for TPSA we set generated labels to: 40, 60, 80; for SAS we set them to: 2.0, 3.0, 4.0, which gives 9 scenarios in total. From density plots of labels of generated and original molecules presented in Figure 7, we can see that PluGeN changes the distribution of values of the attributes and moves it towards the desired value. A slight discrepancy between desired and generated values may follow from the fact that values of labeled attributes were sampled independently, which could make some combinations physically contradictory.

Next, we consider the setting of lead optimization (Jin et al. 2019; Maziarka et al. 2020), where selected compounds are improved to meet certain criteria. For this purpose, we encode a molecule into the latent representation of INF and force PluGeN to gradually increase the value of logP by 3 and decode the resulting molecules. The obtained molecules together with their logP are shown in Figure 8. As can be seen, PluGeN generates molecules that are structurally similar to the initial one, however with optimized desired attributes.

Obtained results show that PluGeN is able to model the physio-chemical molecular features, which is a non-trivial task that could speed up a long and expensive process of

⁴<https://www.rdkit.org/>

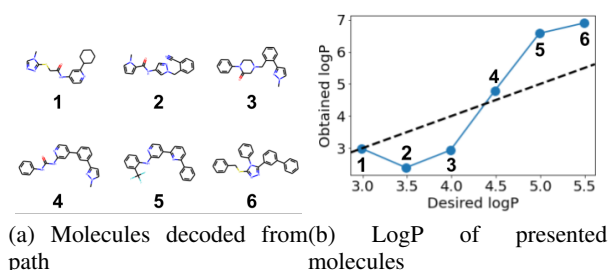


Figure 8: Molecules obtained by the model during an optimization phase (left side), and their LogP (right side).

designing new drugs.

Conclusion

We proposed a novel approach for disentangling the latent space of pre-trained generative models, which works perfectly for generating new samples with desired conditions as well as for manipulating the attributes of existing examples. In contrast to previous works, we demonstrated that PluGeN performs well across diverse domains, including chemical molecule modeling, and can be combined with various architectures, such as GANs and VAEs backbones.

Acknowledgements

The research of M. Wołczyk was supported by the Foundation for Polish Science co-financed by the European Union under the European Regional Development Fund in the POIR.04.04.00-00-14DE/18-00 project carried out within the Team-Net programme. The research of M. Proszewska, Ł. Maziarka, M. Zieba and R. Kurczab was supported by the National Science Centre (Poland), grant no. 2018/31/B/ST6/00993, 2019/35/N/ST6/02125, 2020/37/B/ST6/03463 and the Polish National Centre for Research and Development (Grant LIDER/37/0137/L-9/17/NCBR/2018). The research of M. Śmieja was funded by the Priority Research Area DigiWorld under the program Excellence Initiative — Research University at the Jagiellonian University in Kraków.

Ethical Impact

We did not identify ethical issues concerning our work, as we do not collect data, and we do not foresee malicious applications or societal harm. However, we believe that disentangling factors of variation can have a positive effect on reducing unjust correlations in the data. For example, even though CelebA dataset contains twice as many old men as old women, our method can generate an equal proportion of samples from those classes, thus avoiding amplifying bias in the data.

References

Abdal, R.; Zhu, P.; Mitra, N. J.; and Wonka, P. 2021. Styleflow: Attribute-conditioned exploration of stylegan-

