



Ca' Foscari University  
Department of Environmental Sciences,  
Informatics and Statistics

# **Dissecting Continual Learning** **a Structural and Data Analysis**

Ph.D. Thesis - Computer Science  
XXXIV Cycle

Submitted by:  
[Pelosin Francesco](#)

Supervisor:  
[Prof. Torsello Andrea](#)

Venice, Italy - March, 2022



---

## Abstract

Deep Learning aims to discover how artificial neural networks learn the rich internal representations required for difficult tasks such as recognizing objects or understanding language. This hard question is still unanswered although we are constantly improving the performance of such systems spanning from computer vision problems to natural language processing tasks. Continual Learning (CL) is a field dedicated in devising algorithms able to achieve lifelong learning by overcoming the knowledge disruption of previously acquired concepts, a phenomenon that affects deep learning architectures and that goes by the name of catastrophic forgetting. Currently, deep learning methods can achieve outstanding results when the data modeled does not undergo a considerable distribution shift in subsequent learning sessions, but as we expose the systems to such incremental setting, performance abruptly drops due to catastrophic forgetting. As the data generated in the world is continuously increasing, the demand to model such streams in a sequential fashion is increasing. As such, devising techniques to prevent knowledge corruption in neural networks is fundamental. Overcoming such limitations would allow us to build truly intelligent systems showing adaptability and human-like quality. Secondly, it would allow us to overcome the limitation, and onerous aspect, of retraining the architectures from scratch with the updated data. Such drawback comes from how deep neural networks learn, that is, they require several parameter updates to learn any given concept. This is also the exact reason why catastrophic forgetting happens, as we learn new concepts we overwrite old ones, while a truly intelligent system would show a stability-plasticity optimal trade-off. In this thesis, we first describe the background needed to understand continual learning in the computer vision realm. We do so with the introduction of a notation and a formal description of the problem. Then, we will introduce several CL setting variants and main solution categories proposed in the literature, along with an analysis of the state-of-the-art. We then first analyze one of the baseline approaches to continual learning and discover that in rehearsal-based techniques the quantity of data stored is a more important factor than the quality of memorized data. This trade-off surprisingly holds even for impressively high compression rates of the data. Secondly, this thesis proposes one of the early works on the study of incremental learning on vision transformer architectures (ViTs). In particular, we will compare functional, weight, and attention regularization approaches for the challenging rehearsal-free CL. We then propose an asymmetric loss variant inspired by PODNet, achieving good capabilities in terms of plasticity. Among these contributions, we propose a simple, but effective baseline for off-the-shelf continual learning exploiting pretrained models and discuss its extension to unsupervised continual learning, a topic that deserves further attention from the community. As the final work, we introduce a novel algorithm able to explore the environment through unsupervised visual pattern discovery. We then provide a conclusion and discuss further developments and promising paths to be followed by the CL research.



---

## Akwnnowledgments

First, I would like to express my gratitude to my supervisor Andrea Torsello, for all the deep insights and for welcoming me to pursue this research with him.

Secondly, I would like to thank all the people that I encountered throughout these years, especially colleagues and friends that I met, a personal acknowledgment to Alessandro, Seyum, Fatima, and also the friends I met in Spain Hectór, Laura, and Albin. I would also like to say thank everyone that loved me during this period, you gave me the strength to carry on this tough journey!

Lastly, I would say that I learned a lot during these years, and the force that moved me to pursue a Ph.D., is the same force that allows us to expand and look for answers, to find meanings, and to unfold *something beautiful*.

‘‘You’re pretty good’’



# Contents

<b>1</b>	<b>Introduction</b>	<b>9</b>
<b>2</b>	<b>Background and Motivation</b>	<b>13</b>
2.1	Artificial vs Natural Intelligence . . . . .	14
2.2	What is Continual Learning? . . . . .	17
2.2.1	Stability-Plasticity Dilemma . . . . .	19
2.2.2	Catastrophic Forgetting . . . . .	20
2.2.3	A Visual Example . . . . .	20
<b>3</b>	<b>Continual Learning Framework</b>	<b>25</b>
3.1	Definition and Settings . . . . .	26
3.1.1	Online CL vs Offline CL . . . . .	28
3.1.2	Task-Incremental vs Class-Incremental . . . . .	29
3.2	Baselines . . . . .	30
3.2.1	Cumulative . . . . .	31
3.2.2	Finetuning . . . . .	32

3.3	State-of-the-art . . . . .	33
3.3.1	Structural-based . . . . .	33
3.3.2	Regularization-based . . . . .	34
3.3.3	Rehearsal-based . . . . .	35
<b>4</b>	<b>Works</b>	<b>37</b>
4.1	Smaller is Better: An Analysis of Instance Quantity/Quality Trade-off in Rehearsal-based Continual Learning . . . . .	38
4.2	Towards Exemplar-Free Continual Learning in Vision Transformers: an Account of Attention, Functional and Weight Regularization . . . . .	58
4.3	Simpler is Better: off-the-shelf Continual Learning through Pretrained Backbones . . . . .	76
4.4	Unsupervised Semantic Discovery through Visual Patterns detection	85
<b>5</b>	<b>Conclusions</b>	<b>99</b>



# Chapter 1

## Introduction

*"The measure of intelligence is the ability to change"*

*- Albert Einstein*

The interconnections among entities in our world are growing. Along with this fact, the ability to keep track and record such data has accordingly increased. The need for systems that can cope with such phenomena is essential. Deep Learning (DL) revealed itself to be a powerful weapon to model such complex streams, especially in Computer Vision and Natural Language Processing fields. The advent of DL unlocked the ability to develop outstanding technologies that can directly impact our lives. Self-driving cars are one example. Unfortunately not always the impact is positive, if not properly controlled. Therefore, the need for systems that show generalization abilities and can cope with unexpected scenarios, is nowadays essential. To this end, we also need responsive machines, that can be trained to quickly learn new concepts with low resource consumption. In fact, what happens if the stream of data encountered by a deep learning model changes its quality over time? This particular question is tackled by Continual Learning (CL) whose aim is to develop lifelong learning machines, unlocking fast adaptability to new environments.

Modern deep learning methods for computer vision adapt themselves only to the manifold they are trained on. Instead, we need to devise models which are plastic enough to generalize to distributional shifts in the data and do not require complete retraining. This challenge would be solved if training Deep Learning models would not be such a delicate process affected by unexpected drawbacks. In fact, when we introduce the notion of learning through time and expose the system to face incremental tasks of different nature, things can get really complicated.

One of the drawbacks of incrementally learning is the so-called catastrophic forgetting, where the system is subject to an abrupt deterioration of past knowledge whenever asked to learn new concepts. This big limitation is broadly studied in continual learning. To approach this delicate subject, in this thesis, we start by gently introducing some basic differences between artificial and natural intelligence. Here, we clarify some operative differences between artificial neural networks and some basic brain mechanisms arising from neuroscience. Then, we informally introduce the notion of continual learning and discuss the stability-plasticity dilemma along with the phenomenon of catastrophic forgetting of artificial neural networks. We proceed by introducing a more formal definition of incremental learning along with its fine-grained inclinations. Before moving to the contributions we introduce a brief overview of the state-of-the-art and define the main baselines which act as lower and upper bounds for continual learning methodologies.

We step into the major contributions by focusing on rehearsal systems, a family of methods that exploit cache memories to replay previous knowledge. Here, we study how the compression of stored rehearsal data impacts the performance of the model. Tackling the memory side of CL, we provide a quality/quantity analysis through

the usage of several compression schemes. We consider also extreme compression rates, providing some insights. On top of that, we consider continual learning under low-resource constraints through the usage of random projections and, in particular, Extreme Learning Machines.

To follow, as a second major contribution, we are among the first to investigate Vision Transformers in continual learning. In particular, we analyze several regularization schemes for ViTs, providing a first envision of rehearsal-free CL. We consider weight, functional and attentional regularizations, being the latter unexplored before, we carefully study the application of regularizations to specific parts of the self-attention mechanism. As a side contribution we introduce a new asymmetric loss variant inspired by a contemporary continual learning method (PODNet) principled by the observation that new attention should not penalize the acquisition of new knowledge.

We then further clarify the usage of pretrained models in continual learning through an experimental segment. We compare fully pretrained CNNs and Vision Transformers in several incremental benchmarks. We provide a clear simple baseline that requires few KBytes to operate and does not perform parameter updates. Being simple and effective, we discuss its extension to the unsupervised realm. Here we consider further extensions for future works.

Along with these three contributions, we also study the ability of a system to autonomously discover new visual patterns, a notion embedded in an optimal incremental learner. We, therefore, provide a simple unsupervised pipeline able to discover semantic patterns on different visual scales. Finally, we conclude by wrapping up our perspectives on the main aforementioned challenges.

As a final note, we hope this thesis finds a meaningful purpose in the CL community, contributing to the development of Continual Learning and Computer Vision research.

## Contibution Prefaction

In this thesis we included some papers developed while pursuing the Ph.D.. The main contributions have been reported in Chapter 4. The chapter holds the outcome of several collaborations and with the following list we report the names of the authors and the venues where the works have been submitted:

- The work reported in Section 4.1, has been accepted as oral poster to *IJCNN 2022*. The authors who contributed to the work are (in order): Francesco Pelosin and Andrea Torsello from Ca' Foscari University of Venice
- The work reported in Section 4.2 is the outcome of the collaboration of the research period abroad and has been accepted as poster to the *Continual Learning Workshop of CVPR 2022*. The authors who contributed to the work are (in order): Francesco Pelosin, Ca' Foscari University of Venice (Equal Contrib); Saurav Jha, University of New South Wales, Australia (Equal Contrib); Andrea Torsello, Ca' Foscari University of Venice, Italy; Bogdan Raducanu and Joost van de Weijer from Computer Vision Center, Spain.
- The work reported in Section 4.3 has been accepted as poster to the *Transformers for Vision Workshop of CVPR 2022* and it is single authored by Francesco Pelosin.
- The work reported in Section 4.4, has been accepted to the *S+SSPR 2020*. The authors who contributed to the work are (in order): Francesco Pelosin; Andrea Gasparetto; Andrea Albarelli and Andrea Torsello, Ca Foscari University of Venice, Italy

## **Chapter 2**

# **Background and Motivation**

## 2.1 Artificial vs Natural Intelligence

Although the recent developments and great achievements of the field of Artificial Intelligence, the fundamental nature of Artificial Neural Networks (ANNs) might still be a coarse approximation of how our biological brains work. With the mathematical introduction by [McCulloch and Pitts, 1943] and the introduction of the “Perceptron” by [Rosenblatt, 1958], which constitutes the smallest unit that form a ANN, we shaped our modeling of intelligence. An artificial neuron can be described as a cumulative summation of multiplications over some weights followed by a non-linear function.

Then, after the introduction of the famous Multy Layer Perceptrons (MLPs) the structure of ANNs has not changed much: we work in a connectionist paradigm where the learning happens through a distributed signal activity via connections among artificial neurons. In particular, the learning occurs by modifying connection strengths based on experience, this modification procedure has a particular name and it is the so-called *backpropagation* algorithm whose discovery can be traced back to [Rumelhart et al., 1986] but with some earlier works by [Linnainmaa, 1976] (as an M.Sc. Thesis) as pointed in [Schmidhuber, 2014].

The success of **connectionists models** span over different fields: Convolutional Neural Networks (CNN) for Computer Vision (CV) [He et al., 2016], Language Models for Natural Language Processing (NLP) [Devlin et al., 2019], Deep Q-Learning Networks (DQN) for Reinforcement Learning [Agarwal et al., 2020], Generative Audio Models for Audio [van den Oord et al., 2016] and Graph Convolutional Networks (GCN) for graph data [Kipf and Welling, 2017].

Connectionist models are a composition of several layers of artificial neurons, followed by a non-linearity. There are several types of layers each with its peculiarity. For example with the introduction of Batch Normalization [Ioffe and Szegedy, 2015] we allowed the networks to achieve faster training. The introduction of some specialized units often allowed to excel in particular fields such as the convolutional operation [LeCun et al., 1998] for Computer Vision tasks and the Self-Attention mechanism in Natural Language Processing. [Vaswani et al., 2017], although the attention mechanism has achieved tremendous achievements in vision tasks thanks to [Dosovitskiy et al., 2021] and its introduction of Visual Transformers. Nowadays there is still no perfect mechanism/model for each scenario because we are still in the process of discovering how learning happens. For sure in the future, we might see other methodologies working in fields where they are not born from.

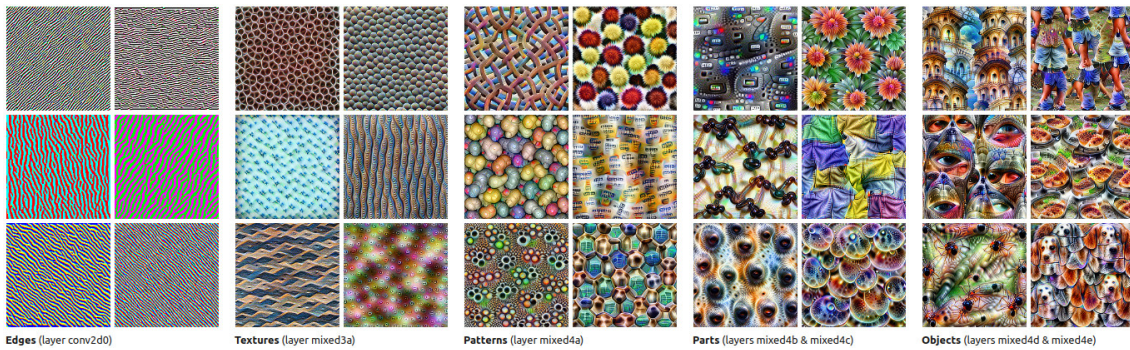


Figure 2.1: Feature visualization of GoogLeNet [Szegedy et al., 2015], trained on the ImageNet [Russakovsky et al., 2015] dataset. Concepts in early layers are reported on left while concepts of last layers are on the right. The image is taken from [Olah et al., 2017]

While attention-based models spread the knowledge, and feature representations, uniformly across the layers [Raghu et al., 2021], in classical convolution-based models (such as ResNets [He et al., 2016]) the knowledge is constructed in a bottom-up fashion. This is a well-known fact. In particular, abstract concepts are always the result of the composition of simpler concepts. For example in early layers of CNNs for CV tasks, each neuron specializes in the detection of low-level features, while, as we move towards the head, the network learns patterns with more semantic relevance for us humans. This can be seen thanks to the beautiful visualization of [Olah et al., 2017] captured in Figure 2.1. This also reflects some neuroscientific discoveries where hierarchies of more and more abstract concepts have been demonstrated repeatedly, especially in the visual brain areas [Riesenhuber and Poggio, 1999].

While those resemblances are appealing to draw a connection between artificial and biological brains, the difference is still striking. For example, quite often Deep Learning models are static, that is, they are not altering their architecture over time but, in our biological brains, new connections can appear, while others can also cease to exist. This is also the so-called *neuroplasticity* of our brains, whose first scientific evidence has been reported by [Bennett et al., 1964]. As we will see, continual learning and few other fields (e.g., dynamic routing, conditional computation, etc.) are the only ones going in this direction.

On another note, **time** seems to be a major factor in both artificial and natural learning. Our current connectionist framework does not exploit the notion of time in learning. To accommodate such a factor we would need to redefine the current learning framework because so far the models process data but without being conditioned to *when* something is learned. There have been some attempts towards this

direction by defining the learning as a system of differential equations taking into consideration time as a fundamental variable and also some attempts to implement it by [Betti et al., 2020], although the majority of the works still operate in the classical scenario.

Another clear distinction between artificial and biological neurons lies in how they decide to fire. The artificial neuron receives inputs and multiplies them by some weights that are adapted during learning. To fire, it uses an activation function (such as ReLu [Agarap, 2018]), but the reality of biological neurons is different. Each biological neuron has its threshold resultant from a complex chemical interaction. A class of models that are trying to bridge this gap is Spiking Neural Networks [MAA, 1997] where the firing of the neuron is determined by a threshold on the signal received. Note that also this simplified model mimics neither the creation nor the destruction of connections (dendrites or axons) between neurons, and ignores signal timing. However, this restricted model alone is powerful enough to work with simple classification tasks.

Another important difference is that biological circuits contain a myriad of additional details and complexity not translated to DL models, including diverse neural cell types [Tasic et al., 2018] with some recent attempts by [Doty et al., 2021] to bridge this gap by changing the activation function for each artificial neuron. Another attempt to introduce more complex structures has been proposed by [Sabour et al., 2017] with the introduction of Capsule Net models, a family of networks where the neurons are structured in hierarchies.

The most widely known neuroscientific framework for the brain is the Complementary Learning Systems (CLS) [McClelland et al., 1995]. This framework explains why the brain requires two differentially specialized learning and memory systems, and it nicely specifies their central properties i.e., the *hippocampus* as a sparse, pattern-separated system for rapidly learning episodic memories, and the *neocortex* as a distributed, overlapping system that gradually integrates experienced episodes and extracts latent semantic structures. Instead, most of the proposed artificial models, are more of a well-engineered pipeline crafted to excel in a particular task such as Computer Vision, NLP, etc. and do not draw inspiration from such theories, although a very recent work proposed by [Arani et al., 2022] explored over this direction. With some recent developments in the CL field, rehearsal systems [Parisi et al., 2019] (systems that replay old data through a buffer) can be recast with such a point of view. In fact, we can think of the rehearsal buffer (or the part of the CL system dedicated to storing “old” patterns used in replay) as a long-term memory while the other part of the architecture is the fast-paced learner of the intelligent agent i.e. the hippocampus. Perhaps the key to continual learning will be in the



inspiration from neuroscientific models. Indeed recently [McCaffary, 2021] proposed a systematic review of the approaches in CL along with some insights into why we should pay more attention to neuroscientific theories.

As we saw, the gap between artificial and biological models is still relevant and the two fields, nowadays, show big differences in their understanding of intelligence. However, one striking fact is that the artificial community has achieved impressive results without *directly mimicking* the current neuroscientific theories, suggesting that, perhaps, several paradigms of intelligence exist.

## 2.2 What is Continual Learning?

*“Every machine is built to make decisions, if it does not have the faculty to learn, it will act always in conformity to a mechanical scheme. We don’t have to let the machine decide about our conduct if we first have not studied the laws that rule its behavior, and made sure that such behavior will be based on principles that we can accept!”*  
- Norbert Wiener

**Definition:** The aforementioned quote is taken from “Introduction to Cybernetics”, and highlights the fact that the fundamental ability to *continually learn* is a very important skill that any intelligent system should possess. Although we are now able to devise powerful artificial systems achieving superhuman performance in some tasks, we, as humans, still exhibit a core ability that would be fundamental to replicate intelligence as we know it. The ability to learn new concepts without erasing past knowledge. These two aspects are the main objectives of Continual Learning. First, exhibiting the ability to assimilate new concepts incrementally. Secondly, showing the capability of memorization i.e. not forgetting what has been previously learned. In a nutshell **Continual Learning studies how to develop systems that learn incrementally over time without forgetting previously acquired knowledge.**

**History:** Continual Learning has drawn a lot of interest from the research community only in the later years even though the question itself is very old. One of the early papers trying to tackle this phenomenon has been proposed by [Carpenter and Grossberg, 1988] where the authors proposed a short-term and long-term memory pattern detector through the Adaptive Resonance Theory. In fact, to the best of our knowledge, this seems to be the earliest work proposed. Later, as connectionist



Figure 2.2: Continual learning spectrum. The optimal algorithm should exhibit enough plasticity to learn new tasks while retaining enough stability to not forget the acquired knowledge.

models pave the way for modern Artificial Intelligence, other attempts and several proposals have been made. Later the work by [Ring et al., 1994] coined the term “Continual Learning”, here the system proposed, aimed to construct hierarchies of knowledge within a neural network. Later, with the works by [Thrun, 1995a] and [Thrun and Mitchell, 1995] Continual Learning started to get attention especially in both the Robotic and Reinforcement Learning research community.

**Terms:** When we say Continual Learning we have two other equivalent terms: *Incremental Learning* and *Lifelong Learning*. These terms can be used interchangeably and denote the same setting. There are no clear distinctions and probably the preference of one over another is just a matter of the research field we are in. For example, in the computer science field, it seems that continual learning and incremental learning are more common. Other terms are used but differ in the specific continual setting they study. For example: *Online Learning* and *Streaming Learning*. These are very similar, and there is no clear distinction yet. These terms are used to describe algorithms that learn by observing an example just one time and, sometimes, the latter can also refer to systems that can respond to queries in real-time. We will introduce a more formal definition in the next chapter.

**Subject of CL:** As we previously discussed, the study of Continual Learning is strictly tight with the widespread usage of connectionist models. In fact, before the advent of Artificial Neural Networks (ANNs), intelligence was modeled, usually, by a mixture of expert systems and clever algorithms. Posing the same “continual learning question” for these systems is still an interesting challenge, but the success of ANNs shifted the focus to connectionist models.

## 2.2.1 Stability-Plasticity Dilemma

Learning incrementally (or continually) with connectionist models requires one core ability, that is, **to adapt to a changing environment**. If the environment would not change over time, and we expose a system to operate on it, we would just need to understand, model, and hard-code the environment's rules to the system and we would achieve perfect functionality. Unfortunately, the real world does not seem to behave in such a predictable way. Instead, our reality constantly changes and we need to redefine our knowledge, reshape it in light of new facts, have room to constantly learn something new, and recombine previous knowledge to understand a novel concept. This is not *the only* necessary property for an intelligent system, the counterpart is also important. In fact, some things do not change in the world, old challenges might propose again, and, therefore, fundamental knowledge should not be forgotten. A truly intelligent system would **behave consistently on past lessons**. It would be able to detect and recognize past challenges, delivering correct solutions. The researchers gave a name to this trade-off and it is called the *stability-plasticity dilemma*. The long-term goal of Continual Learning is to create a system able to achieve a perfect balance between these two abilities as depicted in Figure 2.2. As we will see, it is termed a "dilemma" since achieving the optimal trade-off is a very hard task.

On top of these considerations dissecting new concepts and redefining them as a combination of old knowledge allows the forward transfer of intelligence. That is when we learn we sometimes can abstract the knowledge to solve a related problem. This is not uncommon it is the mechanism of *analogy thinking* where an "operational pattern" can be used to solve problems in apparently different domains. As an example, [Hill et al., 2019] investigates such property of intelligence in artificial networks. On the other hand, continual learning should give the ability to better grasp the past knowledge improving the ability to past challenges. This is even more common and we can think of this kind of ability as the "experience" that an agent accumulates in a certain field or in solving a certain category of tasks.

In a nutshell, the stability-plasticity dilemma can be considered the crux of intelligence. Showing adaptability to new environments while at the same time retaining knowledge of old environments seems to be the major qualities of an intelligent agent.

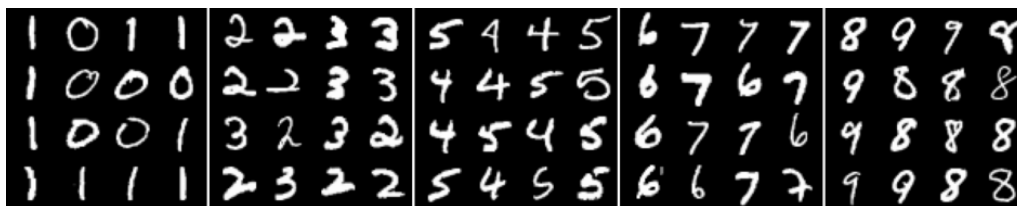


Figure 2.3: The original images for each task. This image shows the ground truth relative to Figure 2.5.

## 2.2.2 Catastrophic Forgetting











One core aspect of deep neural networks lies in the fact that if we do not introduce any kind of mechanism to achieve the balance between stability and plasticity, **the artificial network is naturally inclined to forget**. That is, the neural networks put much more emphasis on plasticity rather than stability. From a neuroscientific point of view, this fact does not make much sense unless we think about neural networks as systems without any form of memory. The reality is that networks do have memory, but by the nature of the learning algorithms we overwrite such memory. As the model incrementally learns, *each parameter in the network is modified by the updates of the backpropagation algorithm*. The optimal continual learning method would be able to modify the parameters without altering the performances of old tasks. This seems not to happen and therefore neural networks are prone to the so-called *catastrophic forgetting*, the phenomenon where old knowledge is corrupted.

### 2.2.3 A Visual Example

To better grasp the phenomenon of catastrophic forgetting, we will provide a visual example in the following section. As we discussed, catastrophic forgetting happens because the parameters tuned to solve a task (usually experienced before in time), are not suited for the currently experienced task. We hope to provide a clear visual example of the effects of catastrophic forgetting in a shallow architecture.

As the name suggests Deep Learning refers to architectures with many layers on top of each other. Because of this huge depth, computer vision (but not only this community) was able to achieve impressive results in the domain of pattern recognition. Unfortunately, we still do not *fully* control how the knowledge is built inside a deep neural network and if we want to counter forgetting we would need such information. To do so, we would need to *keep track of each parameter variation*

as we learn new concepts in a continuous fashion, but doing so, especially in such models, is hard if not an impossible job. Said that, on a small scale, we can still show what is going on inside a network. In the following toy example **we try to track forgetting of an autoencoder by dissecting the learning process per task**. We will use a simple one-layer autoencoder model and try to incrementally learn the famous MNIST [LeCun et al., 1998] dataset, still used in the continual literature to validate the proposed methods. We will divide the dataset into 5 tasks and learn to compress and reconstruct images. By doing so we will show the corruption of old images as we learn new tasks and connect them to the network's variation of the parameters.

The MNIST dataset is a grayscale dataset of  $28 \times 28$  images of handwritten digits going from the digit 0 to the digit 9, here some examples: . The MNIST was constructed from NIST's Special Database 3 and Special Database 1, the first has been collected among Census Bureau employees and the second one among high school students. It has a training set of 60,000 examples, and a test set of 10,000 examples. We will divide the dataset in 5 tasks, the first 1 is composed of the digits , task 2 by  , task 3 by  , task 4 by   and finally task 5 by  .

Although the best practice to work with image data is to use CNNs, we will limit our toy example to a naive autoencoder model of linear layers. This choice allows us to better unfold and analyze the variation of the parameters due to its simplicity. The model is composed of a single layer encoder  $\phi$  that encodes an image into a latent vector and a single layer decoder  $\psi$  that reconstructs the image. In particular, the single-layer encoder is a linear layer  $\phi : \mathbb{R}^{784} \rightarrow \mathbb{R}^{16}$  that will receive in input a flattened ( $28 \times 28 = 784$ ) representation of the image and compress into a latent vector of magnitude 16. The decoder, then take care of the reconstruction of the image by doing the reverse process, that is  $\psi : \mathbb{R}^{16} \rightarrow \mathbb{R}^{784}$  i.e. given a latent vector of size 16 it decompresses it to a flattened image.

More formally an autoencoder can be represented in the following way:

$$\hat{\mathbf{x}} = \psi(\phi(\mathbf{x}))$$

Where  $\mathbf{x} \in \mathbb{R}^{784}$  is the flattened representation of an original image coming from a task  $t$ ,  $\phi$  is the encoder network and  $\psi$  is the decoder network, and  $\hat{\mathbf{x}} \in \mathbb{R}^{784}$  is the flattened representation of the reconstructed image. The objective is to optimize and, in particular, minimize the mean square error (MSE) between the original image and the encoder's reconstruction. More formally we can define the objective function as:

$$\min_{\phi_{\theta}, \psi_{\theta}} \mathcal{L}(\mathbf{x}, \hat{\mathbf{x}}) = \min_{\phi_{\theta}, \psi_{\theta}} \|\mathbf{x} - \hat{\mathbf{x}}\|^2$$

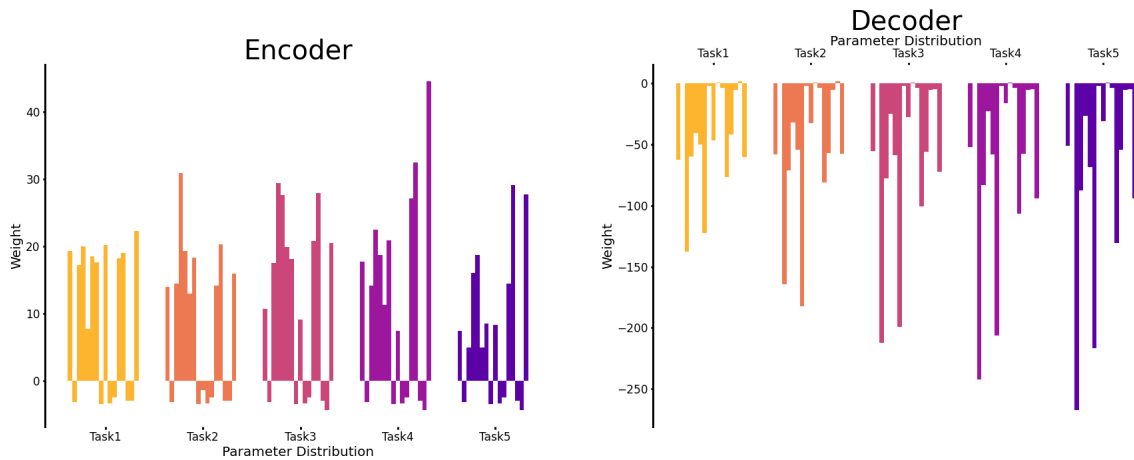


Figure 2.4: Variation of the parameters grouped by task. Each bar plot shows the distribution of the weights, we can see that each task modifies internal parameters. Each weight is computed as the sum of all the connections of the particular latent neuron.

Here  $\phi_{\Theta}$  represents the set of encoder’s parameters to be optimized while we use  $\psi_{\Theta}$  for the decoder.

By *incrementally learning* each task we want to show the **corruption in the ability of reconstruction of previous tasks**. The change in the parameters to accommodate the new task negatively impacts old tasks. In fact, if we try to retrieve old concepts we see catastrophic interference, that is, the network is confusing old concepts with newly learned ones. From now on let us refer to Figure 2.5, where is depicted the complete incremental learning and its effects. The grid reported encodes the performance of the autoencoder. Each row  $i$  refers to the model trained solely on data of task  $i$  but tested on all the other tasks. From the experiment, we can appreciate several effects. First, if we isolate the first column of the grid, we can visualize the performance of the original first task as time passes (we can think of it as the stability of the network as we will discuss in Section 4.2). Here, one can clearly see that feeding new concepts corrupts old ones. On the other hand, if we focus on the upper triangular section of the matrix, we see the ability of the model to generalize knowledge. This stresses the fact that generalization is a key component in continual learning. Intuitively more “general” models might experience less forgetting (further hints on this path can be found in Section 4.2 and Section 4.3). The connected change in the weights for each task is reported in Figure 2.4 (for both the encoder and decoder). As we can see, even a small change in the parameters dramatically impacts the stability plasticity trade-off. As reference in Figure 2.3 we report the ground truths.

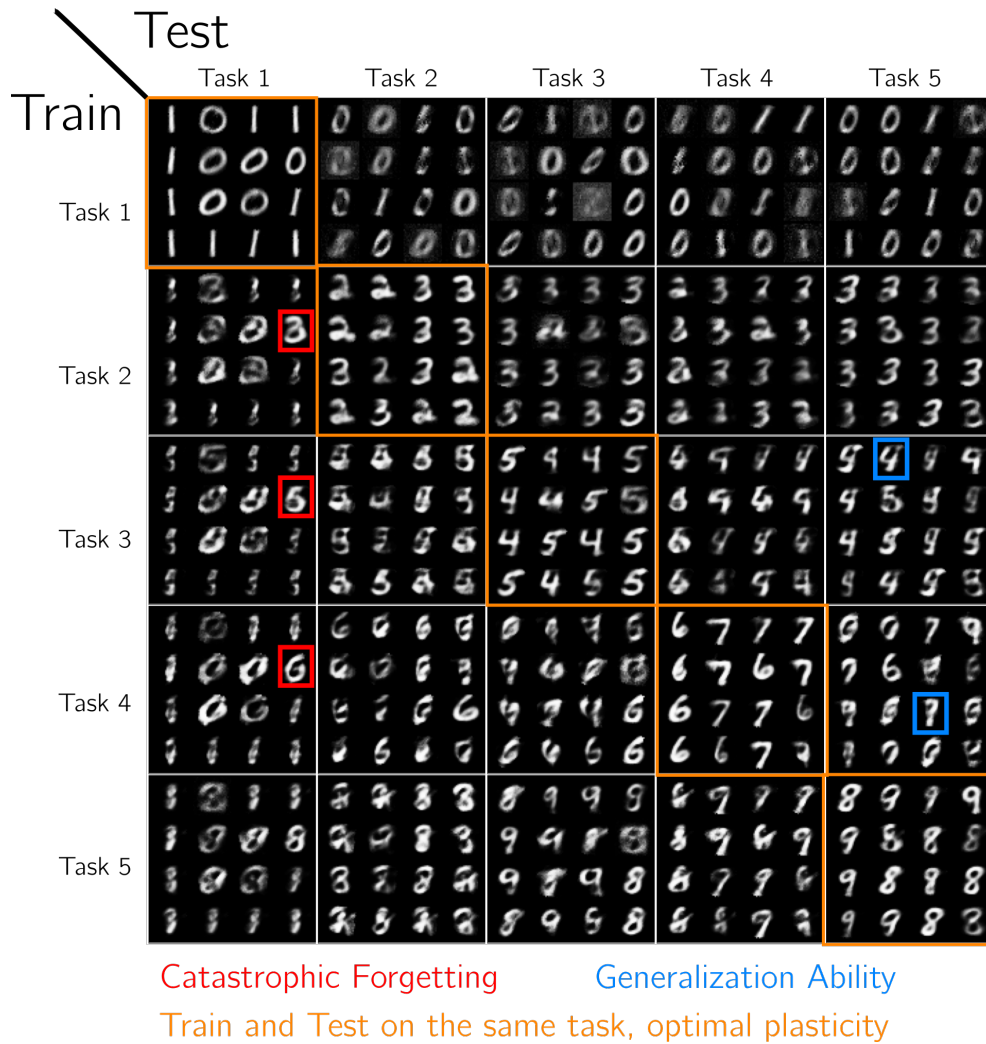


Figure 2.5: Results of the incremental training and test of the autoencoder model in the MNIST dataset were split into 5 tasks. Each row  $i$  of the grid, reports the performance of the model when trained on task  $i$  (or time  $t_i$ ) and tested on both old (left) and future (right) tasks. Training on previous tasks might unlock the intrinsic possibility to solve future tasks. This latest phenomenon is highlighted with the blue boxes. Ground truth in Figure 2.3.





## **Chapter 3**

# **Continual Learning Framework**

### 3.1 Definition and Settings

Being Continual Learning a relatively new discipline, *the community unfortunately still does not fully agree on a formal setting*. This is also corroborated by the fact that incremental learning is under the research light of several communities. Among the most active communities, we have NLP, Computer Vision, Reinforcement Learning, Neuroscience, and Robotics. Each of these communities has a well-established history and standard protocols, therefore, accommodating everyone in a common ground is still an ongoing process. However, in the following, we will introduce the most common definitions and settings shared in the Computer Vision literature.

There have been some attempts to formalize a setting for continual learning [van de Ven and Tolias, 2019, Lomonaco and Maltoni, 2017] through the definition of learning protocols and new terminologies. We will see these different learning paradigms in the following sections, but the core feature underlying learning incrementally is that the data experiences some distributional shift, that is, the **distribution of the data changes over time**. This is sufficient to abruptly cause forgetting in connectionist models, but we can define some settings which are more prone to cause such phenomenon, while others are more simple to overcome.

*The typical continual learning setting in computer vision is composed of a split dataset, where each (usually non-overlapping) split is considered an incremental task*. Therefore, each task contains data from several classes. Although this is not the only way to define a continual learning scenario, this is the most prominent one as pointed out in these surveys [Mai et al., 2022, Delange et al., 2021, Parisi et al., 2019]. Let us define a more formal definition:

**Formal Definition:** Given a dataset  $\mathcal{D}$  containing (in our case) images, we want to split  $\mathcal{D}$  in a sequence of  $n$  disjoint tasks that can be learned sequentially by our model:

$$\mathcal{T} = [t_1, t_2, \dots, t_n] \tag{3.1}$$

where each task  $t_i = (C^i, D^i)$  is represented by a set of classes  $C^t = \{c_1^t, c_2^t, \dots, c_{n^t}^t\}$  and training data  $D^t$ . We use  $N^t$  to represent the total number of classes in all tasks up to and including task  $t$ :  $N^t = \sum_{i=1}^t |C^i|$ . As a side note, usually in literature one would use the notation  $t$  to point at the current task (the task at time  $t$ ) and  $t - 1$  to point to the task before the current one.

A continual learning algorithm aims to model each task sequentially as time passes exposing the model at training time to each task in a sequential fashion. Operatively:

first, the algorithm is trained with mini-batches of patterns coming from task 1. Here we will record the system performance. Then, the model is exposed to task 2 data and the process continues until task  $n$ . One visual example can be seen in Figure 3.1, here the MNIST dataset is split into 5 tasks with 2 classes each <sup>1</sup>.

The previously defined learning scenario takes into consideration a distinct transition among tasks. In this particular case, we implicitly assume a *reset signal* between two tasks. When such signal is not present, and the transition between tasks is *smooth*, the complexity of the continual learning problem increases. If in this particular setting we query the system for real-time response, we are talking about streaming learning [Hayes et al., 2019]. This setting is more challenging because the models are allowed much less time to consolidate previously seen knowledge and therefore are more prone to experience catastrophic forgetting. Since this thesis focuses on computer vision problems, throughout the work we will stick to the introduced setting.

**Fine-Grained** So far we limited the notion of a task as a split of a dataset, but what happens if in a new task we experience new instances of previously seen classes? To this end, more complete settings for continual learning benchmarking have been proposed. One example is constituted by [Lomonaco and Maltoni, 2017]. Here the authors, along with a new dedicated dataset, introduce three different settings by mixing the experience of old and new data. Specifically, here we report the different scenarios:

- **New Instances (NI)**: new training patterns of the same classes become available in subsequent tasks. Here the model can experience new instances of old, previously seen, classes. With the possibility of seeing the same objects in new poses and conditions (illumination, background, occlusion, etc.). Here a good model is expected to incrementally consolidate its knowledge about the known classes without compromising what it has learned before.
- **New Classes (NC)** : new training patterns belonging to different, never seen, classes become available in subsequent tasks. This is the classic scenario (the one we formally introduced) and a model should be able to deal with the new classes without losing accuracy on the previous ones.
- **New Instances and Classes (NIC)**: new training patterns belonging both to known and new classes become available in subsequent training tasks. A good

---

<sup>1</sup>This particular setting takes the name of MNIST-split

model is expected to consolidate its knowledge about the known classes and learn the new ones. This is the most complete and difficult scenario since the addition of new classes poses the challenge of having good plasticity while the introduction of new old patterns asks for stability.

In our opinion, this categorization is preferable since it provides a more complete description of a continual learning benchmark. In fact, if we assume, as an example, that each task data is generated by an independent source, the task data will be continually augmented with new information. This scenario is captured by the NIC setting and cannot be handled by the standard definition. Unfortunately, due to the recent development of the field, we usually assume the NC scenario independently.

### 3.1.1 Online CL vs Offline CL

So far we introduced a basic notation, now we discuss *how a model can be trained* to face a continual learning stream of tasks and introduce the name of these scenarios. The continual learning literature distinguishes two options: online training and offline training.

**Online** In particular, in the online continual learning protocol, the algorithm is required to have a *single parameter update per pattern* (or one forward-pass). This is a very coercive setting and requires maximum performance in knowledge consolidation from the continual learner. In fact, this scenario is quite challenging because of the nature of Stochastic Gradient Descent i.e. the learning algorithm at the core connectionists models. Here the system might not have enough time to assimilate a concept, therefore weakening its understanding and subsequent stability.

**Offline** In the offline learning protocol, instead, we are *free to perform several parameter updates per pattern* i.e. we are allowed to see an image more than once. For an incremental learner, this setting is a double edge sword, in one case it favors the consolidation of the concepts since setting a large number of epochs guarantees the correct training of a model. On the other side, if we do not introduce any forgetting prevention mechanism, this corrupts the old informational content of the network i.e. the system is more exposed to catastrophic forgetting.

In the following paragraphs, we will introduce some of the settings that are now,

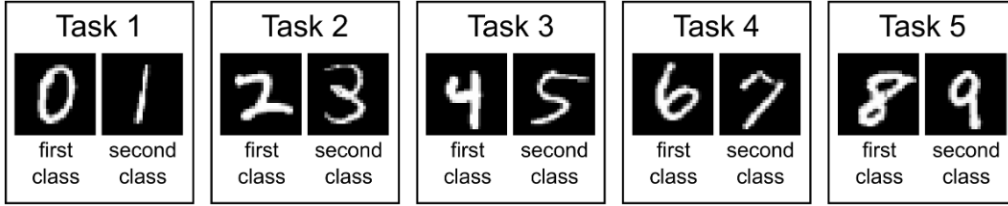


Figure 3.1: Schematic representation of split-MNIST task protocol. Taken from [van de Ven and Tolias, 2019]

de facto, shared among all the research communities researching continual learning.

### 3.1.2 Task-Incremental vs Class-Incremental

Assuming an NC-type of task flow, two sub-settings have been widely adopted by the research community and are well-defined. The Task Incremental (TI) setting and the Class Incremental (CI) setting.

**Task-Incremental** In the Task Incremental scenario, which is sometimes also referred to as *multi-head* scenario or *task aware (TAw)* the learning happens sequentially, but at test time, the learner *has also access to the task label*. This scenario is also known as multi-head because a typical learning system can potentially dedicate a particular subsystem per task, that can be specifically queried at test time through the task label knowledge. Typically the subsystem is a classifier head on top of a backbone.

More formally we consider task-incremental classification problems where at training time the learner has access to:

$$D^t = \{(\mathbf{x}_1, \mathbf{y}_1, \mathbf{z}_1), (\mathbf{x}_2, \mathbf{y}_2, \mathbf{z}_2), \dots, (\mathbf{x}_{m^t}, \mathbf{y}_{m^t}, \mathbf{z}_{m^t})\}$$

while at test time the learner has access to:

$$D^t = \{(\mathbf{x}_1, \mathbf{z}_1), (\mathbf{x}_2, \mathbf{z}_2), \dots, (\mathbf{x}_{m^t}, \mathbf{z}_{m^t})\}$$

where  $\mathbf{x}$  are input features for a training sample, and  $\mathbf{y} \in \{0, 1\}^{N^t}$  is a one-hot class ground truth label vector corresponding to  $\mathbf{x}_i$  while  $\mathbf{z} \in \{0, 1\}^{|\mathcal{T}|}$  is a one-hot task ground truth label vector. In a nutshell, during training for task  $t$ , the learner only has complete access to  $D^t$ , then we assume a reset signal among tasks i.e.

$C^i \cap C^j = \emptyset$  if  $i \neq j$ , and at test time the learner has access to patterns and their task label.

**Class-Incremental** Instead, in class incremental scenario, also known as *single-head* or *Task Agnostic (TAg)* the system has both access to task and class label during training time, but at test time it only has raw data. This constitutes a harder problem, but also a more realistic scenario.

More formally we consider class-incremental classification problems where at training time the learner has access to:

$$D^t = \{(\mathbf{x}_1, \mathbf{y}_1, \mathbf{z}_1), (\mathbf{x}_2, \mathbf{y}_2, \mathbf{z}_2), \dots, (\mathbf{x}_{m^t}, \mathbf{y}_{m^t}, \mathbf{z}_{m^t})\}$$

while at test time the learner has access only to:

$$D^t = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_{m^t}\}$$

where  $\mathbf{x}$  are input features for a training sample, and  $\mathbf{y} \in \{0, 1\}^{N^t}$  is a one-hot class ground truth label vector corresponding to  $\mathbf{x}_i$  while  $\mathbf{z} \in \{0, 1\}^{|\mathcal{T}|}$  is a one-hot task ground truth label vector, same as in TAW setting.

Although tag scenarios are more interesting from a pure machine learning perspective, the tag setting is more realistic. For example, let's draw an analogy: let us consider a baby as our incremental algorithm. We want to teach the baby to recognize elements coming from a particular environment, for example, kitchen accessories. Here the task label would be 'kitchen'. After the learning process has successfully terminated, whenever we ask the baby to recognize a fork, we do not need to provide a hint on the task (kitchen). In fact, the information of **where** he learned the concept should be irrelevant. This is also important because several objects can appear, and could be part of, several environments (tasks). For example, scissors can be found in the kitchen, but also in a studio. Therefore knowledge itself should be independent of the context where it is learned and, we think that class incremental setting provides a more useful challenge.

## 3.2 Baselines

In this chapter, we will see the principal naive approaches and introduce an overview of the state-of-the-art. In particular, we will introduce the cumulative and the finetuning

## Cumulative

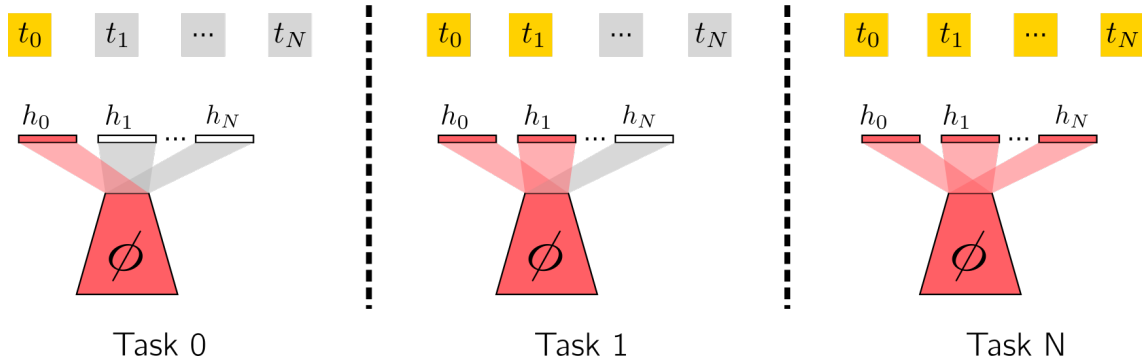


Figure 3.2: Depiction of the Cumulative/Joint approach for continual learning. The model is trained with all the data up to the current task  $t_i$ . The updates flow in the backbone and in all the heads up to  $h_i$ .

methods which constitute, respectively, the upper and the lower bound to evaluate continual learning strategies. Moreover, we consider our model to be composed of a backbone (or a feature extractor) and a dedicated classifier (head) for each task. We do so in light of the majority of the works in continual learning and computer vision, which are composed of this very structure.

### 3.2.1 Cumulative

To evaluate a continual learning algorithm we need an optimal method that acts as an **upper bound**. The cumulative strategy (also known as *joint-training*) constitutes the optimal continual learning strategy since **mimics a learner with perfect memory**. Indeed if we have perfect memory we can recall the past and not experience forgetting, to this end a recent work from [Knoblauch et al., 2020] proved theoretically that optimal continual learning has a perfect memory and is NP-hard.

To have optimal memory of the past, an algorithm should be able to save all the data that has been seen. This is a very inconvenient requirement and it must be avoided when considering the development of real lifelong learning systems. In fact, as the pace of real-world data generation is growing, such constraints would not be satisfied. Training from scratch with all the dataset data could be an upper bound approach, but it does not break down each incremental step upper bound. To this end, the cumulative strategy accumulates all the data seen up to a certain task and trains the network from scratch, therefore, providing an incremental upper bound.

## Finetuning

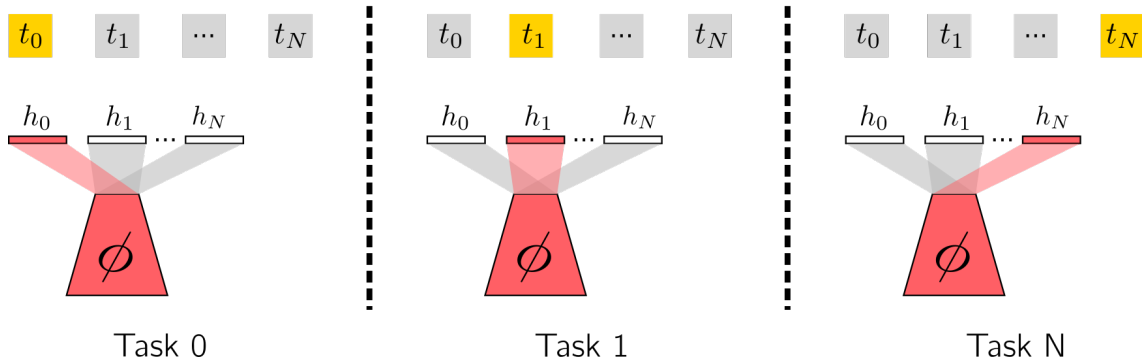


Figure 3.3: Depiction of the Finetuning approach for continual learning. The model is trained exclusively with the data coming from the current task  $t_i$ . The updates flow in the backbone and only in the  $h_i$  head.

More formally, for the cumulative approach, the data of task  $i$  is defined to be:

$$t_i = \bigcup_{j=0}^{j=i} t_j$$

when  $i = 1 \dots n$  to complete the incremental setting. At each time  $t_i$  the model is trained on the cumulative data and therefore we are able to define the upper bound performance for each task  $i$ . One observation is that the cumulative performance in the last task it is equivalent to the performance of the model trained with the whole data. In Figure 3.2 we depict a visual example of the cumulative approach. Here for each task, the backbone is always updated along with the heads of competence. However the updates of the heads can be also shared among all the tasks, that is, each task data alters all heads parameters. Of course, this design choice does not favor the prevention of forgetting, instead, it allows the disruption of consolidated knowledge and we won't consider this case<sup>2</sup>.