- generated images using conditional continuous normalizing flows. *ACM Transactions on Graphics (TOG)*, 40(3): 1–21.
- Beery, S.; Horn, G. V.; and Perona, P. 2018. Recognition in Terra Incognita. In *ECCV*.
- Bengio, Y.; Courville, A.; and Vincent, P. 2013. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8): 1798–1828.
- Brakel, P.; and Bengio, Y. 2017. Learning independent features with adversarial nets for non-linear ica. *arXiv preprint arXiv:1710.05050*.
- Brock, A.; Donahue, J.; and Simonyan, K. 2018. Large Scale GAN Training for High Fidelity Natural Image Synthesis. In *International Conference on Learning Representations*.
- Brown, T. B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; Agarwal, S.; Herbert-Voss, A.; Krueger, G.; Henighan, T.; Child, R.; Ramesh, A.; Ziegler, D. M.; Wu, J.; Winter, C.; Hesse, C.; Chen, M.; Sigler, E.; Litwin, M.; Gray, S.; Chess, B.; Clark, J.; Berner, C.; McCandlish, S.; Radford, A.; Sutskever, I.; and Amodei, D. 2020. Language Models are Few-Shot Learners. In Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M.; and Lin, H., eds., *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.
- Chen, R. T.; Li, X.; Grosse, R.; and Duvenaud, D. 2019. Isolating Sources of Disentanglement in VAEs. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, 2615–2625.
- Chen, X.; Duan, Y.; Houthoofd, R.; Schulman, J.; Sutskever, I.; and Abbeel, P. 2016. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. *arXiv preprint arXiv:1606.03657*.
- Choi, Y.; Uh, Y.; Yoo, J.; and Ha, J.-W. 2020. Stargan v2: Diverse image synthesis for multiple domains. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8188–8197.
- Dinh, L.; Krueger, D.; and Bengio, Y. 2014. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*.
- Gan, Z.; Chen, L.; Wang, W.; Pu, Y.; Zhang, Y.; Liu, H.; Li, C.; and Carin, L. 2017. Triangle generative adversarial networks. *arXiv preprint arXiv:1709.06548*.
- Gao, Y.; Wei, F.; Bao, J.; Gu, S.; Chen, D.; Wen, F.; and Lian, Z. 2021. High-Fidelity and Arbitrary Face Editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 16115–16124.
- Gómez-Bombarelli, R.; Wei, J. N.; Duvenaud, D.; Hernández-Lobato, J. M.; Sánchez-Lengeling, B.; Sheberla, D.; Aguilera-Iparraguirre, J.; Hirzel, T. D.; Adams, R. P.; and Aspuru-Guzik, A. 2018. Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science*, 4(2): 268–276.
- Härkönen, E.; Hertzmann, A.; Lehtinen, J.; and Paris, S. 2020. Ganspace: Discovering interpretable gan controls. *arXiv preprint arXiv:2004.02546*.

- He, Z.; Zuo, W.; Kan, M.; Shan, S.; and Chen, X. 2019. Attgan: Facial attribute editing by only changing what you want. *IEEE Transactions on Image Processing*, 28(11): 5464–5478.
- Heusel, M.; Ramsauer, H.; Unterthiner, T.; Nessler, B.; and Hochreiter, S. 2017. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. In Guyon, I.; von Luxburg, U.; Bengio, S.; Wallach, H. M.; Fergus, R.; Vishwanathan, S. V. N.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, 6626–6637.
- Higgins, I.; Matthey, L.; Pal, A.; Burgess, C.; Glorot, X.; Botvinick, M.; Mohamed, S.; and Lerchner, A. 2017. beta-vae: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*.
- Jang, E.; Gu, S.; and Poole, B. 2016. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*.
- Jin, W.; Barzilay, R.; and Jaakkola, T. 2018. Junction tree variational autoencoder for molecular graph generation. In *International Conference on Machine Learning*, 2323–2332. PMLR.
- Jin, W.; Yang, K.; Barzilay, R.; and Jaakkola, T. 2019. Learning multimodal graph-to-graph translation for molecular optimization. *International Conference on Learning Representations*.
- Jo, Y.; and Park, J. 2019. Sc-fegan: Face editing generative adversarial network with user’s sketch and color. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 1745–1753.
- Kang, S.; and Cho, K. 2018. Conditional molecular design with deep generative models. *Journal of chemical information and modeling*, 59(1): 43–52.
- Karras, T.; Laine, S.; and Aila, T. 2019. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4401–4410.
- Kaya, Y.; Hong, S.; and Dumitras, T. 2019. Shallow-Deep Networks: Understanding and Mitigating Network Overthinking. In Chaudhuri, K.; and Salakhutdinov, R., 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*, 3301–3310. PMLR.
- Kim, H.; and Mnih, A. 2018. Disentangling by factorising. In *International Conference on Machine Learning*, 2649–2658. PMLR.
- Kingma, D. P.; and Dhariwal, P. 2018. Glow: Generative flow with invertible 1x1 convolutions. *arXiv preprint arXiv:1807.03039*.
- Kingma, D. P.; Rezende, D. J.; Mohamed, S.; and Welling, M. 2014. Semi-supervised learning with deep generative models. *arXiv preprint arXiv:1406.5298*.
- Klys, J.; Snell, J.; and Zemel, R. 2018. Learning latent subspaces in variational autoencoders. *arXiv preprint arXiv:1812.06190*.
- Kodali, N.; Abernethy, J.; Hays, J.; and Kira, Z. 2017. On convergence and stability of gans. *arXiv preprint arXiv:1705.07215*.
- Koperski, M.; Konopczynski, T.; Nowak, R.; Semberecki, P.; and Trzcinski, T. 2020. Plugin Networks for Inference under Partial Evidence. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2883–2891.
- Kumar, A.; Sattigeri, P.; and Balakrishnan, A. 2017. Variational inference of disentangled latent concepts from unlabeled observations. *arXiv preprint arXiv:1711.00848*.
- Lample, G.; Zeghidour, N.; Usunier, N.; Bordes, A.; Denoyer, L.; and Ranzato, M. 2017. Fader networks: Manipulating images by sliding attributes. *arXiv preprint arXiv:1706.00409*.
- Li, X.; Lin, C.; Li, R.; Wang, C.; and Guerin, F. 2020. Latent space factorisation and manipulation via matrix subspace projection. In *International Conference on Machine Learning*, 5916–5926. PMLR.
- Liu, R.; Liu, Y.; Gong, X.; Wang, X.; and Li, H. 2019. Conditional adversarial generative flow for controllable image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 7992–8001.
- Locatello, F.; Bauer, S.; Lucic, M.; Raetsch, G.; Gelly, S.; Schölkopf, B.; and Bachem, O. 2019. Challenging common assumptions in the unsupervised learning of disentangled representations. In *international conference on machine learning*, 4114–4124. PMLR.
- Maziarka, Ł.; Pocha, A.; Kaczmarczyk, J.; Rataj, K.; Danel, T.; and Warchoł, M. 2020. Mol-CycleGAN: a generative model for molecular optimization. *Journal of Cheminformatics*, 12(1): 1–18.
- Mirza, M.; and Osindero, S. 2014. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*.
- Nitzan, Y.; Bermano, A.; Li, Y.; and Cohen-Or, D. 2020. Disentangling in latent space by harnessing a pretrained generator. *arXiv preprint arXiv:2005.07728*, 2(3).
- Olivecrona, M.; Blaschke, T.; Engkvist, O.; and Chen, H. 2017. Molecular de-novo design through deep reinforcement learning. *Journal of cheminformatics*, 9(1): 1–14.
- Park, T.; Liu, M.-Y.; Wang, T.-C.; and Zhu, J.-Y. 2019. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2337–2346.
- Perarnau, G.; Van De Weijer, J.; Raducanu, B.; and Álvarez, J. M. 2016. Invertible conditional gans for image editing. *arXiv preprint arXiv:1611.06355*.
- Popova, M.; Isayev, O.; and Tropsha, A. 2018. Deep reinforcement learning for de novo drug design. *Science advances*, 4(7): eaap7885.
- Rebuffi, S.; Bilen, H.; and Vedaldi, A. 2017. Learning multiple visual domains with residual adapters. In Guyon, I.; von