### 3.2.2 Finetuning

We previously saw the upper bound for CL, that is, the optimal continual learning approach for a benchmark. Now, we introduce the finetuning approach which constitutes the **lower bound** methodology. Although we can argue that a random classifier would be the true lower bound, in practice we consider finetuning in which it is absent of any forgetting prevention mechanisms. In fact, it is equal to the practice of

<sup>2</sup>this is valid for finetuning too



**transfer learning among subsequent tasks** and measures the base resilience of the model against incremental scenarios. We also can consider it as a baseline to assess the generalization capabilities of a model.

A depiction of the method is given in Figure 3.3. Here, the model is trained sequentially and each task head is updated with the data of its competence and, as in the cumulative approach, the backbone is always updated.

## 3.3 State-of-the-art

In the following sections, we will introduce the main categorizations of the approaches proposed by the community. In particular, we will explain the core mechanism and show the pros and cons of each category. Although there is no absolute preferred solution, some approaches are more explored than others and show more promising results.

### 3.3.1 Structural-based

Structural-based approaches, also known as *architectural approaches* or *parameter-isolation* methods, fight forgetting by altering the structural composition of the network itself. In particular, structural approaches instantiate dedicated modules as they experience new tasks. The first work falling in this category is perhaps Progressive Neural Networks (PNN) [Rusu et al., 2016] where the network is augmented with new connections spanning both height-wise and width-wise.

In the task-aware setting, this approach constitutes a convenient and naive solution to fight catastrophic forgetting. In fact, having the task label at test time allows us to correctly determine a dedicated subnetwork. Instead, in task agnostic setting, we would not be able to select such a submodule. We can see very few structural-based approaches tackling class incremental setting due to the aforementioned limitation [Lee et al., 2020, Rajasegaran et al., 2019]. That said, Structural approaches can be subdivided into Fixed Architecture (FA) and Dynamic Architecture (DA). FA only activates relevant parameters for each task without modifying the architecture [Mallya and Lazebnik, 2018, Kirkpatrick et al., 2017], while DA adds new parameters for new tasks while keeping old parameters unchanged [Yoon et al., 2018, Rusu et al., 2016]. Although architectural methods are very intuitive, they are

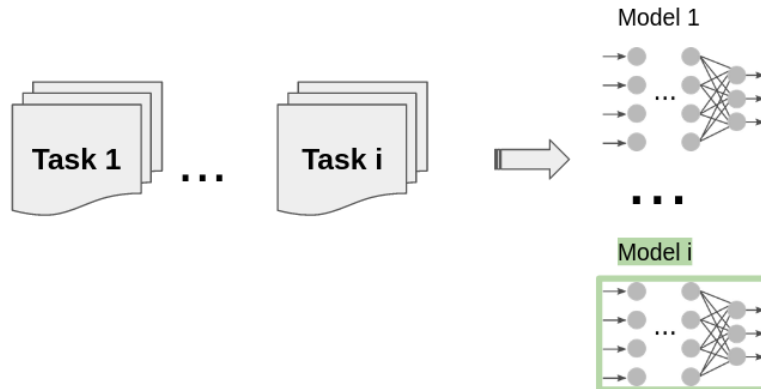


Figure 3.4: Architectural approaches for Continual Learning alter the structural properties of the network itself.

bulky. In fact, the major drawbacks are in the expansion of the parameters which can result in a memory-intensive method (DA), or in the architectural limitation of the number of parameters that can be saturated (FA).

### 3.3.2 Regularization-based

In parameter based approaches also known as *weight-regularization* or *data-regularization* approaches, forgetting is handled with procedures that regularize the parameter updates. Among the most famous ones, there are Elastic Weight Consolidation (EWC) [Kirkpatrick et al., 2017] and Synaptic Intelligence (SI) [Zenke et al., 2017]. EWC was the first regularization-based approach using second-order information. In particular, the procedure regularizes the updates through the Fisher information which is computed at each parameter update.

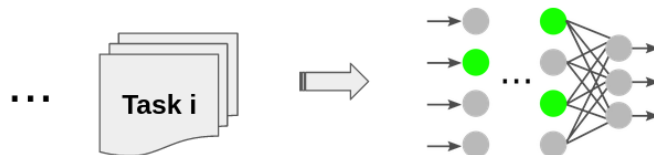


Figure 3.5: Regularization approaches for Continual Learning alter only the parameters properties of the network.

In this category, we can also find Learning without Forgetting (LwF) [Li and Hoiem, 2017], which is one of the most influential methods in continual learning literature. LwF uses Knowledge Distillation [Hinton et al., 2015] in the logits of the

network. The main strength of LwF lies in the fact that it does not use previously-stored examples while still being purely data-driven. In particular by storing the old model at time  $(t - 1)$  the method can distillate old knowledge by forwarding to the old model the current data. Since the introduction LwF, KD has been widely adopted by the continual learning community as part of new methodologies among the works we report [Douillard et al., 2020, Rebuffi et al., 2017, Buzzega et al., 2020, Pourkeshavarz and Sabokrou, 2022, Joseph et al., 2021, Wu et al., 2019, Banerjee et al., 2021, Javed and Shafait, 2018, Ahn et al., 2021, Dhar et al., 2019], but we are aware of many others that we do not report for brevity. The main strength of regularization-based approaches lies in their data/architecture constraint-free nature. In fact, they usually work with an underlying mathematical justification. This property surely allows a more principled continual learning strategy, but it can make the learning procedure cumbersome: computing second-order or estimating gradients directions, might slow down the learning while hindering it.

### 3.3.3 Rehearsal-based

In rehearsal-based approaches (or *data-replay* approaches) the main mechanism exploited to overcome forgetting, lies in the usage of a **replay buffer** for old exemplars. The methods falling under this category, dedicate a memory cache to store data examples encountered during the incremental training i.e. the system samples and stores images experienced in previous tasks. We can think of the buffer as long-term memory. In fact, what typically happens is that the memory is queried to augment the task at hand, that is, we retrieve and inject old examples to the current data batch. This mechanism prevents forgetting by allowing the network to directly recall past examples, a visual depiction can be seen in Figure 3.6.

Perhaps the most famous work among rehearsal-based approaches is Experience Replay (ER) [Rolnick et al., 2019] inspired by the Reinforcement Learning community its strategy is replaying data by randomly selecting old examples. In the evolution of ER, which is Maximally Interfered Retrieval (ER-MIR) [Aljundi et al., 2019a], proposed a controlled sampling of the replays. Specifically, they retrieve the samples which are most interfered with, i.e. whose prediction will be most negatively impacted by the foreseen parameters update. Another famous method is Gradient Episodic Memory (GEM) [Lopez-Paz and Ranzato, 2017] in which the authors devised a system where the gradient update of the replay examples should follow the original direction.

A closely related mechanism is **generative replay** (GEN) [Shin et al., 2017, van de



Figure 3.6: Rehearsal approaches for Continual Learning store old patterns to augment the data of the current task.

[Ven and Tolia, 2018, Wu et al., 2018]. In this approach, old data is recorded in a buffer and then compressed, after that, a generative model such as a GAN [Goodfellow et al., 2014], generates a *synthetic version* of the old distribution and augments the data of the current task. The main disadvantages of generative replay are that it takes a long time to train and it does not constitute a viable option for more complex datasets given the current state of deep generative models. Another approach devised by [Liu et al., 2020a] tries to overcome such limitations by generating intermediate features instead of the original data, trying to decrease the computational complexity of the generation procedure.

The pros of rehearsal-based approaches are their simplicity and effectiveness. In fact, the methods with best performances in continual learning exploit exemplars as shown in this challenge review [Lomonaco et al., 2022] where the best approaches used exemplars. The drawback of rehearsal continual learning is the usage of a memory buffer, which can be saturated as the number of tasks to be learned grows. To overcome such drawback some methods propose the usage of representative exemplars [Hayes et al., 2019] and herding [Liu et al., 2020b] techniques aimed to reduce the amount of memory required. Here, an interesting work (GDumb) proposed by [Prabhu et al., 2020] offers a simple baseline to rehearsal systems and questions the advancements of continual learning research itself due to its outstanding performance. Besides its performance, the system is very simple. In particular, the model samples data as experiences the stream of incoming task data. It does so until it fills a rehearsal buffer, by taking care to balance the proportion among classes. When the task data stream ends the dumb learner (a simple MLP or CNN) is trained only on the buffer data. GDumb achieves state-of-the-art performances.

# **Chapter 4**

## **Works**

## 4.1 Smaller is Better: An Analysis of Instance Quantity/Quality Trade-off in Rehearsal-based Continual Learning

We begin our dissection by focusing on rehearsal-based methods i.e., solutions in where the learner exploits memory to revisit past **data**. Due to its prominent performance and the abrupt usage, rehearsal systems are nowadays one of the preferred countermeasures to fight catastrophic forgetting.

So far, the focus from the community has been put into finding smart methodologies to improve the incremental performance. Instead, we ask ourselves what happens if we boost the capacity of the memory buffer. How much does impact altering the data storable in the memory? Indeed, in this study, we propose an analysis of the memory quantity/quality trade-off adopting various data reduction approaches to increase the number of instances storable in memory. By apply complex instance compression techniques to the original data, such as deep encoders, but also trivial approaches such as image resizing and linear dimensionality reduction, we offer a simple study on the trade-off.

Then we introduce the usage of Random Projections as compression scheme and offer a simple pipeline through Extreme Learning Machines to resource-constrained continual learning, an appealing scenario where computational and memory resources are limited.

Continual Learning (CL) is increasingly at the center of attention of the research community due to its promise of adapting to the dynamically changing environment resulting from the huge increase in size and heterogeneity of data available to learning systems. It has found applications in several domains. Its prime application, and still most active field, is computer vision, and in particular object detection [Gidaris and Komodakis, 2018, Thrun, 1995b, Parisi et al., 2019]; however it has since found applications in several other domains such as segmentation [Cermelli et al., 2020, Michieli and Zanuttigh, 2019, Yu et al., 2020a], where each segmented class has to be learned in an incremental fashion, as well as in other fields, among which we mention Reinforcement Learning (RL) [Xu and Zhu, 2018, Lomonaco et al., 2020] and Natural Language Processing (NLP) [Gupta et al., 2020, Sun et al., 2020, de Masson d'Autume et al., 2019].

Ideally, the behaviour of CL systems should resemble human intelligence in its ability to incrementally learn in a dynamical environment [Hadsell et al., 2020], with minimal waste of resources, spatial or computational. The main problem encountered by these systems resides in the famous stability-plasticity dilemma of neuroscience, resulting in the so called *catastrophic forgetting* [McCloskey and Cohen, 1989], a phenomenon where new information dislodges or corrupts previously learned knowledge, resulting in the deterioration of the ability to solve previously learned tasks.

Solutions to this problem typically incur in a increase in resource requirements [Lomonaco et al., 2022] both for CL's very nature (the more tasks arrive the more data the agent need to process), and for the nature of the systems that try to solve it, both in the increased complexity of the typically deep learning models, and in the time and space requirements of continuously learning multiple models. This problem become particularly evident in rehearsal-based methods.

Rehearsal-based methods, *i.e.*, approaches that leverage a memory buffer to cope with catastrophic forgetting, are emerging as the most effective methodology to tackle CL. Their performance, backed by extensive empirical evidence [Lomonaco et al., 2022], finds also a theoretical justification in Knoblauch and co-workers' finding that optimally solving CL would require perfect memory of the past [Knoblauch et al., 2020]. In fact, if we were able to completely re-train a new system with all previous data every time a new task arrives, Continual Learning would not appear to be any different from any other learning problem. However, this approach is both spatially and computationally infeasible for most real-world problems and we can argue it is precisely these memory and computational limitations that characterize CL and distinguish it from other learning problems.

Our investigation aims to analyze the trade-offs on limited-memory CL systems.

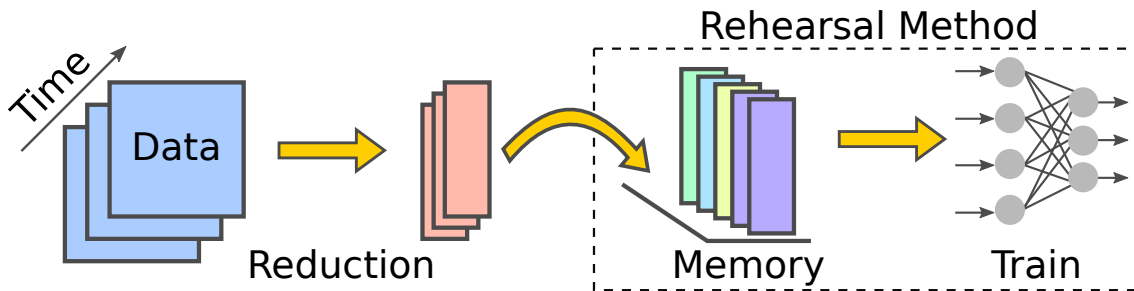


Figure 4.1: Our work analyzes the optimal instance quantity/quality trade-off in memory buffers of rehearsal-based Continual Learning systems. We carry out our analysis by applying several dimensionality reduction schemes to increase the quantity of storable data.

In particular, we focus on the quantity/quality trade-off for memory instances. We do so through the analysis of several dimensionality-reduction schemes applied to data instances that allows us to increase the number of examples storable in our fixed-capacity memory. In particular we adopted deep learning encoders such as a variation of ResNet18 [He et al., 2016] and Variational Autoencoders (VAE) [Kingma and Welling, 2014], the simple yet surprisingly effective extreme resizing of image data, and, lastly, we explored Random Projections for dimensionality reduction. The latter scheme turns out to be very effective in low memory scenarios also reducing the model’s parameter complexity. Indeed, we will show that a variation of Extreme Learning Machines (ELM) offers a simple yet effective solution for resources-constrained CL systems.

Our analysis will focus on computer vision tasks and use GDumb [Prabhu et al., 2020] as a rehearsal-baseline. GDumb is a model that has been proposed to question the community’s progress in CL thanks to the fact that in lieu of its outstanding simplicity, it was still able to provide state-of-the-art performance. Further, its simplicity also results in high versatility, as it proposes a general CL formulation comprising all task formulations in the literature. GDumb is fully rehearsal-based, and it is composed by a greedy sampler and a dumb learner, that is, the system does not introduce any particular strategy in the selection of replay data. Therefore, it represents the ideal candidate method to carry out our analysis.

The experimental findings highlighted in this study are multiple: first, we show that when the memory buffer is fixed and extreme values of resizing of instance data is applied, we can easily push the state-of-the-art of CL rehearsal systems by a minimum of +6% to a maximum of +67% in terms of final accuracy. This surprising result suggests that the optimal trade-off between data quantity and quality is severely skewed toward the former and that in general the informational content required to



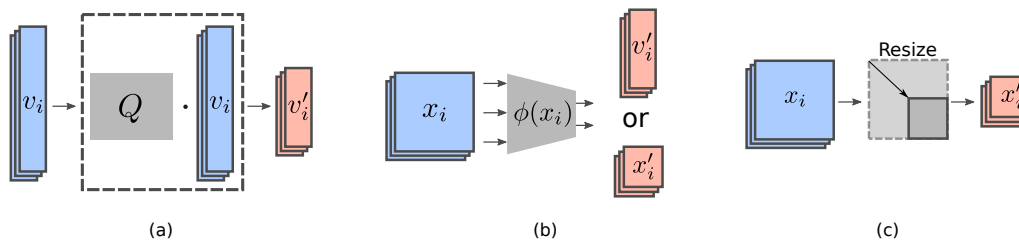


Figure 4.2: Depiction of the three main dimensionality reduction techniques analyzed. In (a) random projection (RP) each image is vectorized ( $v_i$ ) and then orthogonally-projected through a random matrix  $Q$  into  $v'_i$ . In (b), the encoder  $\phi$  outputs a latent vector  $v'_i$  (such as in VAEs) or a noise-free / shrunk image  $x'_i$  (as in CutR). In (c), we adopt a simple image resizing strategy through standard bilinear interpolation.

correctly classify images in standard datasets is relatively low. Then, we analyze the consumption of resources of rehearsal CL systems as we saturate the rehearsal buffer, and show that ELM offer a clear solution on CL systems constrained by very low resources environments.

## Related Works

Following some recent surveys [Parisi et al., 2019, Hadsell et al., 2020, Mundt et al., 2020], we divide CL approaches into three main categories: regularization-based approaches, data rehearsal-based approaches and architectural-based approaches. Although a few novel theoretical frameworks based on meta-learning have been introduced recently [Hadsell et al., 2020], the majority still fall within these categories (or in a mixture of them).

Regularization-based approaches address catastrophic forgetting by controlling each parameter’s importance through the subsequent tasks, by means of the addition of a finely-tuned regularizing loss criterion. Elastic Weight Consolidation (EWC) [Kirkpatrick et al., 2017] was the first well established approach of this class. It uses Fisher information to estimate each parameter’s importance while discouraging the update for parameters with greatest task specificity. Learn without Forgetting (LwF) [Li and Hoiem, 2017] exploits the concept of “knowledge distillation” to preserve and regularize the output for old tasks. More recently, Learning without Memorizing (LwM) [Dhar et al., 2019] adds in the loss an information preserving penalty exploiting attention maps, Continual Bayesian Neural Networks (UCB) [Ebrahimi et al., 2020] adapts the learning rate according to the uncertainty

defined in the probability distribution of the weights in the network, while Pomponi *et al.* [Pomponi *et al.*, 2020] propose a regularization of network's latent embeddings.

**Rehearsal-based** Rehearsal-based solutions allocate a memory buffer of a predefined size and devise some smart schemes to store previously used data to be replayed in the future, *i.e.*, to be added to future training samples. One of the first methodologies developed is Experience Replay (ER) [Rolnick *et al.*, 2019], which stores a small subset of previous samples and uses them to augment the incoming task-data. Aljundi *et al.* [Aljundi *et al.*, 2019a] propose an evolution of ER which takes in consideration Maximal Interfered Retrieval (ER-MIR). Their proposal lies between rehearsal and regularization methods, its strategy is to retrieve the samples that are most interfered, *i.e.* whose prediction will be most negatively impacted by the foreseen parameters update. Among other mixed approaches we have Rebuffi *et al.* [Rebuffi *et al.*, 2017] that proposes a method which simultaneously learns strong classifiers and data representation (iCaRL). Gradient Episodic Memory (GEM) [Lopez-Paz and Ranzato, 2017] and its improved version Averaged-GEM (AGEM) [Chaudhry *et al.*, 2019a] exploits the memory buffer to constrain the parameter updates and stores the previous samples as trained points in the parameter space, while Gradient based Sample Selection (GSS) [Aljundi *et al.*, 2019a] diversifies/prioritizes the gradient of the examples stored in the replay memory. Finally, a recent method proposed by Shim *et al.* [Shim *et al.*, 2021] scores memory data samples according to their ability to preserve latent decision boundaries (ASER).

**Architectural-based** Architectural methods alter their parameter space for each task. The most influential architectural-based approach is arguably Progressive Networks (PN) [Rusu *et al.*, 2016], where a dedicated network is instantiated for each task while Continual Learning with Adaptive Weights (CLAW) [Adel *et al.*, 2020] grows a network that adaptively identifies which parts to share between tasks in a data-driven approach. Note that, in general, the approaches that use incremental modules suffer the lack of task labels at test time, since there is no easy way to decide which module to adopt.

## Method

Before introducing the dimensionality reduction approaches adopted in our quantity/quality analysis we have to introduce the CL scenario considered and its task composition. Unfortunately the community has not yet converged to a unique

standard way to define a CL setting [van de Ven and Tolias, 2019]. Here we adopt GDumb’s formulation which is the most general one and specifically resembles Lomonaco and Maltoni’s formulation [Lomonaco and Maltoni, 2017]. In particular, we focus on the new class (NC)-type scenario [Lomonaco and Maltoni, 2017] where each task  $T_i$  introduces data instances of  $C_{T_i}$  new, previously unseen, classes. More formally a dataset benchmark  $\mathcal{D}$ , containing examples from  $C_{\mathcal{D}}$  classes, is divided into  $n$  tasks. Each task,  $T_i$  with  $i = 1 \dots n$ , carries a set of examples  $T_i = \{\mathbf{X}_{T_i}, \mathbf{Y}_{T_i}\}$  whose class is previously unseen i.e.  $\mathbf{Y}_{T_j} \cap \mathbf{Y}_{T_i} = \emptyset$  with  $j = 1 \dots i$  and  $\mathbf{Y}_{T_i} = \{c_1 \dots c_{T_i}\}$ . In other words, the model experiences a shift in the distribution of data as we train on each new task. We also consider the more realistic class incremental scenario (CI), that is, we are not allowed to know task labels at test time.

As incremental approach we use the recently proposed GDumb, which is composed of a simple learner and a greedy balancer. That is, given a fixed amount of memory  $\mathcal{M}$ , each instance of task data is randomly sampled in order to balance class instances in the memory, so that, at the end of the  $T_i$  task experience, the memory contains an equal number of instances of all previously encountered classes i.e. each class has  $\left\lfloor \frac{\mathcal{M}}{C_{\mathcal{D}} * i} \right\rfloor$  instances in memory.

Besides providing state-of-the-art performances, GDumb has been proposed as standard baseline to question our progresses in continual learning research, since after experiencing a task, the simple learner (such as a ResNet18 [He et al., 2016] or a MLP) is trained *only* with memory data, making GDumb a fully rehearsal based approach with random filtering of incoming data, and thus the ideal candidate to carry our study. In the following paragraphs, we briefly describe all the strategies adopted for dimensionality reduction.

## Random Projections (RP)

Extreme Learning Machines (ELM) [Huang et al., 2006] are a set of algorithms that exploit random projections as dimensionality reduction technique to preserve computational and spatial resources while learning. ELM have been introduced in 2006 and recently have found application in neuroscience [Qureshi et al., 2016, Lama et al., 2017] and in other problems such as in molecular biology [Chen et al., 2020]. The idea can be roughly described as a composition of two modules where the first one performs a random projection of the data, while the second one is a learning model. The appealing property of RP lies in the Johnson-Lindenstrauss lemma [Johnson, 1984] which states that given a set of points in a high dimensional plane, there is a

linear map to a subspace that roughly preserves the distances between data points by some approximation factor.

The Johnson-Lindenstrauss lemma guarantees that we can obtain a low-distortion to the dimensionality reduction by multiplying each instance vector by a semi-orthogonal random matrix  $Q^{m \times n}$  in the  $(m, n)$  Stiefel manifold. More formally, let  $x_i$  be an image of the current task of width, height and number of channels  $w$ ,  $h$ , and  $c$  respectively, then the size of  $x_i$  is  $n = hwc$ . We can consider its vectorization as  $v_i \in \mathbb{R}^n$  and its compressed representation

$$v'_i = Qv_i \quad \text{s.t.} \quad Q^T Q = I_m \quad (4.1)$$

with  $v'_i \in \mathbb{R}^m$ .

The usage of ELM unsuspectedly unlocks two main advantages: First it allows us to exploit the dimensionality reduction by *increasing the number data instances* storable in the memory buffer. Secondly and, more importantly, allows us to *use models with significantly fewer parameters*. On the other hand, the approach loses coordinate contiguity and, with that, shift co-variance, rendering convolutional approaches inapplicable.

After the random projection, data instances will be forwarded to the greedy sampler of GDumb to fill the memory  $\mathcal{M}$ . Then, we perform a rehearsal train with any MLP-like architecture, resulting in an order-of-magnitude reduction in the amount of parameters needed to process visual data allowing the usage of CL rehearsal based solutions in very low resource scenarios.

## Deep Encoders

Deep encoders are neural models  $\phi$  that take as input an image  $x_i$  and, depending from the structure of such model, can output either a latent vectorial representation  $v'_i$ , or a squared feature map which we consider as a noise-free shrunked image  $x'_i$ . Figure 4.2 (b) reports visually the two possible encoding scenarios. In this work, we adopt a Variational AutoEncoder (VAE) [Kingma and Welling, 2014] for the first case and a pretrained ResNet18 [He et al., 2016] cut up to a predefined block (CutR) as a prototype for the second.

Method	CIFAR10		
	Acc@600KiB	Acc@1.5MiB	Acc@3MiB
EWC [Kirkpatrick et al., 2017]	17.9 ± 0.3	17.9 ± 0.3	17.9 ± 0.3
GEM [Lopez-Paz and Ranzato, 2017]	16.8 ± 1.1	17.1 ± 1.0	17.5 ± 1.6
AGEM [Chaudhry et al., 2019a]	22.7 ± 1.8	22.7 ± 1.9	22.6 ± 0.7
iCARL [Rebuffi et al., 2017]	28.6 ± 1.2	33.7 ± 1.6	32.4 ± 2.1
ER [Rolnick et al., 2019]	27.5 ± 1.2	33.1 ± 1.7	41.3 ± 1.9
ER-MIR [Aljundi et al., 2019a]	29.8 ± 1.1	40.0 ± 1.1	47.6 ± 1.1
ER5 [Aljundi et al., 2019a]	-	-	42.4 ± 1.1
ER-MIR5 [Aljundi et al., 2019a]	-	-	49.3 ± 0.1
GSS [Aljundi et al., 2019c]	26.9 ± 1.2	30.7 ± 1.2	40.1 ± 1.4
ASER [Shim et al., 2021]	27.8 ± 1.0	36.2 ± 1.1	43.1 ± 1.2
ASER <sub>μ</sub> [Shim et al., 2021]	26.4 ± 1.5	36.3 ± 1.2	43.5 ± 1.4
GDumb [Prabhu et al., 2020]	35.0 ± 0.6	45.8 ± 0.9	61.3 ± 1.7
Resize (8 × 8)	55.5 ± 0.2	64.5 ± 0.2	73.1 ± 0.2
ELM (128)	43.0 ± 0.3	47.1 ± 0.2	50.0 ± 0.2
CutR (8 × 8)	54.4 ± 0.2	60.9 ± 0.2	71.6 ± 0.6

Table 4.1: CIFAR10 experiments (5 runs)

**VAE** Variational Autoencoders [Kingma and Welling, 2014] have been introduced as an efficient approximation of the posterior for arbitrary probabilistic models. A VAE is essentially an autoencoder that is trained with a reconstruction error between the input and decoded data, with a surplus loss that constitutes a variational objective term attempting to impose a normal latent space distribution. The variational loss is typically computed through a Kullback-Leibler divergence between the latent space distribution and the standard Gaussian, the total loss can be summarized as follows:

$$\mathcal{L} = \mathcal{L}_r(x_i, \hat{x}_i) + \mathcal{L}_{KL}(q(z_i|x_i), p(z_i)) \quad (4.2)$$

given an input data image  $x_i$ , the conditional distribution  $q(z_i|x_i)$  of the encoder, the standard Gaussian distribution  $p(z_i)$ , and the reconstructed data  $\hat{x}_i$ . We use the encoding part of a VAE pretrained on a dataset by feeding each incoming image and retrieving the vectorial output representation  $v'_i$ , then the data point is forwarded to GDumb's greedy sampler to feed  $\mathcal{M}$ .

**CutR** As our second encoding approach, we use a pretrained ResNet18 [He et al., 2016] cut up to a predefined block. ResNets models are Convolutional Neural Networks (CNNs) introducing skip connections between convolutional blocks to alleviate the so called vanishing gradient [Hochreiter, 1998] problem afflicting deep architectures. The idea behind it, is to use the cut ResNet18 as a *filtering module* that outputs a smaller feature map, giving us  $x'_i$ . In fact, we cut the network towards later blocks, since neurons in the last layers, encode more structured semantics with

Method	ImageNet100		CIFAR100	
	Acc@12MiB	Acc@24MiB	Acc@3MiB	Acc@6MiB
AGEM [Chaudhry et al., 2019a]	7.0 ± 0.4	7.1 ± 0.5	9.05 ± 0.4	9.3 ± 0.4
ER [Rolnick et al., 2019]	8.7 ± 0.4	11.8 ± 0.9	11.02 ± 0.4	14.6 ± 0.4
EWC [Kirkpatrick et al., 2017]	3.2 ± 0.3	3.1 ± 0.3	4.8 ± 0.2	4.8 ± 0.2
GSS [Aljundi et al., 2019c]	7.5 ± 0.5	10.7 ± 0.8	9.3 ± 0.2	10.9 ± 0.3
ER-MIR [Aljundi et al., 2019a]	8.1 ± 0.3	11.2 ± 0.7	11.2 ± 0.3	14.1 ± 0.2
ASER [Shim et al., 2021]	11.7 ± 0.7	14.4 ± 0.4	12.3 ± 0.4	14.7 ± 0.7
ASER <sub>μ</sub> [Shim et al., 2021]	12.2 ± 0.8	14.8 ± 1.1	14.0 ± 0.4	17.2 ± 0.5
GDumb [Prabhu et al., 2020]	13.0 ± 0.3	21.6 ± 0.3	17.1 ± 0.2	25.7 ± 0.7
Resize (8 × 8)	33.6 ± 0.2	33.6 ± 0.3	38.5 ± 0.4	45.1 ± 0.2
ELM (128)	13.3 ± 0.2	15.4 ± 0.4	22.4 ± 0.3	25.7 ± 0.3
CutR (8 × 8)	36.25 ± 0.4*	36.27 ± 0.5*	32.6 ± 0.6	37.1 ± 0.2

Table 4.2: ImageNet and CIFAR100 experiments (5 runs)

respect to the early ones [Olah et al., 2017]. Therefore, we are able to extract semantic knowledge from unseen images leveraging transfer learning [Tan et al., 2018], that is, we exploit the ability of a model to generalize over unseed data. We refer to this method with the name CutR(esnet18). We use CutR instance encoding by feeding each image belonging to the current task and retrieving the shrunked output  $x'_i$  which is then forwarded to the greedy sampler module of GDumb to fill the memory  $\mathcal{M}$ .

In our analysis, we adopted the less resource-hungry VAE scheme for datasets where shift co-variance is not as important, such as the MNIST, in which the digits are centered in the image and thus most approaches at the state-of-the-art use a MLP as classifier. In all other instances, we used the CutR scheme.

## Resizing

We used also the simplest instance reduction approach one can think of *i.e.*, resizing the images to very low resolution through standard bilinear interpolation. The resized images are then fed to the sampler of GDumb to balance the classes in  $\mathcal{M}$  and all training and prediction is performed on the lowered resolution images.

Independently of the approach adopted, all data instances are reduced before storing them in memory  $\mathcal{M}$ , then we use GDumb’s greedy sampler to select and balance class instances, and finally, we use a suitable learner to fit memory data and assess the performance. In general, following GDumb, we adopt ResNet18 for large-scale image classification tasks for all approaches that maintain shift co-variance, reverting to a simple MLP for approaches without shift co-variance like RP.

## Experiments

We performed our analysis on the following standard benchmarks:

- MNIST [LeCun et al., 1998]: the dataset is composed by 70000  $28 \times 28$  grayscale images of handwritten digits divided into 60000 training and 10000 test images belonging to 10 classes.
- CIFAR10 [Krizhevsky, 2009]: consists of 60000 RGB images of objects and animals. The size of each image is  $32 \times 32$  divided in 10 classes, with 6000 images per class. The dataset is split into 50000 training images and 10000 test images.
- CIFAR100 [Krizhevsky, 2009]: is composed by 60000,  $32 \times 32$  RGB images subdivided in 100 classes with 600 images each. The dataset is split into 60000 training images and 10000 test images.
- ImageNet100 [Deng et al., 2009]: the dataset is composed of  $64 \times 64$  RGB images divided in 100 classes; it is composed of 60000 images split into 50000 training and 10000 test.
- Core50 [Lomonaco and Maltoni, 2017]: the dataset is composed of  $128 \times 128$  RGB images of domestic objects divided in 50 classes. The set consists of 164866 images split into 115366 training and 49500 test.

Following [Prabhu et al., 2020], we use final accuracy as the evaluation metric throughout the work. The metric is computed *at the end of all tasks* against a test set of never seen before images composed of an equal number of instances per class. This allows us to directly compare against the largest number of competitors in the literature.

All the experiments has been conducted with an Intel i7-4790K CPU with 32GB RAM and a 4GB GeForce GTX 980 machine running PyTorch 1.8.1+cu102.

### Parameter Sensitivity

In the first experiment, we compared different dimensionality reduction strategies as we altered the parameters. The analysis was conducted on three different datasets: MNIST, CIFAR10 and ImageNet100. In this evaluation we fixed the amount of

memory buffer used for GDumb during rehearsal training, and we measured the final accuracy as the parameters varied for each dimensionality reduction method. In particular we subdivided both MNIST and CIFAR10 datasets into 5 tasks of 2 classes each, with 600 KiB dedicated memory buffer, while ImageNet100 was divided into 10 tasks of 10 classes each, with 12 MiB memory buffer.

Figure 4.3 plots the performance of the various schemes as we reduce the dimensionality of the instances and thus increase their number in the allocated memory. The orange line represents the performance of the resize scheme. For the MNIST dataset, we considered eight different target sizes<sup>1</sup>  $x'_i \in \{27 \times 27, 24 \times 24, 20 \times 20, 16 \times 16, 12 \times 12, 8 \times 8, 4 \times 4, 2 \times 2, 1 \times 1\}$ . We performed the same resizing for CIFAR10 data. We did not report CIFAR100 analysis since the data format is the same as CIFAR10 and the result would be analogous. For ImageNet100, we resized each instance to  $x'_i \in \{32 \times 32, 24 \times 24, 16 \times 16, 6 \times 6, 4 \times 4, 2 \times 2\}$ .

The green line of Figure 4.3 represents the deep encoders. In particular, for MNIST we used a VAE [Kingma and Welling, 2014] pretrained on KMNIST [Clanuwat et al., 2018] and analyzed the performance of GDumb with compressed instances as we altered the size of the latent embedding vector to  $v'_i \in \{128, 64, 32, 16\}$ . On the other hand, for the CIFAR10 and ImageNet100 dataset we considered different parameters for CutR. In particular, we cut the ResNet18 up to the sixth layer to get a  $4 \times 4$  output, to the fifth to have a  $8 \times 8$  encoding, and lastly up to the third block to get a  $16 \times 16$  feature map.

The CutR Resnet18 has been pretrained on the complete ImageNet, thus the results in the ImageNet100 benchmark can be biased. We denote these biased results with CutR\*.

Lastly, the blue line of Figure 4.3 reports the accuracy of Random Projection followed by an MLP classifier. We recall that this kind of architecture is a variation of an Extreme Learning Machine (ELM), therefore we will refer to it with the term ELM. We analyzed the final accuracy as the size of the random projection changes, in particular the embedding sizes considered are  $v'_i \in \{512, 256, 128, 64, 32, 16\}$  for all the datasets.

For all the experiments in MNIST data, we used a 2-layer MLP with 400 hidden nodes as learning module, while we used a Resnet18 [He et al., 2016] for all the other analysis with exception of ELM scheme that maintains the 2-layer MLP model throughout. We did not perform any hyperparameter tuning on the learning module

---

<sup>1</sup>throughout the work we omit to write the channel component for brevity



MNIST	
Method	Acc@382KiB
GEN [Hsu et al., 2018]	75.5 ± 1.3
GEN-MIR [Aljundi et al., 2019a]	81.6 ± 0.9
ER [Rolnick et al., 2019]	82.1 ± 1.5
GEM [Lopez-Paz and Ranzato, 2017]	86.3 ± 1.4
ER-MIR [Aljundi et al., 2019a]	87.6 ± 0.7
GDumb [Prabhu et al., 2020]	91.9 ± 0.5
Resize (8 × 8)	97.2 ± 0.1
ELM (128)	95.0 ± 0.4
VAE (32)	94.6 ± 0.1

Table 4.3: MNIST final accuracy (5 runs) analysis as we vary the memory for all schemes considered.

in accordance with the GDumb [Prabhu et al., 2020] experimental protocol. For completeness we report the learning parameters: the system uses an SGD optimizer, a fixed batch size of 16, learning rates [0.05, 0.0005], an SGDR [Loshchilov and Hutter, 2017] schedule with  $T_0 = 1$ ,  $T_{mult} = 2$  and warm start of 1 epoch. Early stopping with patience of 1 cycle of SGDR, along with standard data augmentation is used (normalization of data). GDumb uses cutmix [Yun et al., 2019] with  $p = 0.5$  and  $\alpha = 1.0$  for regularization on all datasets except MNIST.

As we can also see from Figure 4.3 all the strategies considered unlock performance greatly above GDumb, thus suggesting that the quantity/quality trade-off is severely skewed toward quantity since each dimensionality reduction technique greatly improves the amount of data instances that can be stored in the memory buffer. It is also evident that the simple resizing strategy gives the best performance improving GDumb by +6% on MNIST and roughly by +20% on both CIFAR10 and ImageNet100 datasets.

Moreover, we chose to consider extreme levels of encoding. We did so to find the level of compression that irreversibly corrupts spatial information and thus makes learning impossible. Surprisingly, it turns out that a  $2 \times 2$  resizing still works on CIFAR10 data with performances above GDumb while a  $1 \times 1$  resize is still better than a random classifier whose performance would be 20% of final accuracy. This is a strong evidence that the amount of data storable in the memory buffer plays a central role, but also that CIFAR10 dataset constitutes an unrealistic benchmark and should not be considered to assess novel methodologies in the future.

After choosing and fixing the optimal parameters for each compression scheme, we study the performance of the rehearsal system as we alter the quantity of the memory allocated. In Tables 4.3, 4.2 we compute the final accuracy for all the

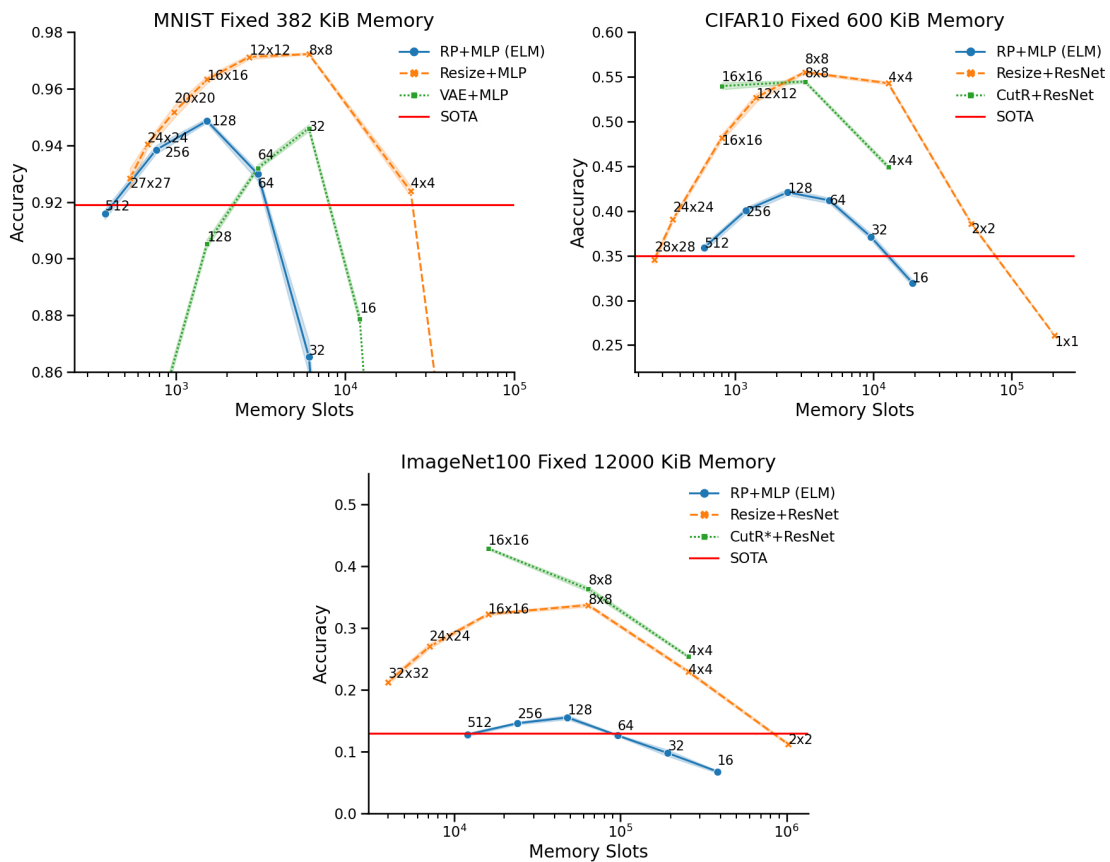


Figure 4.3: At top-left the accuracy analysis of the MNIST dataset. In top-right we have the analysis of CIFAR10 and at bottom we have ImageNet100. The state-of-the-art (SOTA) method is plain GDumb with an MLP as incremental learner in the MNIST experiment and Resnet18 in the others. The number of instances in memory (*i.e.* the  $x$  axis) is in  $\log$  scale. We report the results of (5 runs).

datasets previously considered, with the addition of CIFAR100 with an increase of 20% in performance. The amount of dedicated memory for the rehearsal buffer, has been chosen in order to be consistent with several other methods at GDumb, allowing us to compare GDumb's performance on optimized memory schemes against other methods. As we can see, all memory optimizations still provide huge advantages as the memory buffer varies, suggesting again, that instance quantity plays a fundamental role in rehearsal systems even with extreme encoding settings.

Finally, we note that the deep models used for classification have a large number of degrees of freedom and require a large amount of instances to be properly trained to capture the complexity of the task at hand. Simpler, lower dimensionality instances allow both for more instances and simpler classifiers with fewer parameters without

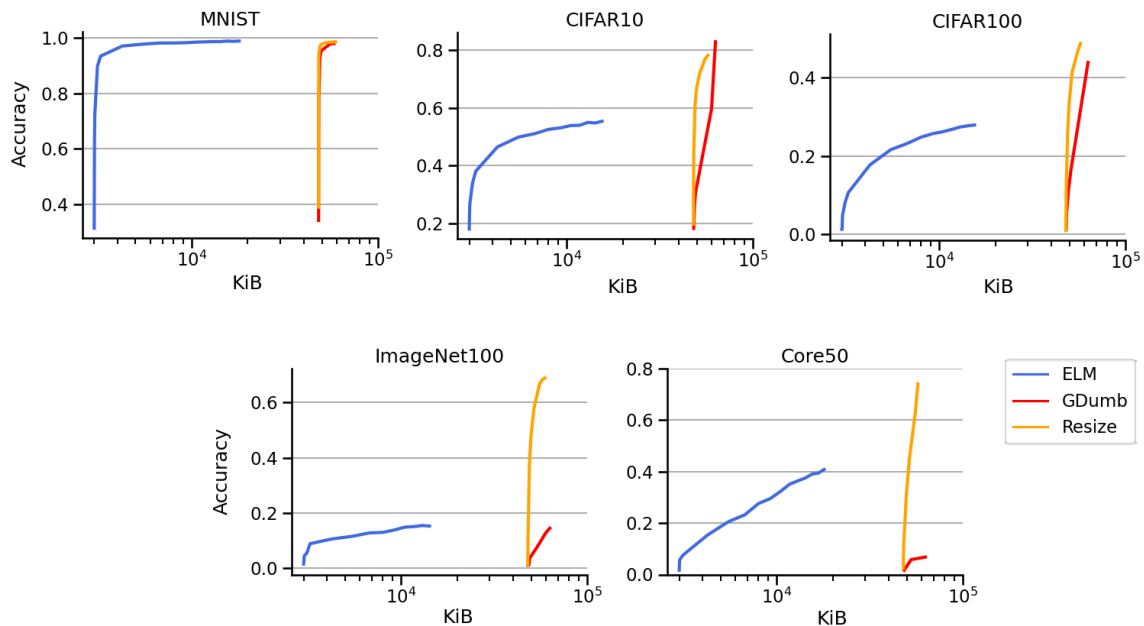


Figure 4.4: We show the total amount of KiB used by the whole CL system. We measure the consumption as we saturate the rehearsal memory plus the storage of model parameters. The  $x$ -axis is in  $\log$  scale.

losing lot of informational content.

## Resource Consumption

With the second experiment, we wanted to analyze the performance versus the total memory requirement for each approach. Here, we increased the number of instances in the memory buffer and added to the total consumption the working memory used by the classifier to store (and train) the parameters.

We considered three different scenarios: first we used the plain GDumb CL system without dimensionality reduction (representing GDumb), then we used ELM (with fixed embedding size of ( $v'_i = 128$ )), and lastly the resizing scheme (images resized to  $x'_i = 8 \times 8$ ). We selected the best parameters resulting from the previous experiment.

We then assessed the performance and resource usage using a new dataset, namely the Core50 [Lomonaco and Maltoni, 2017]. The reason behind the use of Core50 to validate our findings is twofold: first, we test again whether the quantity of extremely encoded data plays a central role on our rehearsal scheme. Secondly, we measure the performance and the resource usage of a CL system on a more complex

set of tasks. We divided the dataset into 10 tasks of 5 classes each.

In Figure 4.4, we report the results of this experiment. We can see that extreme levels of resizing still provide optimal results in all the datasets considered. One striking finding is that in Core50 with extreme resizing, even if the size was not optimized for the dataset, the final accuracy is increased by +67% with respect to GDumb . Second, we note that ELM constitute a viable solution in low resources scenarios. Indeed, we can surpass the performance of GDumb for low memory scenarios where even just the classifier used in other approaches could not fit in the allocated memory, much less the rehearsal buffer. This is clearly observed from the Core50 results. We can appreciate that by randomly projecting image data and learning in a low resource scenario provides a boost of +34% in the final accuracy.

Finally, it is worth noting there is a striking dissonance in the literature of rehearsal-based method when the narrative around buffer-memory sizes revolves around decisions among sizes of the order of 300KiB to 600KiB when then the same systems adopt complex classifiers using several megabytes of memory just for the learned parameters and in the order of gigabytes of working memory for learning. In a real constrained-memory scenario a simpler classifier with more instances offers a clear advantage.

## Conclusion

In this study, we analyzed the quantity/quality trade-off in rehearsal-based Continual Learning systems adopting several dimensionality reduction schemes to increase the number of instances in memory at the cost of possible loss in information. In particular, we used deep encoders, random projections, and a simple resizing scheme. What we found is that even simple, but extremely compressed encodings of instance data provide a notable boost in performance with respect to the state of the art, suggesting that in order to cope with catastrophic forgetting, the optimization of the memory buffer can play a central role. Notably, the performance boost of extreme instance compression suggests that the quality/quantity trade-off is severely biased toward data quantity over data quality. We suspect that some fault might be in the overly simplistic datasets adopted by the community, but mostly the deep models used for classification are well known to be data-hungry and the instances stored are not sufficient to properly train them, but can suffice for simpler classifiers with fewer parameters working on simplified instances.

It is worth noting there is a striking dissonance in the literature of rehearsal-based

method. The narrative on buffer-memory sizes revolves around decisions among sizes of the order of 300KiB to 600KiB when then the same systems adopt complex classifiers using several megabytes of memory just for the learned parameters and in the order of gigabytes of working memory for training. In a real constrained-memory scenario, a simpler classifier with more instances offers a clear advantage.

In fact, in a real low-resources scenario deep convolutional systems using several megabytes of memory for the model parameters and gigabytes of working memory for learning are not a viable solution. In this case, a variation of Extreme Learning Machines offer a simple and effective solution.

## Other Experiments

### Fixed Data Instances

With this experiment we aim to better show that instance quantity is preferable over instance quality. We fixed the number of data slots in the memory buffer, and we analyzed the performance as we alter the encoding size. In particular, we tested two datasets namely CIFAR10 and Core50. In CIFAR10 we fixed the buffer to 1000 data slots, while in the latter benchmark we fixed it to be 8000 slots. What we can see from Figure 4.5 is that the improvement of performance is not given by the encoding's smoothing property, and, again, we confirm that rehearsal systems are skewed towards data quantity.