Luxburg, U.; Bengio, S.; Wallach, H. M.; Fergus, R.; Vishwanathan, S. V. N.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, 506–516.

Shen, Y.; Yang, C.; Tang, X.; and Zhou, B. 2020. Interfacegan: Interpreting the disentangled face representation learned by gans. *IEEE transactions on pattern analysis and machine intelligence*.

Sohn, K.; Lee, H.; and Yan, X. 2015. Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems*, 28: 3483–3491.

Sorrenson, P.; Rother, C.; and Köthe, U. 2020. Disentanglement by nonlinear ica with general incompressible-flow networks (gin). *arXiv preprint arXiv:2001.04872*.

Spurek, P.; Nowak, A.; Tabor, J.; Maziarka, Ł.; and Jastrzebski, S. 2020. Non-linear ICA based on Cramer-Wold metric. In *International Conference on Neural Information Processing*, 294–305. Springer.

Sterling, T.; and Irwin, J. J. 2015. ZINC 15–ligand discovery for everyone. *Journal of chemical information and modeling*, 55(11): 2324–2337.

Tewari, A.; Elgharib, M.; Bernard, F.; Seidel, H.-P.; Pérez, P.; Zollhöfer, M.; and Theobalt, C. 2020. Pie: Portrait image embedding for semantic control. *ACM Transactions on Graphics (TOG)*, 39(6): 1–14.

Tov, O.; Alaluf, Y.; Nitzan, Y.; Patashnik, O.; and Cohen-Or, D. 2021. Designing an encoder for stylegan image manipulation. *ACM Transactions on Graphics (TOG)*, 40(4): 1–14.

Wang, H.-P.; Yu, N.; and Fritz, M. 2021. Hijack-gan: Unintended-use of pretrained, black-box gans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 7872–7881.

Weininger, D. 1988. SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *Journal of chemical information and computer sciences*, 28(1): 31–36.

Wołczyk, M.; Wójcik, B.; Bałazy, K.; Podolak, I.; Tabor, J.; Śmieja, M.; and Trzciński, T. 2021. Zero Time Waste: Recycling Predictions in Early Exit Neural Networks. *arXiv preprint arXiv:2106.05409*.

Yan, X.; Yang, J.; Sohn, K.; and Lee, H. 2016. Attribute2image: Conditional image generation from visual attributes. In *European Conference on Computer Vision*, 776–791. Springer.

Zhou, W.; Xu, C.; Ge, T.; McAuley, J. J.; Xu, K.; and Wei, F. 2020. BERT Loses Patience: Fast and Robust Inference with Early Exit. In Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M.; and Lin, H., eds., *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.