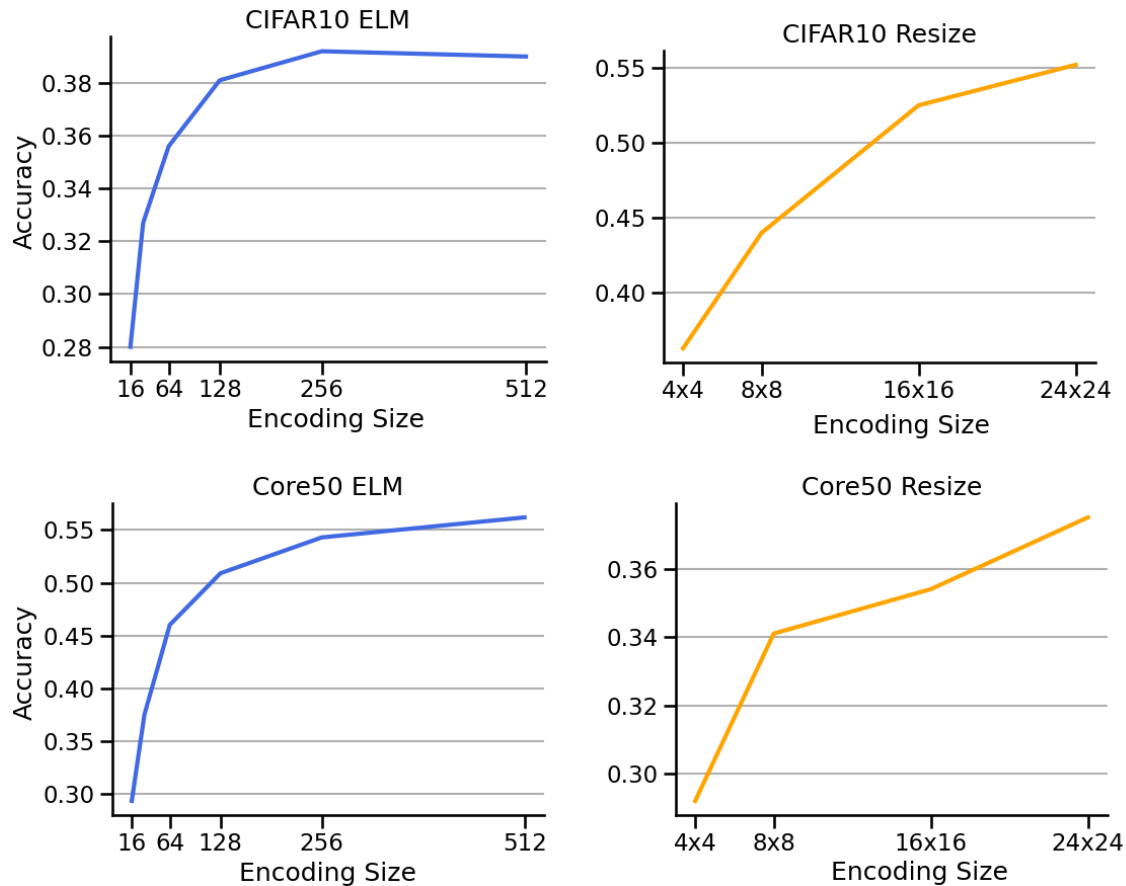


Figure 4.5: Performance as we vary the parameters for each scheme on CIFAR10 and Core50. In the former benchmark, the memory buffer is of 1000 fixed instances, while in the latter is of 8000.

## ELM Width Analysis

As we specified in the work, we used a variation of an Extreme Learning Machine. In particular, the architecture is composed by a random projection module and a learning module. The first is implemented through an orthogonal random matrix. While the second is a two layer MLP. Throughout the study we used 400 hidden units as last layer before the output. We choose to do so to be consistent with GDumb experimental settings. With this experiment we analyze the accuracy metric as we change the number of hidden units. We fixed the encoded size of data to be  $v_i/128$ . As memory buffer, we used a different number of data slots for different datasets. That is, for MNIST and CIFAR10 we adopted 2400 slots (600 KiB), in ImageNet100 we used 48000 instances i.e. 12 MiB, while for Core50 we used 8000 slots (2 MiB). In Figure 4.6 we can see that 100 hidden units are sufficient to achieve

the maximum performance. This, again, shows that more deep classifiers which are common in CL rehearsal literature, might need more data to be trained properly.

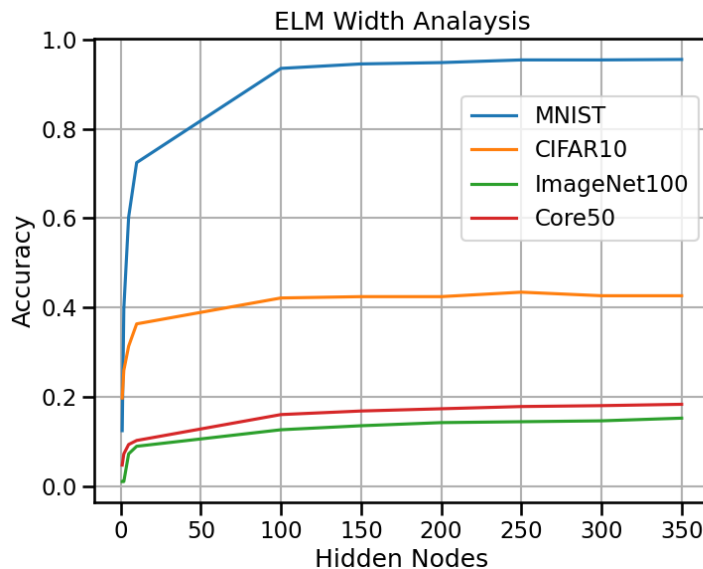


Figure 4.6: Analysis of final accuracy as we alter the number of hidden units in ELM.

## Experiments with other Rehearsal Systems

Throughout our study, we used GDumb to carry out our analysis. Although we extensively motivated this choice, we also tested two different rehearsal systems. In particular we studied ER Rolnick et al. [2019] and ER-MIR Aljundi et al. [2019a] performance as we adapt them to work in a low resource scenario. We simply substitute the original learner with our ELM proposal. In Table 4.4 we report the performance of CIFAR10 with 600 KiB buffer memory and  $v'_i = 128$  encoding. As validation metrics we used the final accuracy and the average forgetting Chaudhry et al. [2018] (lower is better). In order to train the systems, we used the official implementations found at [https://github.com/optimass/Maximally\\_Interfered\\_Retrieval](https://github.com/optimass/Maximally_Interfered_Retrieval) without any alteration of training hyperparameters. As we can see, the results suggest again that ELMs constitute a valid solution for low resource CL systems and that rehearsal solutions are biased toward data quantity over data quality.

<i>Method</i>	CIFAR10 Fixed Memory 600 KiB			
	<i>Accuracy (A)</i>	<i>Forgetting (F)</i>	<i>ELM (A)</i>	<i>ELM (F)</i>
ER Rolnick et al. [2019]	27.5 ± 1.20	48.0 ± 0.40	42.0 ± 0.10	41.2 ± 0.16
ER-MIR Aljundi et al. [2019a]	29.8 ± 1.10	44.6 ± 0.48	45.6 ± 0.10	31.6 ± 0.01

Table 4.4: Experiments in CIFAR10 with two different rehearsal systems in low resource scenario.

## Other Specifications

### Resource Consumption

In Table 4.5 we report some summary statistics. In particular, we report GDumb’s performance improvements for two encoding schemes i.e. Resize ( $8 \times 8$ ) and ELM ( $v_i' = 128$ ). We reported only the accuracy according to optimal parameters. We also added the compression factor  $\mathcal{C}$ , the requirements to store model’s parameters  $\Theta$  and the memory buffer  $\mathcal{M}$ . We also report the quantity of GPU memory usage to train GDumb for each encoding scheme. We can see that there is a big gap on the training requirements and memory buffers.

	MNIST	CIFAR10	CIFAR100	ImageNet100	Core50	Compression	Params + $\mathcal{M}$	GPU Training
Resize ( $8 \times 8$ )	(+6%)	(+21%)	(+20%)	(+20%)	(+67%)	253:1	60 MiB	2.2 GiB
ELM (128)	(+10%)	(+10%)	(+10%)	(+10%)	(+10%)	192:1	16 MiB	0.72 GiB

Table 4.5: Performance summary and memory compression

### Datasets Specification

For completeness, we report in Table 4.6 some specifications for the considered datasets. In particular, we provide the task subdivision for each dataset. As we can see MNIST and CIFAR10 have been split in 5 tasks of 2 classes each. This splitting is also known in literature as Split-CIFAR10 and Split-MNIST. For CIFAR10 and ImageNet100 benchmarks we used 10 tasks of 10 classes each, meanwhile for Core50 we shuffled all scenarios and created 10 tasks of 5 classes each. The majority of the works fix the memory slots to define the memory buffer. In our case we used memory requirements expressed in KiB or MiB so that we could alter each slot consumption.



We provide a correspondence between memory requirements and memory slots in the case we consider original image sizes, we do so to ease future comparisons against our work.

<i>Dataset</i>	<i>Image size</i>	<i>Experimental Settings</i>		
		<i>Memory Size</i>	<i># Instances</i>	<i>Task Composition</i>
MNIST	28x28x1	382 KiB	500	5 tasks, 2 classes
CIFAR10	32x32x3	600 KiB	200	5 tasks, 2 classes
		1.5 MiB	500	
		3 MiB	1000	
		6 MiB	2000	
CIFAR100	-	-	-	10 tasks, 10 classes
ImageNet100	64x64x3	12 MiB	1000	10 tasks, 10 classes
		24 MiB	2000	
Core50	128x128x3	15 MiB	312	10 tasks, 5 classes

Table 4.6: Dataset and memory statistics, in CIFAR100 row we omit the 2nd, 3rd and 4th columns since are equal to CIFAR10 row.

## 4.2 Towards Exemplar-Free Continual Learning in Vision Transformers: an Account of Attention, Functional and Weight Regularization

While in the previous work we considered old data points as pivotal instrument to investigate catastrophic forgetting, now we focus on the **structural** properties of the model considered. In particular, we ask ourselves how some parts of a network, when properly regularized, impact to the overall performance of an incremental scenario. We decided to investigate the continual learning of Vision Transformers (ViT) for the challenging exemplar-free scenario. We opted to study ViTs since there are several works tackling CNNs while virtually no one focused to ViTs yet although they are getting consistently better at vision tasks.

This work takes an initial step towards a surgical investigation of the self attention mechanism (SAM) for designing coherent continual learning methods in ViTs. We first carry out an evaluation of established continual learning regularization techniques. We then examine the effect of regularization when applied to two key enablers of SAM: (a) the contextualized embedding layers, for their ability to capture well-scaled representations with respect to the values, and (b) the prescaled attention maps, for carrying value-independent global contextual information. We depict the perks of each distilling strategy on two image recognition benchmarks (CIFAR100 and ImageNet-32) – while (a) leads to a better overall accuracy, (b) helps enhance the rigidity by maintaining competitive performances. Furthermore, we identify the limitation imposed by the symmetric nature of regularization losses. To alleviate this, we propose an asymmetric variant and apply it to the pooled output distillation (POD) loss adapted for ViTs. As we will see through the section, our experiments confirm that introducing asymmetry to POD boosts its plasticity while retaining stability across (a) and (b). Moreover, we acknowledge low forgetting measures for all the compared methods, indicating that ViTs might be naturally inclined continual learners.

Transformers have shown excellent results for a wide range of language tasks [Brown et al., 2020, Roy et al., 2021] over the course of the last couple of years. Influenced by their initial results, Dosovitskiy *et al.* [Dosovitskiy et al., 2021] proposed Vision Transformers (ViTs) as the first firm yet competitive application of transformers within the computer vision community.<sup>2</sup> ViTs' applications have since spanned a range of vision tasks, including, but not limited to image classification [Touvron et al., 2021], object recognition [Liu et al., 2021], and image segmentation [Wang et al., 2021]. The singlemost essential element of their architecture remains the self-attention mechanism (SAM) that allows the learning of long-range interdependence between the elements of a sequence (or patches of an image). Another feature vital to their performance is the way they are pretrained in an often unsupervised or self-supervised manner over a large amount of data. This is then followed by the finetuning stage where they are adapted to a downstream task [Devlin et al., 2019].

For ViTs to be able to operate in real-world scenarios, they must exploit streaming data, *i.e.*, sequential availability of training data for each task.<sup>3</sup> Storage limitations or privacy constraints further imply the restrictions on the storage of data from previous tasks. Task-incremental continual learning (CL) seeks to find solutions to such constraints by alleviating the event of *catastrophic forgetting* - a phenomena where the network has a dramatic drop in performance on data from previous tasks. Several solutions have been proposed to address forgetting, including regularization [Kirkpatrick et al., 2017, Aljundi et al., 2018, Zenke et al., 2017, Ritter et al., 2018], data replay [Chaudhry et al., 2019b, Aljundi et al., 2019a, Lopez-Paz and Ranzato, 2017] and parameter isolation [Mallya and Lazebnik, 2018, Rusu et al., 2016, Aljundi et al., 2017, Lee et al., 2020]. Most works on CL *de nos jours* study recurrent [Sodhani et al., 2020, Chiaro et al., 2020] and convolutional neural networks (CNNs) [Kirkpatrick et al., 2017]. However, little has been done to investigate different CL settings in the domain of ViTs. We, therefore, mark the first step for the domain by considering the further restrictive setting of *exemplar-free* CL with a zero overhead of storing any data from previous tasks. We consider this restriction for its real-world aptness to scenarios involving privacy regulations and/or data security considerations.

Given that regularization-based methods form one of the main techniques for exemplar-free CL, we consider an in-depth analysis of these for ViTs. Regularization-based techniques are mainly organized along two branches: *weight regularization* methods (such as EWC [Kirkpatrick et al., 2017], SI [Zenke et al., 2017], MAS [Aljundi et al., 2018]) and *functional regularization* methods (such as LwF [Li and Hoiem, 2017], PODNET [Douillard et al., 2020]). As discussed above, the architectural

---

<sup>2</sup>By firmness, we refer to the non-reliance on convolutional operations.

<sup>3</sup>A task may encompass training data of one or more classes.

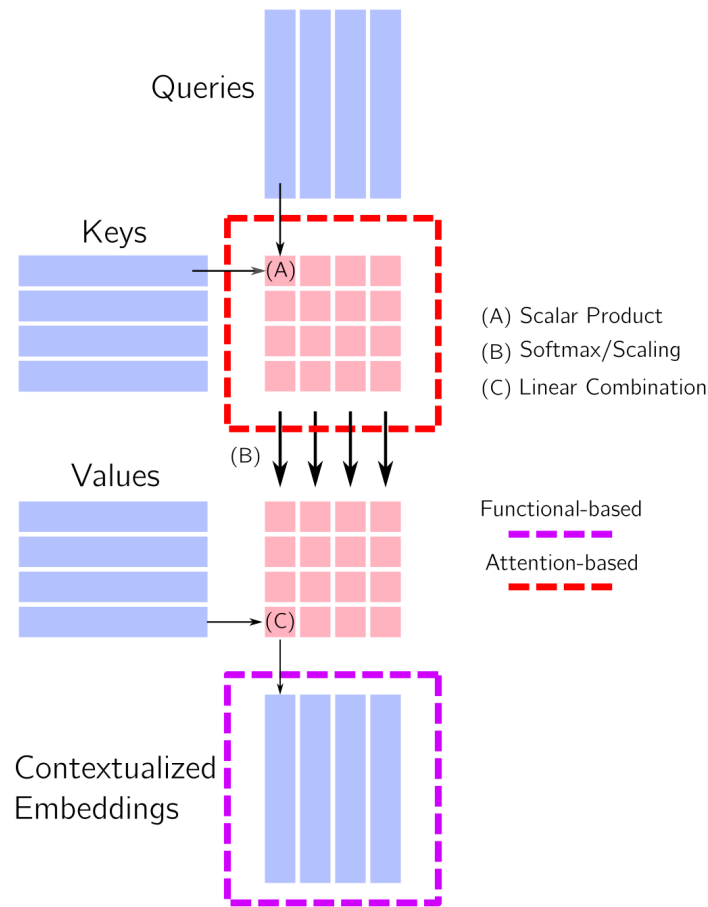


Figure 4.7: Self-attention mechanism comprising a vision transformer encoder. We compare **Attention**-based approaches computed prior to the softmax operation and **Functional**-based approaches computed on the contextualized embeddings.

novelty of transformers lies in the SAM building a representation of a sequence by exhaustively learning relations among query-key pairs of its elements [Vaswani et al., 2017]. We show that for ViTs (and subsequently, all other architectures leveraging SAM), this property allows for a third form of regularization, which we coin *Attention Regularization* (see Figure 4.7). We ground our idea in the hypothesis that when learning new tasks, the attention of the new model should still remain in the neighborhood of the attention of the previous model. As another contribution, we question the temporal symmetry currently applied to regularization losses; referring to the fact that they penalize the forgetting of previous knowledge and the acquiring of new knowledge equally (see Figure 4.8). With the aim of countering forgetting while mitigating the loss of plasticity, we then propose an *asymmetric* regulariza-

tion loss that penalizes the loss of previous knowledge but not the acquiring of new knowledge. We index the major contributions of our work below:

- We are the first to investigate continual learning in vision transformers in the more challenging *exemplar-free* setting. We perform a full analysis of regularization techniques to counter catastrophic forgetting.
- Given the distinct role of self-attention in modeling short and long-range dependencies [Yang et al., 2021], we propose distilling the attention-level matrices of ViTs. Our findings show that such distillation offers accuracy scores on par with that of their more common functional counterpart while offering superior plasticity and forgetting. Motivated by the work of Douillard *et al.* [Douillard et al., 2020], we pool spatiality-induced attention distillation across our network layers.
- We propose an asymmetric variant of functional and attention regularization which prevents forgetting while maintaining higher plasticity. Through our extensive experiments, we show that the proposed asymmetric loss surpasses its symmetric variant across a range of task incremental settings.

## Related Works

Continual learning has been gaining contributions from the deep learning research community during the last few years. In the following, we list the most prominent ones:

- Weight-based: these methods operate in the parameter space of the model through gradient updates. Elastic Weight Consolidation (EWC) [Kirkpatrick et al., 2017] and Synaptic Intelligence (SI) [Zenke et al., 2017] are two widely used methods in this family with the former being probably, the most well-known. EWC uses fisher information to identify the parameters important to individual tasks and penalizes their updates to preserve knowledge from older tasks. SI makes the neurons accumulate and exploit old task-specific knowledge to contrast forgetting.
- Functional-based: these methods rely upon trading the plasticity for stability by training either the current (new) model on older data or vice-versa. Learning Without Forgetting (LWF) [Li and Hoiem, 2017] remains among the most

widely known approaches in this family. It employs Knowledge Distillation [Hinton et al., 2015] upon the logits of the network.

- Parameter Isolation-based: also known as architectural approaches, these methods tackle CF through a dynamic expansion of the network’s parameters as the number of tasks grow. Among the first widely known methods in this family remain Progressive Neural Network (PNN) [Rusu et al., 2016] followed by Dynamically Expandable Network (DEN) [Yoon et al., 2018] and Reinforced Continual Learning (RCL) [Xu and Zhu, 2018].

The majority of the aforementioned works target CL in CNNs mainly due to their inductive bias allowing them to solve almost all problems that involve visual data. This can also be seen in several reviews [Mai et al., 2022, Biesialska et al., 2020, Delange et al., 2021, Parisi et al., 2019, Belouadah et al., 2021, Mai et al., 2022] reporting few approaches that consider architectures besides CNNs, despite the attempts to investigate CL in RNNs [Sodhani et al., 2020, Chiaro et al., 2020].

Only recently have some works analyzed catastrophic forgetting in transformers. Among the earliest to do so remains that of Li *et al.* [Li et al., 2022] proposing the continual learning with transformers (COLT) framework for object detection in autonomous driving scenarios. Using the Swin Transformer [Liu et al., 2021] as the backbone for a CascadeRCNN detector, the authors show that the extracted features generalize better to unseen domains hence achieving lesser forgetting rates compared to ResNet50 and ResNet101 [He et al., 2016] backbones. In case of ViTs, Yu *et al.* [Yu et al., 2021] show that their vanilla counterparts are more prone to forgetting when trained from scratch. Alongside heavy augmentations, they employ a set of techniques to mitigate forgetting: (a) knowledge distillation, (b) balanced re-training of the head on exemplars (inspired by LUCIR [Hou et al., 2019]), and (c) prepending a convolutional stem to improve low-level feature extraction of ViTs.

In their work studying the impact of model architectures in CL, Mirzadeh *et al.* [Mirzadeh et al., 2022] also experiment with ViTs in brief (with the rest of the work focusing mainly on CNNs). While they vary the number of attention heads of ViTs to show that this has little effect on the accuracy and forgetting scores, they further conclude that ViTs do offer more robustness to forgetting arising from distributional shifts when compared with their CNN-based counterparts with an equivalent number of parameters. The conclusion remains in line with previous works [Paul and Chen, 2021]. Finally, [Douillard et al., 2021] attempt to overcome forgetting in ViTs through a parameter-isolation approach which dynamically expands the tokens processed by the last layer. For each task, they learn a new task-specific token per head. They then couple such approach through the usage of exemplars and knowledge dis-

tillation on backbone features. It is worth noting that these works rely either on pretrained feature extractors [Li et al., 2022] or rehearsal [Yu et al., 2021, Douillard et al., 2021] to defy forgetting. Thus the challenging scenario of *exemplar-free* CL in ViTs remains unmarked.

## Methodology

We start by shortly describing the two main existing regularization techniques for continual learning. We then propose attention regularization as an alternative approach tailored for ViTs. Lastly, we put forward an adaptation for functional and attention regularization which is designed to elevate plasticity while retaining stability levels.

### Functional and Weight Regularization

**Functional Regularization:** We include LwF [Li and Hoiem, 2017] in this component since it constitutes one of the most prominent, and perhaps the most widely used regularization method acting on data. The appealing property of LwF lies in the fact it is exemplar-free, *i.e.*, it uses only the data of the current task and maintains only the model at task  $t - 1$  to exploit Knowledge Distillation [Hinton et al., 2015]. Formally, LwF can be defined as:

$$\mathcal{L}_{\text{LwF}}(\theta) = \lambda_o \mathcal{L}_{\text{KD}}(Y_o, \hat{Y}_o) + \mathcal{L}_{\text{CE}}(Y_n, \hat{Y}_n) + \mathcal{R}(\theta) \quad (4.3)$$

where  $\mathcal{L}_{\text{KD}}$  is the knowledge distillation loss incorporated to impose stability on the outputs,  $\hat{Y}_o$  the predictions on the current task data from the old model and  $Y_o$  the ground truth of such data.  $\lambda_o$  remains the temperature annealing factor for softmax logits while  $\mathcal{L}_{\text{CE}}$  is the standard cross entropy loss calculated upon the new task examples.

**Weight Regularization:** These methods encourage the network to adapt to the current task data mainly by using those parameters of the network that are not considered important for previous tasks. As representative method we select EWC [Kirkpatrick et al., 2017]. EWC exploits second-order information to estimate the importance of parameters for the current task. The importance is approximated by

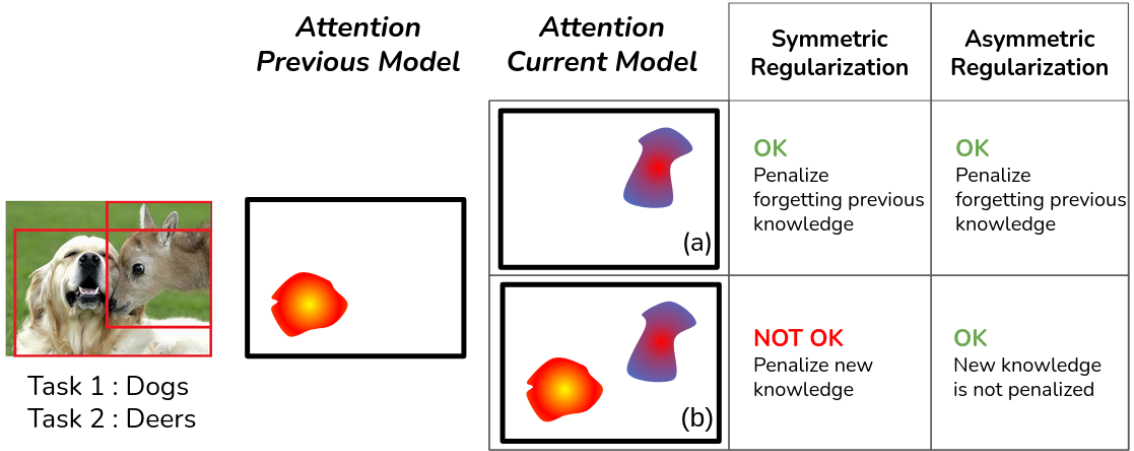


Figure 4.8: Visual illustration of the asymmetric loss. The image considers two generated attention maps (a) and (b) while training task 2. In case (a), when previous knowledge is lost, both the symmetric and asymmetric regularization work correctly. However, in case (b), when new knowledge is acquired, this is penalized by the symmetric loss but not by the asymmetric loss. The idea is that the asymmetric loss leads to higher plasticity without hurting stability.

the diagonal of the Fisher Information Matrix  $F$ :

$$\mathcal{L}_{\text{EWC}}(\theta) = \mathcal{L}_X(\theta) + \sum_j \frac{\lambda}{2} F_j (\theta_j - \theta_{Y,j}^*)^2 \quad (4.4)$$

where  $\mathcal{L}_X(\theta)$  is the loss for task  $X$ ,  $\lambda$  the regularization strength, and  $\theta_{Y,j}^*$  the optimal value of  $j^{\text{th}}$  parameter after having learned task  $Y$ .

## Attention Regularization

**Self-Attention Mechanism:** The self-attention mechanism (SAM) [Vaswani et al., 2017] forms the core of Transformer-based models and can be defined as:

$$\mathbf{z} = \text{softmax} \left( \frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_e}} \right) \mathbf{V} \quad (4.5)$$

where  $\mathbf{Q}$ ,  $\mathbf{K}$ , and  $\mathbf{V}$  are respectively the projections of the Query, Key, and Values of the  $\mathbb{R}^{d_e}$  input embeddings while  $\mathbf{z}$  constitutes the new contextualized embeddings. Our novel attention-based regularization intervenes prior to the computation



of the softmax operation of the standard self-attention mechanism as illustrated in Figure 4.7.

In particular, given a ViT model at incremental step  $t$  and an SAM head  $k$  of layer  $l$ , we define the prescaled attention matrix  $\mathbf{A}_{k^l}^t$  prior to the softmax operation as:

$$\mathbf{A}_{k^l}^t = \frac{\mathbf{QK}^T}{\sqrt{d_e}} \quad (4.6)$$

We denote the attention matrix corresponding to the model at time step  $(t - 1)$  computed in a similar way as  $\mathbf{A}_{k^l}^{t-1}$ . We employ this predecessor in the calculation of knowledge distillation in what follows.

**Pooled Attention Distillation:** Functional approaches leverage network’s submodules typically to apply knowledge distillation [Hinton et al., 2015]. When the regularization takes place in intermediate layers, the model can experience excessive stability, therefore losing in plasticity abilities [Douillard et al., 2020, Liu et al., 2020a, Yu et al., 2020b]. Amongst these methods, PODNet [Douillard et al., 2020] clearly identifies the problem of excessive stability. We devise a regularization approach which instead of regularizing functional submodules targets attention maps, the core mechanisms of SAMs.

More formally, given the attention maps at steps  $t$  and  $(t - 1)$ , we define  $\mathcal{L}_{\text{PAD}}(\mathbf{A}_{k^l}^{t-1}, \mathbf{A}_{k^l}^t)$  [Douillard et al., 2020] to be:

$$\mathcal{L}_{\text{PAD-width}}(\mathbf{A}_{k^l}^{t-1}, \mathbf{A}_{k^l}^t) + \mathcal{L}_{\text{PAD-height}}(\mathbf{A}_{k^l}^{t-1}, \mathbf{A}_{k^l}^t) \quad (4.7)$$

$$\begin{aligned} \text{where } \mathcal{L}_{\text{PAD-width}}(\mathbf{A}_{k^l}^{t-1}, \mathbf{A}_{k^l}^t) &= \sum_{h=1}^H \mathcal{D}_W(\mathbf{A}_{k^l}^{t-1}, \mathbf{A}_{k^l}^t), \\ \mathcal{L}_{\text{PAD-height}}(\mathbf{A}_{k^l}^{t-1}, \mathbf{A}_{k^l}^t) &= \sum_{w=1}^W \mathcal{D}_H(\mathbf{A}_{k^l}^{t-1}, \mathbf{A}_{k^l}^t), \end{aligned} \quad (4.8)$$

$$\mathcal{D}_X(\mathbf{A}_{k^l}^{t-1}, \mathbf{A}_{k^l}^t) = \left\| \sum_{x=1}^X \mathbf{A}_{k^l, w, h}^{t-1} - \sum_{x=1}^X \mathbf{A}_{k^l, w, h}^t \right\|^2 \quad (4.9)$$

where,  $W$  and  $H$  indicate the width and height dimensions of the attention maps, and  $\mathcal{D}_X(a, b)$  is the sum total of the distance measure between maps  $a$  and  $b$  along  $X$ -th dimension. As shown in equation 4.9, the standard  $\mathcal{L}_{\text{PAD}}$  uses the difference

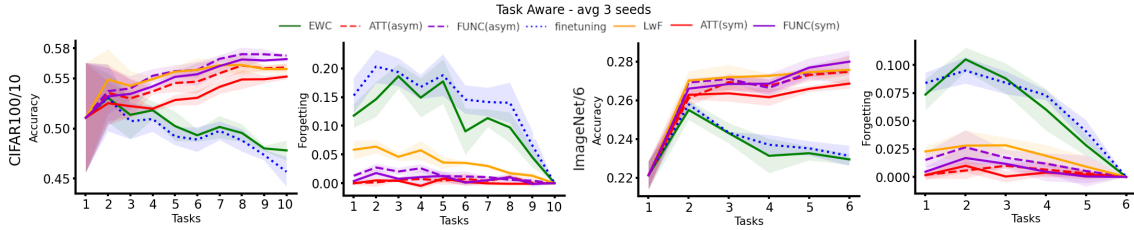


Figure 4.9: Mean and standard deviation of task-aware accuracy and forgetting scores for CIFAR100/10 and ImageNet/6 settings (over 3 random runs). Asymmetric approaches depict higher accuracy with respect to their symmetric counterparts. The low forgetting scores across all methods suggest an intrinsic forgetting resilience in vision transformer architectures.

operator as the choice for  $\mathcal{D}$ . We now point out the limitation of such symmetric  $\mathcal{D}$  and introduce in the next section the notion of *asymmetry* into our distance measure.

As previously mentioned, Douillard *et al.* [Douillard et al., 2020] propose the pooled outputs distillation PODNet loss which leverages the symmetric Euclidean distance between the L2-normalized outputs of the convolutional layers of models at  $t$  and  $(t - 1)$  after pooling them along specific dimension(s). They achieve the best results upon combining the pooling along the spatial width and height axes which they term as the POD-spatial loss. Given the generic correspondence among the various pooling variants in their paper, our work is particularly influenced by POD-spatial as we pool attention maps of ViTs along two dimensions. In fact, throughout the experiments, we analyze this formulation when applied to the contextualized embeddings  $\mathbf{z}$  resulting from a SAM operation. We would like to highlight that PAD differs from PODNet in two important factors: its applied to the attention and not directly on the layer output, and secondly, its marginalization is not on the spatial dimensions due to the fact that  $\mathbf{z}$  does not encode the spatial dimension.

## Asymmetric Regularization

The proposed attention regularization prevents forgetting of previous task by ensuring that the old attention maps be retained while the model learns to attend to new regions over tasks. However, the symmetric nature of  $\mathcal{D}_X$  (with respect to the two attention maps) means that any differences between the older and the newly learned attention maps lead to increased loss values (see Equation 4.8). We agree that penalizing a loss in attention with respect to previous knowledge is crucial in

addressing forgetting. However, also penalizing a gain in attention for newly learned knowledge is undesirable and may actually hurt the performance over subsequently learned tasks. In other words, punishing additional attention can be counterproductive. As a result, we propose using an asymmetric variant of  $\mathcal{D}_X$  that can better retain previous knowledge:

$$\mathcal{D}_X(\mathbf{A}_{k'}^{t-1}, \mathbf{A}_{k'}^t) = \left\| \mathcal{F}_{\text{asym}} \left( \sum_{x=1}^X \mathbf{A}_{k',w,h}^{t-1} - \sum_{x=1}^X \mathbf{A}_{k',w,h}^t \right) \right\|^2 \quad (4.10)$$

where,  $\mathcal{F}_{\text{asym}}$  is an asymmetric function. We experimented with ReLU [Nair and Hinton, 2010], ELU [Clevert et al., 2016] and Leaky ReLU [Maas et al., 2013] as choices for  $\mathcal{F}_{\text{asym}}$  and found that in general, ReLU performed the best across our settings. By introducing the ReLU function, new attention generated by the current model at task  $t$  is not penalized. Attention present at task  $t - 1$  but missing in the current model  $t$  is penalized. An illustration of the functioning of the new loss is provided in Figure 4.8.

Based on our choice for  $\mathcal{D}_X$  from equations 4.9 and 4.10, we classify our final PAD loss as symmetric  $\mathcal{L}_{\text{PAD-sym}}$  or asymmetric  $\mathcal{L}_{\text{PAD-asym}}$ , respectively. Each of these losses are computed separately for each of the SAM head and model layer. The final asymmetric variant can thus be stated as:

$$\begin{aligned} \mathcal{L}_{\text{PAD-asym}}(\mathbf{A}_{k'}^{t-1}, \mathbf{A}_{k'}^t) = \\ \frac{1}{L} \sum_1^L \frac{1}{K} \sum_1^K \mathcal{L}_{\text{PAD}}(\mathbf{A}_{k'}^{t-1}, \mathbf{A}_{k'}^t) \end{aligned} \quad (4.11)$$

where,  $K$  is the total number of heads per layer and  $L$  is the total number of layers of the model. Note that equation 4.11 can be adapted for  $\mathcal{L}_{\text{PAD-sym}}$  without loss of generality.

**Overall loss:** We augment the asymmetric and symmetric PAD losses from equation 4.11 with knowledge distillation loss  $\mathcal{L}_{\text{LwF}}$  [Li and Hoiem, 2017] and standard cross entropy loss  $\mathcal{L}_{\text{CE}}$ . The overall loss term takes the form:

$$\mathcal{L} = \mu \mathcal{L}_{\text{PAD-(a)sym}} + \lambda \mathcal{L}_{\text{LwF}} + \mathcal{L}_{\text{CE}} \quad (4.12)$$

where  $\mu, \lambda \in [0, 1]$  are two hyperparameters regulating the respective contributions. Note that when  $\mu = 0$ ,  $\mathcal{L}$  degenerates to baseline finetuning for  $\lambda = 0$  and to LwF for  $\lambda = 1$ .

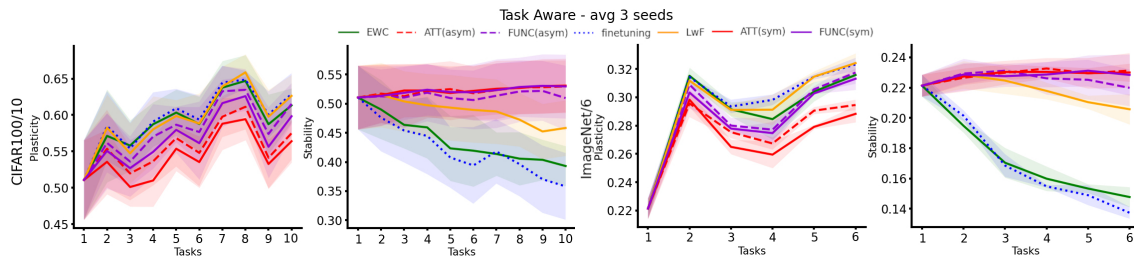


Figure 4.10: Mean and standard deviation of task-aware plasticity-stability scores for CIFAR100/10 and ImageNet/6 settings (over 3 random runs). Asymmetric approaches are more plastic compared to their symmetric counterparts while retaining competitive stability.

**Stability-Plasticity Curves:** Several measures have been proposed in the CL literature to assess the performance of an incremental learner. Besides the standard incremental accuracy, Lopez-Paz et al [Lopez-Paz and Ranzato, 2017] introduce the notion of Backward Transfert (BWT) and Forward Transfert (FWT). BWT measures the ability of a system to propagate knowledge to past tasks, while FWT assesses the ability to generalize to future tasks. The CL community, however, still lacks consensus on a specific definition of the stability-plasticity dilemma. An elemental formulation for such quantification is thus desirable for allowing us to better grasp the balancing capabilities of an incremental learner at acquiring new knowledge without discarding previous concepts. To this end, we introduce *stability-plasticity curves* computed using task accuracy matrices.

A task accuracy matrix  $\mathbf{M}$  for an incremental learning setting composed of  $T$  tasks is defined to be a  $[0, 1]^{T \times T}$  matrix, whose entries are the accuracies computed at each incremental step.<sup>4</sup> For instance,  $\mathbf{M}_{i,j}$  would constitute the test accuracy of task  $j$  when the system is learning task  $i$ . Subsequently, the diagonal entries of  $\mathbf{M}_i, i$  give us the accuracies at the respective current tasks while the entries below the diagonal, *i.e.*,  $j < i$ , give the performance of the model on past tasks. A visual depiction can be seen in Figure 4.11.

We define the stability to be the performance on the first experienced task at any given time and plasticity to be the ability of the model to adapt to the current task. Namely, these constitute the first column  $\mathbf{M}_{:,0}$  and the diagonal of the matrix  $diag(\mathbf{M})$ . We employ the curves derived from these definitions to better dissect the stability-plasticity dilemma of the methods analyzed in our work.

<sup>4</sup>This calls for  $\mathbf{M}$  to be lower trapezoidal.

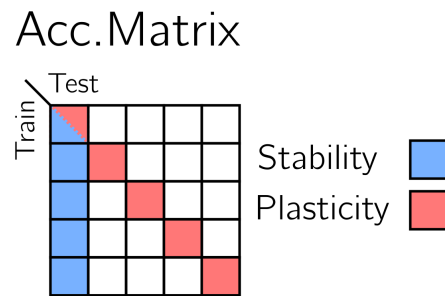


Figure 4.11: Illustration of a task accuracy matrix: we fix **stability** to be the performance of the first task across time steps while we define **plasticity** to be the performance at the current step.

## Experiments

In this section, we compare regularization-based methods for *exemplar-free* continual learning. We evaluate the newly proposed attention regularization and compare it with the existing functional (*LwF*) and feature regularization methods. We then ablate the usefulness of the newly proposed asymmetric loss as well as the importance of pooling before applying the regularization.

### Experimental Setup

**Setting:** For our experiments, we adopt the variation of ViTs introduced by Xiao *et al.* [Xiao *et al.*, 2021]. Here, the standard linear embedder of a ViT model is replaced by a smaller convolutional stem which helps build more resilient low-level features. Convolutional stems have previously been shown to improve performance and convergence speed in incremental learning settings [Yu *et al.*, 2021]. We therefore define our architecture to be a lightweight variation of a ViT-Base by setting  $L = 12$  layers,  $K = 12$  heads per layer and a  $d_e = 192$  embedding size. The choice of a small embedding size has been made to speed up the training procedure and unlock the ability to handle larger batch sizes (1024 for our work).

We analyze our task-incremental setting on two widely used image recognition datasets - namely CIFAR100 and ImageNet-32 with 100, and 300 classes each. Both datasets host  $32 \times 32$  images. On CIFAR100, we consider a split of 10 tasks (denoted as CIFAR100/10 setting) where each incremental task is composed of 10 disjoint set of classes. On ImageNet-32, we split 6 tasks with 50 disjoint set of classes each

(denoted as ImageNet/6).<sup>5</sup>

Our total training epochs remain 200 (per task) for CIFAR100 and 100 for ImageNet32 with an initial learning rate of 0.01 and patience set of 20 epochs. We report our scores averaged over 3 random runs. We apply a constant padding of size 4 across all our datasets. The train images are augmented using random crops of sizes  $32 \times 32$  and random horizontal flips with a flipping probability of 50%. For test images, we only apply center crops of sizes  $32 \times 32$ .

We compare the attentional and functional symmetric and asymmetric versions of  $\mathcal{L}_{\text{PAD-(a)sym}}$ . We use LwF [Li and Hoiem, 2017] and EWC [Kirkpatrick et al., 2017] as our basic functional and weight regularization approaches. For all our experiments relying on PAD losses, we performed a hyperparameter search (using equation 4.12) for  $\mu$  and  $\lambda$  by varying each in the range  $[0.5, 1.0]$  and found  $\mu = \lambda = 1.0$  to perform reasonably well. We thus stick to these values unless otherwise specified. For the sake of brevity, we indicate  $\mathcal{L}_{\text{PAD-asym}}$  with `Asym_att` and  $\mathcal{L}_{\text{PAD-sym}}$  with `Sym_att`. Note that these are both variations of equation 4.12. The functional approaches are analogous to their attentional counterparts except for the fact that they rely on the regularization of the contextualized embeddings rather than the attention matrix (see Figure 4.7). The latter correspond to `Asym_func` and `Sym_func` accordingly.

## Results

We report accuracy as well as forgetting [Chaudhry et al., 2018] scores in task aware (tau) setting<sup>6</sup>. We further report tau plasticity-stability curves (based on Figure 4.11) to provide insights upon how well the different models handle the trade-off.

**Accuracy and Forgetting:** As seen in Figure 4.9, all asymmetric approaches show better performances with respect to their symmetric counterparts on CIFAR100/10 with `Asym_att` offering the best accuracy of 57.3% on the last task. The trend continues for ImageNet/6 with an exception of asymmetric functional approach with an accuracy of 27.55% falling behind its symmetric counterpart by 0.44%. In general, the asymmetric and symmetric losses lead to improved accuracy scores with respect to other methods. Moreover, we observe that all the methods depict good forgetting resilience with their forgetting scores running around  $\approx 0.01\%$ ) except for EWC. This suggests us that vision transformers are *better incremental learners* but require more

---

<sup>5</sup>Refer to Section 4.2 for experiments on additional settings.

<sup>6</sup>The corresponding task agnostic scores can be found in Figure 4.14, Section 4.2.

CIFAR100/10 (taw)					
	<i>Asym_Func Spatial</i>	<i>Sym_Func Spatial</i>	<i>Asym_Func Intact</i>	<i>Sym_Func Intact</i>	<i>LwF</i>
<i>Average Incr. Accuracy</i>	<b>56.18%</b>	55.67%	54.43%	53.12%	55.11%
<i>Last Task Accuracy</i>	<b>57.26%</b>	56.92%	56.04%	54.59%	55.93%

Table 4.7: Comparison of intact (no pooling), spatial (pooling along width and height), and LwF.

training and tuning efforts to achieve reasonable accuracies. This remark remains in accordance with prior studies [Mirzadeh et al., 2022, Paul and Chen, 2021]. In the particular case of EWC, we observe poor compatibility in terms of accuracy as well as forgetting – with the scores falling behind finetuning at times. We suspect that the method might not be less suited for ViTs due to its reliance on exhaustive fisher information estimation.

**Plasticity-stability tradeoff:** We compare the dilemma for various methods in Figure 4.10. With no distillation, finetuning is prone to the worst trading of plasticity for stability. Meanwhile, our asymmetric losses can be seen to be more plastic with respect to their symmetric counterparts while depicting comparable stability scores. This confirms our hypothesis regarding the nature of the asymmetry keeping it from discarding older attention while favoring the integration of new attention at the same time. Although, LwF with a last task score of 47.74% on CIFAR100/10 and 32.0% on ImageNet/6, reports the best plasticity among our approaches, it clearly lags behind the pooling-based approaches at retaining stability. On the contrary, the (a)symmetric attention losses and the symmetric functional loss perform similar with a last task stability score of  $\approx 0.23\%$  on ImageNet/6 and  $\approx 53\%$  on CIFAR100/10. EWC shows good plasticity but virtually zero stability. This trend is in line with our previous comment on the limitation of EWC in Figure 4.9.

## Ablation study

Towards the end goal of evaluating the effectiveness of PAD losses, we ablate the contribution of pooling on the CIFAR100/10 setting. In particular, we consider distilling the attention maps when these are: (a) pooled along both dimensions, *i.e.*, (A)sym\_Func Spatial (see Equation 4.7), and (b) not pooled at all, *i.e.*,

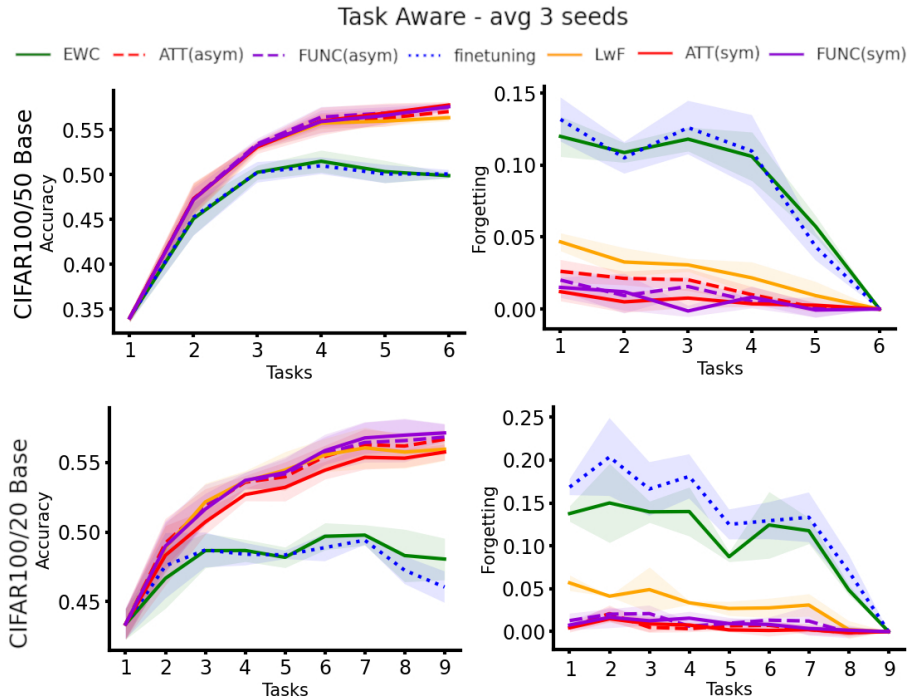


Figure 4.12: Mean and standard deviation of task-aware accuracy and forgetting scores for the additional CIFAR100/20 and CIFAR100/50 settings (over 3 random runs).

(A)sym\_Func Intact. Distilling the intact maps of the latter setting imply enhanced stability over their pooled counterparts. Our standard accuracy and plasticity-stability measures across tasks can therefore be deemed redundant in this setting. As a consequence, we choose to compare the task-aware average incremental accuracy [Rebuffi et al., 2017] and the last task accuracy across (a) and (b) while contrasting these with LwF as a strong baseline. For further crisper observations, we limit our comparisons to the functional setting. As shown in Table 4.7, we find that Asym\_Func Spatial consistently performs the best across both the metrics (with a gain of  $> 2\%$  over Sym\_Func Intact in either metric). In general, distilling the intact attention maps can be seen to be hurting the performance of the models as their accuracy drop below that of the baseline LwF.

## Conclusion

In this work, we adapted and analyzed several continual learning methods to counter forgetting in Vision Transformers mainly with the help of regularization. We then



introduced a novel PODNet-inspired regularization, based on the attention maps of self-attention mechanisms which we termed as Pooled Attention Distillation (PAD). Shedding light on its limitation at learning new attention, we devised its asymmetric version that avoids penalizing the addition of new knowledge in the model. We validated the superior plasticity of the asymmetric loss on several benchmarks.

Besides the meticulous comparison of a range of regularization approaches, *i.e.*, functional (LwF), weight (EWC), and the proposed attention-based regularization, we extended the application of PAD to the functional submodules of ViTs. To this end, we investigated regularization in the contextualized embeddings of ViTs. The latter exploration led us to discover that the regularization of functional submodules can help achieve the best overall performances while the regularization of their attentional counterparts endow CL models with superior stability. Finally, we remarked the low forgetting scores of vision transformers across the incremental tasks and concluded that their enhanced generalization capabilities may endow them with a natural inclination for incremental learning. By making our code open-source, we hope to open the doors for future research along the direction of efficient continual learning with transformer-based architectures.

## Additional Settings

We experiment on two further CIFAR100 settings with distinct cardinality of base task classes:

- CIFAR100/20 Base, with 20 base task classes followed by 8 incremental tasks with 10 classes each,
- CIFAR100/50 Base, with 50 base task classes followed by 5 incremental tasks with 10 classes each.

The task aware accuracy and forgetting scores on these are shown in Figure 4.12. We find the PAD-based losses to consistently outperform other regularization approaches with LwF being the closest tie. Along the direction of plasticity-stability tradeoff (see Figure 4.13), we observe that: (a) the attentional PAD losses retain better rigidity than their functional counterparts, and (b) the asymmetric variants of PAD losses are more plastic than their symmetric counterparts across these settings. These trends further validate our hypotheses in sections 4.2 and 4.2, respectively.

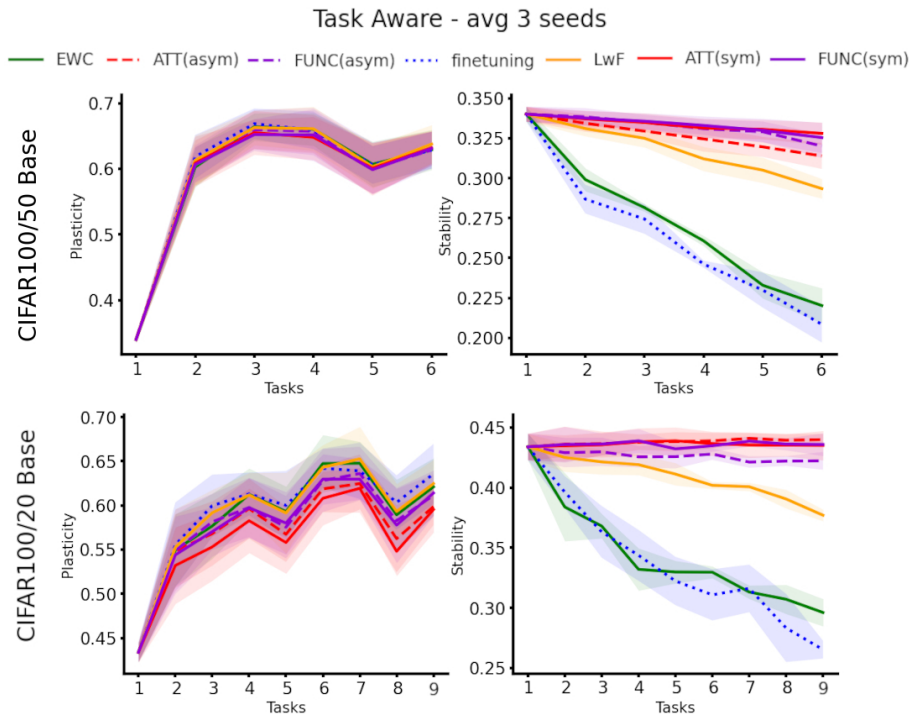


Figure 4.13: Mean and standard deviation of task-aware plasticity-stability scores for the additional CIFAR100/20 and CIFAR100/50 settings (over 3 random runs).

## Task Agnostic Results

Figure 4.14 depicts the task-agnostic accuracy and forgetting scores for the settings mentioned in the main section as well as in Section 4.2. Given the contradictory terms of resource-scarce *exemplar-free* CL and data-hungry ViTs, task-agnostic evaluations can be seen to be particularly challenging. The further avoidance of heavier data augmentations in our training settings can be seen to give rise to two major repercussions across the task-agnostic accuracies: (a) the scores remain consistently low, and (b) the models show smaller yet consistent variations in performances across all settings.

That said, we find functional PAD losses to be performing the best on all but CIFAR100/50 setting. The larger proportion of base task classes in the latter setting can be seen to be greatly benefiting the learning of LwF (the least parameterized loss term). Further on the note of class proportions, we observe that an equal spread of classes across the tasks can be seen to have a smoothing effect on the variations of scores across different methods.

On the contrary, the CIFAR100/50 setting leads to low variability of task-agnostic forgetting scores across the methods. This can again be attributed to the fact that a very large first task better leverages the generalization capabilities of ViTs thus making them better at avoiding forgetting over the subsequent incremental steps. This further adds to our reasoning regarding the natural resilience of ViTs to incremental learning settings. When compared across methods, the attentional variants of PAD losses can be seen to display the least amount of forgetting followed by their functional counterparts.

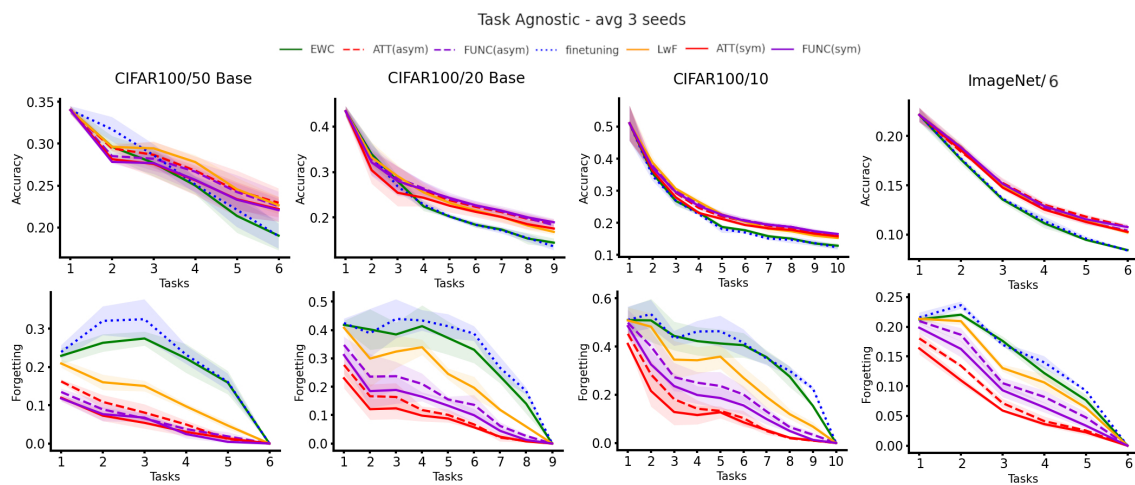


Figure 4.14: Mean and standard deviation of task-agnostic accuracy and forgetting scores for CIFAR100/10, CIFAR100/20, CIFAR100/50, and ImageNet/6 settings (over 3 random runs). The larger proportion of base task classes (for example, CIFAR100/50) gives rise to higher variations of accuracies and lower variation of forgetting scores across methods – with the latter indicating the inclination of ViTs towards better generalization and preservation of knowledge.

### 4.3 Simpler is Better: off-the-shelf Continual Learning through Pretrained Backbones

In this section we propose a simple baseline for continual learning that leverages pretrained backbones. The approach devised is fast, since requires *no parameters updates* and has *minimal memory requirements* (order of KBytes). By providing such a simple baseline, and achieving strong performance on all the major benchmarks used in literature, we follow the concerns raised in Section 4.1 on the simplicity of the benchmarks used. Secondly, we show that pretraining cause the network to generalize at a point where the incremental learning of new tasks is very simple.

In particular, the "training" phase reorders data and exploit the power of pre-trained models to compute a class prototype and fill a memory bank. At inference time we match the closest prototype through a knn-like approach, providing us the prediction. We will see how this naive solution can act as an off-the-shelf continual learning system. In order to better consolidate our results, and merge the above two works, we use the devised pipeline with CNN and Vision Transformers. We will discover that thew latter have the ability to produce features of higher quality. As a side note we discuss some extension to the unsupervised realm.

In a nutshell, this simple pipeline raises the same questions raised by previous works such as [Prabhu et al. \[2020\]](#) on the effective progresses made by the CL community especially in the dataset considered and the usage of pretrained models.

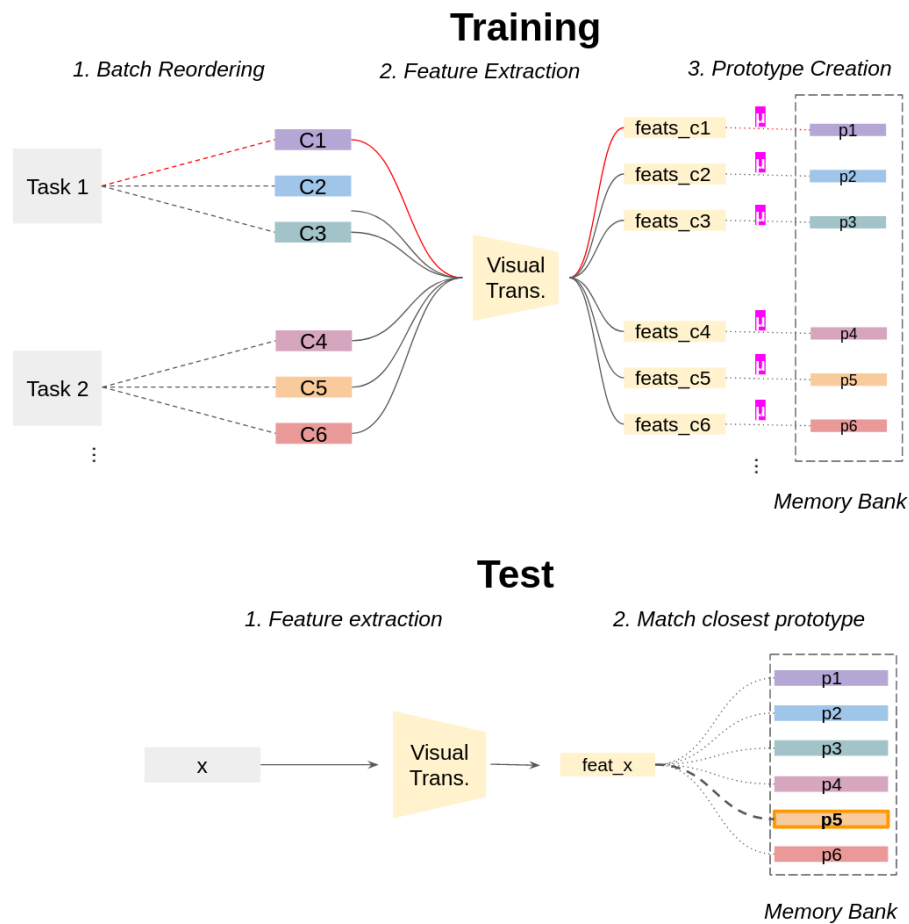


Figure 4.15: Depiction of our simple baseline. Our pipeline does not perform parameters updates and consumes few KBytes as memory bank.

Until now, the CL community mainly focused in the analysis of catastrophic forgetting in Convolutional Neural Networks (CNN) models. But, as can be seen by some recent works, Vision Transformers (ViT) are asserting themselves as a valuable alternative to CNNs for computer vision tasks, sometimes, achieving better performances with respect to CNNs [Chen et al. \[2022\]](#). The power of ViTs lies in their less inductive bias [Morrison et al. \[2021\]](#) and in their subsequent better generalization ability. Thanks to this ability ViTs are naturally inclined continual learners, as pointed in [Section 4.2](#).

In transformer literature, the usage of *pretrained backbones* is becoming a must, in fact, training such systems requires extensive amount of data and careful hyper-parameters optimization. Using pretrained backbones is common also in Computer

Vision communities where CNNs are the main player. In CL literature, the pretraining is frequent, but not constant. It is typically carried on half of the analyzed dataset or through a big initial task that has the objective of facilitating the learning of low level features. The very best results, however, have been achieved when we do not skip pretraining. This can be confirmed by the CVPR 2020 Continual Learning Challenge summary report [Lomonaco et al. \[2022\]](#), where the authors noted that *all the methods* proposed solutions leveraging pretrained backbones.

On top of that, simple baselines sometimes provide better results with respect to overly engineered CL solutions, GDumb [Prabhu et al. \[2020\]](#) is such an example. In the work, the authors showed superior performance against several methods at the state-of-the-art through a system composed just by a memory random sampler and a simple learner (CNN or MLP). From a practical point of view, these methods often constitute a simple, clear, fast, intuitive and efficient solution.

Following these lines, we explore a knn-like method to perform *off-the-shelf online continual learning* leveraging the power of pretrained vision transformers. Our system constitutes a simple and memory-friendly architecture requiring zero parameters updates. Being our work one of the first using ViTs in CL, we propose a robust baseline for future works and provide an extensive comparison against CNNs.

In brevity, the contributions are the following:

- We devise a *simple pipeline* composed by a pretrained feature extractor and an incremental prototype bank. The latter is updated as new data is experienced. The overall cost of the method is in the storage of a pretrained backbone and few Kbytes for the memory bank.
- We devise a *baseline for future CL methodologies* that will exploit pretrained Vision Transformers or Resnets. The baseline is fast and does not require any parameter update, yet achieving robust results in 200 lines of Python, unlocking reproducibility too.
- We provide a *comparison for our pipeline between Resnets and Visual Transformers*. We discover that Vision Transformers produce more discriminative features, appealing also for the CL setting.
- In light of such results, *we arise the same questions, as GDumb [Prabhu et al. \[2020\]](#) does, in the progresses made by the CL community so far specifically in the quality of the datasets and in the usage of pretrained backbones.*

---

**Algorithm 1** Off-the-shelf CL. “Training”

---

```

Require:  $t_i, \phi, \mathcal{M}$ 
for  $t_i \in \mathcal{T}$  do
     $\mathcal{G} = \text{GroupByClass}(t_i)$ 
    for  $g \in \mathcal{G}$  do
         $f = \phi(g)$                                 Extract features
         $p = \mu(f)$                                 Compute mean feature
         $\mathcal{M} \leftarrow p$                         Store prototype in memory

return  $\mathcal{M}$ 

```

---

## Related Works

Only recently few works considered self-attention models in continual learning. Li et al. [Li et al. \[2022\]](#) proposed a framework for object detection exploiting Swin Transformer [Liu et al. \[2021\]](#) as pretrained backbone for a CascadeRCNN detector, the authors show that the extracted features generalize better to unseen domains hence achieving lesser forgetting rates compared to ResNet50 [He et al. \[2016\]](#) backbones. This also follows the conclusions made by Paul and Chen [Paul and Chen \[2021\]](#) on the fact that vision transformers are more robust learners with respect to CNNs.

Several methods in CL use pretrained backbones as feature extractors such as in Hayes et al [Hayes and Kanan \[2020\]](#) or [Aljundi et al. \[2019b\]](#), [Hocquet et al. \[2020\]](#) and sometimes the pretraining is carried on half (or a big portion) the dataset considered, as in PODNet [Douillard et al. \[2020\]](#) or in Yu et al. [Yu et al. \[2021\]](#). For a more complete review on CL methodologies we point out these recent surveys [Parisi et al. \[2019\]](#), [Hadsell et al. \[2020\]](#), [Mundt et al. \[2020\]](#).

A similar study on pretraining for CL has been conducted by Mehta et al. [Mehta et al. \[2021\]](#). In particular, they study the impact on catastrophic forgetting that a linear layer might induce while using a pretrained backbone. Their study focuses only on Resnet18 for vision tasks, but they also include NLP tasks.

## Method

**Setting** Continual Learning characterizes the learning by introducing the notion of subsequent tasks. In particular, the learning happens in an incremental fashion, that is, the model incrementally experiences different training sessions as time advances.

Practically, a learning dataset is split in chunks where each split is considered an incremental task containing data. CL being a relatively new field, the community is still converging to a common setting notation, but we focus on an online, task-agnostic NC-type scenario. That is, the model forwards a pattern just once and does not have the task label at test time. As a more fine grained specific we follow [Lomonaco and Maltoni \[2017\]](#) categorization and use a NC-type scenario where each task contains a disjoint group of classes.

More formally, given a dataset  $\mathcal{D}$  and a set of  $n$  disjoint tasks  $\mathcal{T}$  that will be experienced sequentially:

$$\mathcal{T} = [t_1, t_2, \dots, t_n] \quad (4.13)$$

each task  $t_i = (C_i, D_i)$  represented by a set of classes  $C_i = c_1^i, c_2^i \dots, c_{n_i}^i$  and training data  $D_i$  (images). We assume that the classes of each task do not overlap i.e.  $C^i \cap C^j = \emptyset$  if  $i \neq j$

**“Training” Phase** In the training phase, given a task  $t_i \in \mathcal{T}$ , a feature extractor  $\phi$  and a memory bank as a dictionary  $\mathcal{M}$ , the procedure does the following:

1. First it performs *batch reordering*, that is, it groups the images of a given task by their class
2. After grouping, it forwards each new subset to the feature extractor  $\phi$
3. Given the feature representations of a group, it computes the mean of the features to create a *class prototype*
4. Updates the memory bank  $\mathcal{M}$  by storing the each computed prototype

At the end of the training procedure for a given task  $t_i$ , we would have a representative prototype vector *for each class* contained in  $t_i$ . As we said, the prototype vector is computed as the mean feature representation of the patterns of the same class. A depiction of the “training” phase is reported in [Figure 4.15](#), we also provide a pseudocode in [Algorithm 1](#). We also point out that there is not formal “training” of the network, in fact **we do not perform any parameter update**, we simply exploit the pretrained models and construct a knn-like memory system.



Memory KiB class	Params	Model	CIFAR100	CIFAR10	Core50	Oxford Flowers102	Tiny ImgNet200
2 KiB	11.7M	<i>resnet18</i>	0.53	0.76	0.72	0.73	0.55
2 KiB	21.8M	<i>resnet34</i>	0.55	0.81	0.74	0.67	0.62
8 KiB	25.5M	<i>resnet50</i>	0.59	0.80	0.71	0.70	0.63
8 KiB	60.1M	<i>resnet152</i>	0.67	0.89	0.72	0.66	0.76
0.75 KiB	5.6M	<i>ViT-T/16</i>	0.36	0.63	0.49	0.54	0.24
3 KiB	86.4M	<i>ViT-B/16</i>	0.64	0.87	0.74	0.95	0.63
0.75 KiB	5.6M	<i>DeiT-T/16</i>	0.57	0.80	0.73	0.68	0.64
3 KiB	86.4M	<i>DeiT-B/16</i>	0.68	0.90	0.80	0.74	0.79

Table 4.8: Off-the-shelf accuracy performance on different dataset benchmarks, we both analyzed a CNN model and a ViT pretrained models.

**Test Phase** After completing the training phase for a task  $t_i$  the memory bank  $\mathcal{M}$  will be populated by the prototypes of the classes encountered so far. During this test phase, we simply use a *knn-like approach*. Given an image  $x$ , the updated memory bank  $\mathcal{M}$  and the feature extractor  $\phi$  we devise the test phase as follows:

1. Forward the test image  $x$  to the feature extractor  $\phi$
2. Compute a distance between the feature representation of the image and all prototypes contained in  $\mathcal{M}$
3. We match the prototype with minimum distance and return its class

In a nutshell, we perform k-nn with  $k=1$  over the feature representation of an image, matching the class of the closes prototype in the bank. If the class selected is the same of the test example we would have a hit, a miss otherwise. Figure 4.15 reports a visual depiction of the test procedure. As distance we use a simple  $l^2$ , but several tests have been made with cosine similarity. Although the results with the cosine similarity are better, we opt for the  $l^2$  since provides the best speedup in the implementation through Pytorch.

## Experiments

It is suspected that Visual Transformers generalize better with respect to CNN models. To this end, we compare CNNs models and ViTs models as feature extractors. We selected four CNN models to compare against four attention-based models. In

particular, we selected DeiT-Base/15, DeiT-Tiny/15 [Touvron et al. \[2021\]](#), ViT-Base/16 and ViT-Tiny/16 [Dosovitskiy et al. \[2021\]](#) as visual transformers. While we opted for Resnet18/34/50/152 [He et al. \[2016\]](#) as CNN models. We used the `timm` [Wightman \[2019\]](#) library to fetch the pretrained models where all the models have been trained on ImageNet [Deng et al. \[2009\]](#) and the `continuum` [Douillard and Lesort \[2021\]](#) library to create the incremental setting for 5 datasets, namely CIFAR10/100, Core50, OxfordFlowers102 and TinyImageNet200.

In all dataset benchmarks, we upscaled the images to  $224 \times 224$  pixels in order to accommodate visual transformers which needs such input dimension. We apply such transformation to resnet data too for a fair comparison. In order to match the closes prototype at test time, we used  $l^2$  as preferred measure.

The main results are reported in Table 4.8. The pipeline is extremely simple, yet it achieves *impressive performance* as an off-the-shelf method, at cost of a very small overhead to store the prototype memory. In fact, at the end of the training phase, the memory bank translates only into **few KBytes** of storage. Although this preliminary work only consider task-agnostic setting, we remind that if at test time we are given the task label of the data, we can recast the method to work in task-aware setting. In this case, performing the test phase would be easier since the comparison of the test data will be carried only on a subset of the prototypes. On the same line, one can see that in Table 4.8 we do not report each dataset task split. In fact, our method works for **any dataset split** since it just need any partition of the datasets that respect a NC protocol i.e. as long as tasks are formed by images that can be grouped in classes. We can also appreciate that transformer architectures work best in all benchmarks, suggesting direct *superior generalization capabilities* with respect to CNNs or, at least, more discriminative features.

## Discussion

In light of these results, we think that this work may be extended to be considered as a baseline to assess the performance continual learning methodologies using pretrained networks as feature extractors. In particular, a thorough investigation should be carried by substituting the k-nn approach with a linear classifier, this would allow also a better comparison between resnets and visual transformers. However, we think that these preliminary results are of interest to the Vision Transformer and CL research community.

We then raise some concerns with respect to the procedure and the benchmarks

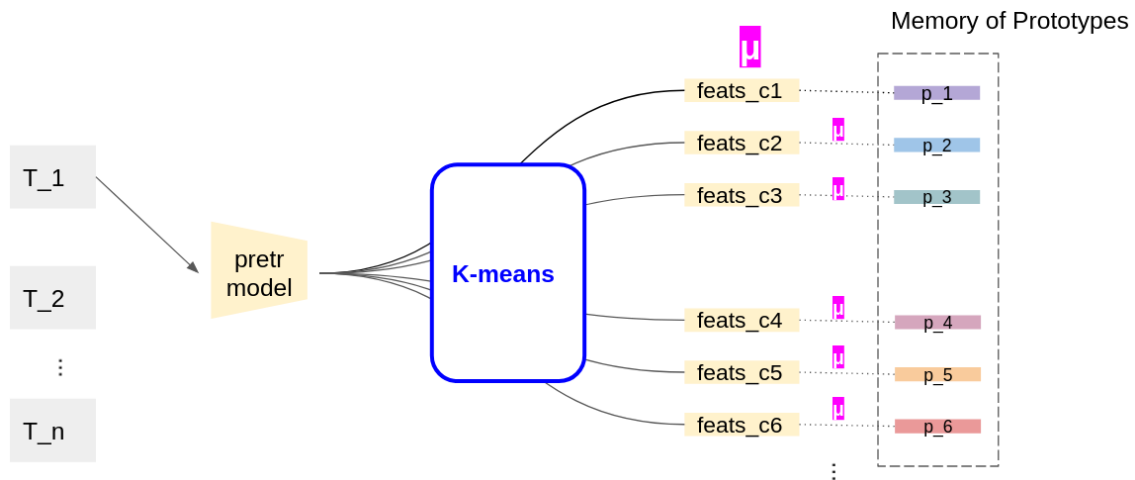


Figure 4.16: Direct off-the-shelf extension of the baseline proposed to tackle unsupervised continual learning.

used to assess new CL methodologies. As we can see, through a pretrained model, we can achieve impressive results with respect to the current CL state-of-the-art [Parisi et al. \[2019\]](#), [Hadsell et al. \[2020\]](#), [Mundt et al. \[2020\]](#). This point have been also raised by GDumb [Prabhu et al. \[2020\]](#) where the authors questioned the progresses by providing a very simple baseline.

Moreover, we can further extend this simple pipeline to be used in *unsupervised continual learning*. Actually, the extension is straightforward. In an unsupervised scenario the batch reordering step cannot be performed since we are not allowed to know each data class label. To cope with this lack of information one can substitute the step with any clustering algorithm such as K-means (we tried it but with no luck) or a more sophisticated approach such as autoencoders, self-organizing maps etc.. The test phase of the unsupervised extension would be analogous to the supervised counterpart.

## Conclusion

In this short ex[erimental] segment we proposed a baseline for continual learning methodologies that exploit pretrained Vision Transformers and Resnets. We tackle online NC-type class-incremental learning scenario, the most common one, even though, our pipeline can be extended to different scenarios. Our off-the-shelf method

is conceptually simple yet gives strong results and can be implemented in 200 lines of Python therefore enhancing reproducibility. To assess the performance of different backbones our pipeline we compared Resnets models against Vision Transformers feature extractors pretrained on the same dataset, and show that vision transformers provide more powerful features. This suggests that Vision Transformers ability to encode knowledge is broader. Then we raise some questions about CL research progress and note that with a pretrained model and a simple pipeline one can achieve strong results and, therefore, new methodologies should drop the usage of pretrained backbones when testing on such dataset benchmarks.

## 4.4 Unsupervised Semantic Discovery through Visual Patterns detection

So far, we directly investigated the impact of performance by altering structural and data properties of object recognition frameworks. If we step back a bit and consider a more broader vision about continual learning, we understand that, in order to adapt to a changing environment, an artificial agent should manifest also the ability to continuously discover new patterns, in our case visual patterns.

We propose a smart pipeline that it is able to discover repetitive patterns in an image, by means of a threshold parameter. That is, if we alter this specific parameter, we are able to discover new semantic levels in a scene. This work goes a bit in another direction from the *dissection* of current continual learning methodologies treated in this thesis. Instead, it is a step towards the ability to build a system able to incrementally explore.

To this end, we propose a new fast fully unsupervised method to discover semantic patterns. Our algorithm is able to hierarchically find visual categories and produce a segmentation mask. Through the modeling of what is a visual pattern in an image, we introduce the notion of “semantic levels” and devise a conceptual framework along with measures and a dedicated benchmark dataset for future comparisons. Our algorithm is composed by two phases. A filtering phase, which selects semantical hotspots by means of an accumulator space, then a clustering phase which propagates the semantic properties of the hotspots on a superpixels basis. We provide both qualitative and quantitative experimental validation, achieving optimal results.

While the vast majority of supervised object detection and segmentation approaches leverage rich datasets with semantically labelled categories, unsupervised methods cannot rely on such a luxury. Indeed they are expected to infer from the image content itself what is a relevant object and which are its boundaries. This is a daunting task, as relevance is totally domain-specific and also highly subjective, especially when taking in account human judgement, which exploits a lot of out-of-band information that cannot be found in the sheer image data.

As a matter of fact, little effort have been put to investigate unsupervised automatic approaches to detect and segment semantically relevant objects without any additional information than the image or any *a priori* knowledge of the context. This is due to the fact that a unique definition of what is a relevant object (or, how we prefer to call it, a *visual category*) does not actually exist.

This is especially true if we are seeking to set a formal definition that can be adopted across all the domains in a consistent manner with respect to human judgement.

Within this section, we try to address this problem by considering a visual category each pattern which appearance is consistent enough across the image. In other words, we consider something to be a relevant object if it appears more than once, exhibiting consistent visual features in different parts of the scene.

From a cognitive and perceptual point of view this makes a lot of sense. In fact, it is easy to observe that if a human is presented with images representing several different but recurring objects, even in a cluttered scene, he does not need to know what the objects actually are representing in order to be able to assign semantically-consistent labels to each of them. He would even be able to label each pixel, defining the boundaries of the objects.

As an example, if someone takes a look at a large bin of different (but to some extent repeated) mechanical parts he never saw before, he is still able to tell one part from the other by exploiting their coherent visual and structural appearance. This ability is also preserved with slight changes in scale, orientation or partial occlusion of the objects.

Since this automatic assignment to a visual category of recurrent object is both well-defined and quite natural in humans, it is a very good candidate as a rule for automatically detecting relevant objects in an unsupervised manner that has good chances of being coherent with human judgement applied to the same image.



Figure 4.17: A real world example of unsupervised segmentation of a grocery shelf. Our method can automatically discover both low-level coherent patterns (brands, flavor images and logos) and high-level compound objects (multi-packs and bricks) by controlling the semantical level of the detection and segmentation process.

To be fair, we must also underline the fact that, in order to define the boundaries of a visual category and thus obtain a meaningful segmentation, also the level of detail must be taken into account. As an example, if we present to a human an image of a crowded road captured from a side, and we ask him to segment visual categories according to recurrent patterns, we could get slightly different results from different people depending on their attention to details. Some people will segment cars and trees. Other could consider the car body to be a different object from the wheels and branches from the tree trunk. The most picky could even separate tires from wheel rims and segment out each single leaf. In practice semantic consistency can happen at different scale when dealing with compound objects presenting themselves internal self repetitions or made up of single parts that are also present in other objects.

To address this aspect we also have to design a proper strategy to perform visual category detection and interpretation at a particular scale, according to the level of detail we want to express during the segmentation process. We define this level of detail as *semantical level*. Semantical levels, of course, do not map directly on specific high level concepts, such as whole objects, large parts or minute components. Rather the semantic level will act as a coarse degree of granularity of the segmentation process that will result in a hierarchical split of segments as it changes.

These two definitions of *visual categories* and *semantical levels*, that will be developed throughout the remainder of the work, are the two key concepts driving our novel segmentation method.

The ability of our approach to leverage repetitions to capture the internal representation in the real world and then extrapolates visual categories at a specific semantical level is actually achieved through the combination of a couple of standard techniques, slightly modified for the specific task, and of a few key steps specifically crafted to make the process work in a consistent way with respect to the cognitive process adopted by humans. This happens, for instance, by seeking for highly relevant repetitive structural patterns, called *semantical hotspots*, characterized by a novel feature descriptor, called *splash*. We do this through a scale-invariant method and with no continuous geometrical constraints on the visual pattern disposition.

We also do not constrain ourselves to find only one visual pattern, which is another very common assumption with other approaches in literature. Rather our technique is designed from the start to be able to detect more patterns at once, being able to assign to each of them a different visual category label, corresponding to a different real world object or object part, according to the selected semantical level.

Overall, with this study, we are offering to the community the following contributions:

- A new pipeline, including the definition of a specially crafted feature descriptor, to capture semantical categories with the ability to hierarchically span over semantical levels;
- A specially crafted conceptual framework to evaluate unsupervised semantic-driven segmentation methods through the introduction of the semantical levels notion along with a new metric;
- A new dataset consisting of a few hundredths labelled images that can be used as a benchmark for visual repetition detection in general.

The remainder of the section is organized as follows. Section 4.4 describes the related works with respect to feature extraction and automatic visual patterns detection. Section 4.4 introduces our method, giving details on the overall pipeline and on the implementation details. Section 4.4 presents an experimental evaluation and comparison with similar approaches. Finally, the conclusions are found in Section 4.4.

Code, dataset and notebooks used in this study will be made available for public use.



---

## Related Works

Several works have been proposed to tackle visual pattern discovery and detection. While the paper by Leung and Malik [Leung and Malik, 1996] could be considered seminal, many other works build on their basic approach, working by detecting contiguous structures of similar patches by knowing the window size enclosing the distinctive pattern.

One common procedure in order to describe what a pattern is, consists to first extract descriptive features such as SIFT to perform a clustering in the feature space and then model the group disposition over the image by exploiting geometrical constraints, as in [Pritts et al., 2014] and [Chum and Matas, 2010], or by relying only on appearance, as in [Doubek et al., 2010, Liu and Liu, 2013, Torii et al., 2015].

The geometrical modeling of the repetitions usually is done by fitting a planar 2-D lattice, or a deformation of it [Park et al., 2009], through RANSAC procedures as in [Schaffalitzky and Zisserman] [Pritts et al., 2014] or even by exploiting the mathematical theory of crystallographic groups as in [Liu et al., 2004]. Shechtman and Irani [Shechtman and Irani, 2007], also exploited an active learning environment to detect visual patterns in a semi-supervised fashion. For example Cheng et al. [Cheng et al., 2010] use input scribbles performed by a human to guide detection and extraction of such repeated elements, while Huberman and Fattal [Huberman and Fattal, 2016] ask the user to detect an object instance and then the detection is performed by exploiting correlation of patches near the input area.

Recently, as a result of the new wave of AI-driven Computer Vision, a number of Deep Learning based approaches emerged, in particular Lettry et al. [Lettry et al., 2017] argued that filter activation in a model such as AlexNet can be exploited in order to find regions of repeated elements over the image, thanks to the fact that filters over different layers show regularity in the activations when convolved with the repeated elements of the image. On top of the latter work, Rodríguez-Pardo et al. [Rodríguez-Pardo et al., 2019] proposed a modification to perform the texture synthesis step.

A brief survey of visual pattern discovery in both video and image data, up to 2013, is given by Wang et al. [Wang et al., 2014], unfortunately after that it seems that the computer vision community lost interest in this challenging problem. We point out that all the aforementioned methods look for *only one* particular visual repetition except for [Liu and Liu, 2013] that can be considered the most direct competitor and the main benchmark against which to compare our results.

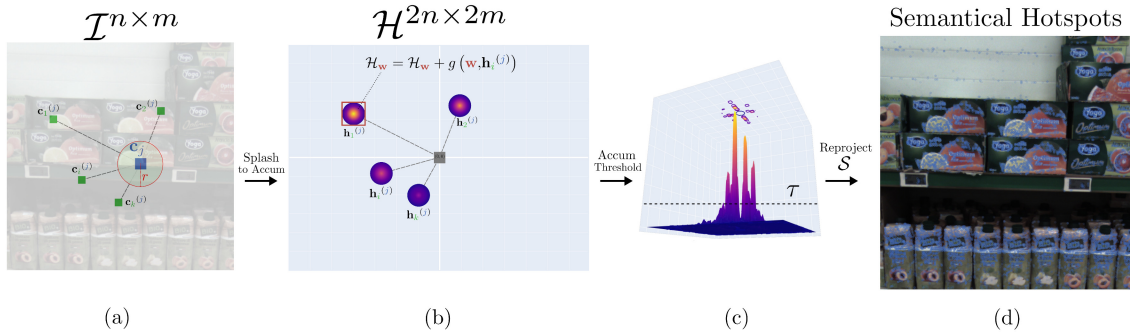


Figure 4.18: (a) A splash in the image space with center in the keypoint  $\vec{c}_j$ . (b)  $\mathcal{H}$ , with the superimposed splash at the center, you can note the different levels of the vote ordered by endpoint importance i.e. descriptor similarity. (c) 3D projection showing the gaussian-like formations and the thresholding procedure of  $\mathcal{H}$ . (d) Backprojection through the set  $\mathcal{S}$ .

## Method Description

### Features Localization and Extraction

We observe that any visual pattern is delimited by its contours. The first step of our algorithm, in fact, consists in the extraction of a set  $\mathcal{C}$  of contour *keypoints* indicating a position  $\vec{c}_j$  in the image. To extract keypoints, we opted for the Canny algorithm, for its simplicity and efficiency, although more recent and better edge extractor could be used [Liu et al., 2019] to have a better overall procedure.

A descriptor  $d_j$  is then computed for each selected  $\vec{c}_j \in \mathcal{C}$  thus obtaining a *descriptor set*  $\mathcal{D}$ . In particular, we adopted the DAISY algorithm because of its appealing dense matching properties that nicely fit our scenario. Again, here we can replace this module of the pipeline with something more advanced such as [Ono et al., 2018] at the cost of some computational time.

### Semantic Hot Spots Detection

In order to detect self-similar patterns in the image we start by associating the  $k$  most similar descriptors for each descriptor  $\vec{d}_j$ . We can visualize this data structure as a star subgraph with  $k$  endpoints called *splash* “centered” on descriptor  $\vec{d}_j$ . Figure 4.18 (a) shows one.

Splashes potentially encode repeated patterns in the image and similar patterns are then represented by similar splashes. The next step consists in separating these splashes from those that encode noise only, this is accomplished through an accumulator space.

In particular, we consider a 2-D *accumulator space*  $\mathcal{H}$  of size double the image. We then superimpose each splash on the space  $\mathcal{H}$  and cast  $k$  votes as shown in Figure 4.18 (b). In order to take into account the noise present in the splashes, we adopt a gaussian vote-casting procedure  $g(\cdot)$ . Similar superimposed splashes contribute to similar locations on the accumulator space, resulting in peak formations (Figure 4.18 (c)). We summarize the voting procedure as follows:

$$\mathcal{H}_{\vec{w}} = \mathcal{H}_{\vec{w}} + g(\vec{w}, \vec{h}_i^{(j)}) \quad (4.14)$$

where  $\vec{h}_i^{(j)}$  is the  $i$ -th splash endpoint of descriptor  $\vec{d}_j$  in accumulator coordinates and  $\vec{w}$  is the size of the gaussian vote. We filter all the regions in  $\mathcal{H}$  which are above a certain *threshold*  $\tau$ , to get a set  $\mathcal{S}$  of the locations corresponding to the peaks in  $\mathcal{H}$ . The  $\tau$  parameter acts as a coarse filter and is not a critical parameter to the overall pipeline. A sufficient value is to set it to  $0.05 \cdot \max(\mathcal{H})$ . Lastly, in order to visualize the semantic hotspots in the image plane we map splash locations between  $\mathcal{H}$  and the image plane by means of a *backtracking structure*  $\mathcal{V}$ .

In summary, the key insight here is that similar visual regions share similar splashes, we discern noisy splashes from representative splashes through an auxiliary structure, namely an accumulator. We then identify and backtrack in the image plane the semantic hotspots that are candidate points part of a visual repetition.

## Semantic Categories Definition and Extraction

While the first part previously described acts as a filter for noisy keypoints allowing to obtain a good pool of candidates, we now transform the problem of finding visual categories in a problem of dense subgraphs extraction.

We enclose semantic hotspots in superpixels, this extends the semantic significance of such identified points to a broader, but coherent, area. To do so we use the SLIC [Achanta et al., 2012] algorithm which is a simple and one of the fastest approaches to extract superpixels as pointed out in this recent survey [Stutz et al., 2018]. Then we choose the cardinality of the *superpixels*  $\mathcal{P}$  to extract. This is the

**Algorithm 2** Semantic categories extraction algorithm**Require:**  $G$  weighted undirected graph $i = 0$  $s^* = -\text{inf}$  $K^* = \emptyset$ **while**  $G_i$  is not fully disconnected **do** $i = i + 1$ Compute  $G_i$  by corroding each edge with the minimum edge weightExtract the set  $K_i$  of all connected components in  $G_i$  $s(G_i, K_i) = \sum_{k \in K_i} \mu(k) - \alpha |K_i|$ **if**  $s(G_i, K_i) > s^*$  **then** $s^* = s(G_i, K_i)$  $K^* = K_i$ **return**  $s^*, K^*$ 

second and most fundamental parameter that will allow us to span over different semantic levels.

Once the superpixels have been extracted, let  $\mathcal{G}$  be an *undirected weighted graph* where each node correspond to a superpixel  $p \in \mathcal{P}$ . In order to put edges between graph nodes (i.e. two superpixels), we exploit the splashes origin and endpoints. In particular the strength of the connection between two vertices in  $\mathcal{G}$  is calculated with the number of splashes endpoints falling between the two in a mutual coherent way. So to put a weight of 1 between two nodes we need exactly 2 splashes endpoints falling with both origin and end point in the two candidate superpixels.

With this construction scheme, the graph has clear dense subgraphs formations. Therefore, the last part simply computes a partition of  $\mathcal{G}$  where each connected component correspond to a cluster of similar superpixels. In order to achieve such objective we optimize a function that is maximized when we partition the graph to represent so. To this end we define the following *density score* that given  $G$  and a set  $K$  of connected components captures the optimality of the clustering:

$$s(G, K) = \sum_{k \in K} \mu(k) - \alpha |K| \quad (4.15)$$

where  $\mu(k)$  is a function that computes the average edge weight in a undirected weighted graph.

The first term, in the score function, assign a high vote if each connected compo-

ment is dense. While the second term acts as a regulator for the number of connected components. We also added a weighting factor  $\alpha$  to better adjust the procedure. As a proxy to maximize this function we devised an *iterative algorithm* reported in Algorithm 2 based on graph corrosion and with temporal complexity of  $O(|E|^2 + |E||V|)$ . At each step the procedure corrupts the graph edges by the minimum edge weight of  $G$ . For each corroded version of the graph that we call *partition*, we compute  $s$  to capture the density. Finally the algorithm selects the corroded graph partition which maximizes the  $s$  and subsequently extracts the node groups.

In brevity we first enclose semantic hotspots in superpixels and consider each one as a node of a weighted graph. We then put edges with weight proportional to the number of splashes falling between two superpixels. This results in a graph with clear dense subgraphs formations that correspond to superpixels clusters i.e. *semantic categories*. The semantic categories detection translates in the extraction of dense subgraphs. To this end we devised an iterative algorithm based on graph corrosion where we let the procedure select the corroded graph partition that filters noisy edges and let dense subgraphs emerge. We do so by maximizing score that captures the density of each connected component.

## Experiments

### Dataset

As we introduced in Section 4.4 one of the aims of this work is to provide a better comparative framework for visual pattern detection. To do so we created a public dataset by taking 104 pictures of store shelves. Each picture has been took with a 5mpx camera with approximatively the same visual conditions. We also rectified the images to eliminate visual distortions.

We manually segmented and labeled each repeating product in two different semantic levels. In the **first semantic level** *products made by the same company* share the same label. In the **second semantic level** visual repetitions consist in the *exact identical products*. In total the dataset is composed by 208 ground truth images, half in the first level and the rest for the second one.

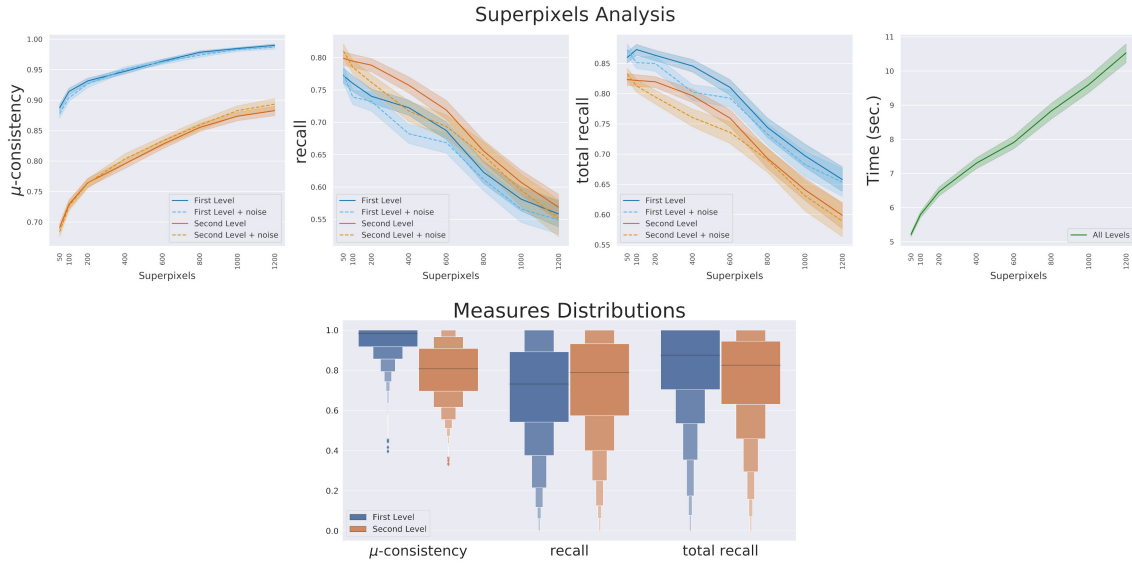


Figure 4.19: (top) Analysis of measures as the number of superpixels  $|\mathcal{P}|$  retrieved varies. The rightmost figure shows the running time of the algorithm. We repeated the experiments with the noisy version of the dataset but report only the mean since variation is almost equal to the original one. (bottom) Distributions of the measures for the two semantic levels, by varying the two main parameters  $r$  and  $|\mathcal{P}|$ .

## $\mu$ -consistency

We devised a new measure that captures the semantic consistency of a detected pattern that is a proxy of the average precision of detection.

In fact, we want to be sure that all pattern instances fall on similar ground truth objects. First we introduce the concept of semantic consistency for a particular pattern  $\vec{p}$ . Let  $\vec{P}$  be the set of patterns discovered by the algorithm. Each pattern  $\vec{p}$  contains several instances  $\vec{p}_i$ .  $\vec{L}$  is the set of ground truth categories, each ground truth category  $\vec{l}$  contain several objects instances  $\vec{l}_j$ . Let us define  $\vec{t}_p$  as the vector of ground truth labels touched by all instances of  $\vec{p}$ . We say that  $\vec{p}$  is consistent if all its instances  $\vec{p}_i, i = 0 \dots |\vec{p}|$  fall on ground truth regions sharing the same label. In this case  $\vec{t}_p$  would be uniform and we consider  $\vec{p}$  a good detection. The worst scenario is when given a pattern  $\vec{p}$  every  $\vec{p}_i$  falls on objects with different label  $\vec{l}$  i.e. all the values in  $\vec{t}_p$  are different.

To get an estimate of the overall consistency of the proposed detection, we average the consistency for each  $\vec{p} \in \vec{P}$  giving us:

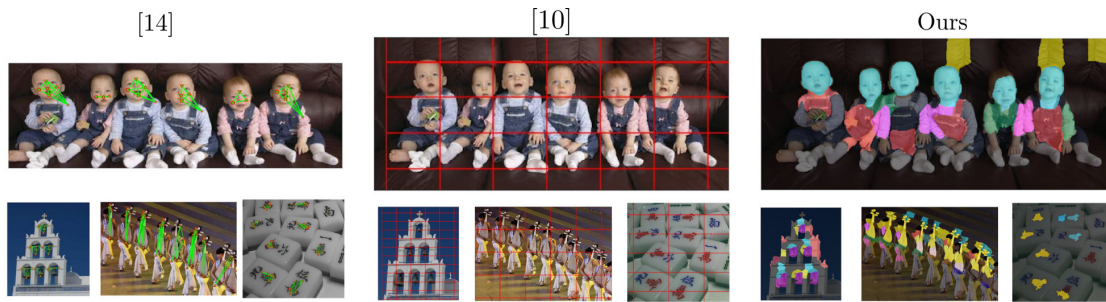


Figure 4.20: Qualitative comparison between [Liu and Liu, 2013] [14], [Lettry et al., 2017] [10] and our algorithm. Our method detects and segments more than one pattern and does not constrain itself to a particular geometrical disposition.

$$\mu\text{-consistency} = \frac{1}{|\bar{\mathcal{P}}|} \sum_{\bar{p} \in \bar{\mathcal{P}}} \frac{|\text{mode}(\vec{t}_p)|}{|\vec{t}_p|} \quad (4.16)$$

## Recall

The second measure is the classical recall over the objects retrieved by the algorithm. Since our object detector outputs more than one pattern we average the recall for each ground truth label by taking the best fitting pattern.

$$\frac{1}{|\bar{\mathcal{L}}|} \sum_{\vec{l} \in \bar{\mathcal{L}}} \max_{\bar{p} \in \bar{\mathcal{P}}} \text{recall}(\bar{p}, \vec{l}) \quad (4.17)$$

The last measure is the **total recall**, here we consider a hit if any of the pattern falls in a labeled region. In general we expect this to be higher than the recall.

We report the summary performances in Figure 4.20. As can be seen the algorithm achieves a very high  $\mu$ -consistency while still able to retrieve the majority of the ground truth patterns in both levels.

One can observe in Figure 4.19 an inverse behaviour between recall and consistency as the number of superpixels retrieved grows. This is expected since less superpixels means bigger patterns, therefore it is more likely to retrieve more ground truth patterns.

In order to study the robustness we repeated the same experiments with an altered version of our dataset. In particular for each image we applied one of the following corruptions: Additive Gaussian Noise ( $scale = 0.1 * 255$ ), Gaussian Blur ( $\sigma = 3$ ), Spline Distortions (grid affine), Brightness (+100), and Linear Contrast (1.5).

## Qualitative Validation

Firstly we begin the comparison by commenting on [Liu and Liu, 2013]. One can observe that our approach has a significant advantage in terms of how the visual pattern is modeled. While the authors model visual repetitions as geometrical artifacts associating points, we output a higher order representation of the visual pattern. Indeed the capability to provide a segmentation mask of the repeated instance region together the ability to span over different levels unlocks a wider range of use cases and applications.

As qualitative comparison we also added the latest (and only) deep learning based methodology [Lettry et al., 2017] we found. This methodology is only able to find a single instance of visual pattern, namely the most frequent and most significant with respect to the filters weights. This means that the detection strongly depends from the training set of the CNN backbone, while our algorithm is fully unsupervised and data agnostic.

## Quantitative Validation

We compared quantitatively our method against [Liu and Liu, 2013] that constitutes, to the best of our knowledge, the only work developed able to detect more than one visual pattern. We recreated the experimental settings of the authors by using the Face dataset [Li et al., 2007] as benchmark achieving 1.00 precision vs. 0.98 of [Liu and Liu, 2013] and 0.77 in recall vs. and 0.63. We considered a miss on the object retrieval task, if more than 20% of a pattern total area falls outside from the ground truth. The parameter used were  $|\mathcal{C}| = 9000$ ,  $k = 15$ ,  $r = 30$ ,  $\tau = 5$ ,  $|\mathcal{P}| = 150$ . We also fixed the window of the gaussian vote to be  $11 \times 11$  pixels throughout all the experiments.



## Conclusions

With this study we introduced a fast and unsupervised method addressing the problem of finding semantic categories by detecting consistent visual pattern repetitions at a given scale. The proposed pipeline hierarchically detects self-similar regions represented by a segmentation mask.

As we demonstrated in the experimental evaluation, our approach retrieves more than one pattern and achieves better performances with respect to competitors methods. We also introduce the concept of *semantic levels* endowed with a dedicated dataset and a new metric to provide to other researchers tools to evaluate the consistency of their approaches.

## Acknowledgments

We would like to express our gratitude to Alessandro Torcinovich and Filippo Bergamasco for their suggestions to improve the work. We also thank Mattia Mantoan for his work to produce the dataset labeling.



# **Chapter 5**

## **Conclusions**

In this thesis, we contributed spanned the dissection of continual learning by providing several structural and data analyses. First we provide a gentle introduction to the topic of continual learning starting by highlighting the difference between natural and artificial models. Among the differences we stress the importance of time, which is an essential component for developing lifelong learning machines. Then, we informally introduce the main challenges that continual learning systems must tackle. In particular, catastrophic forgetting and the stability plasticity dilemma. To better provide an intuition about these topics, we provided a visual example of catastrophic forgetting in an autoencoder model, showing how distributional shifts in the subsequent tasks result in the abrupt damage of past knowledge.

Later, we move on by giving a more formal definition of continual learning settings prominently adopted in literature. We introduced the notions of class-incremental, task-incremental, online/offline learning along with a specification on other common settings in the field. Before moving on the contributions we provided a small literature review on the state-of-the-art by describing the main categories under which continual learning methods have been grouped.

Finally, we move on the main contributions. First, we introduced a study on the quality/quantity trade-off in rehearsal-based continual learning. Here, we selected one of the most performant baselines, that is GDumb, and analyzed several compression techniques when applied to the replay buffer. We highlighted that the quantity of data is a far more important factor when storing exemplars in the replay buffer. We do so by considering different compression schemes with extreme rates. Then, we moved into the second major contribution which considers Visual Transformers in an incremental setting. Here, besides being one of the first works on visual transformers for continual learning, we provided a surgical investigation on regularization methods for ViTs in the challenging setting of rehearsal-free CL. We compared functional, weight and attentional regularizations, with the latter being a regularization in the matrix of the self-attention mechanism. Attentional regularizations provide comparable performance with respect to the other methods. As second contribution we also introduced a loss inspired by a method nowadays in vogue (PODNet) and devised an asymmetric variant. We show that the introduction of the asymmetric variant allows achieving more plasticity to the model when applied to different part of the mechanism of self-attention. Then, we proposed a study on off-the-shelf continual learning exploiting fully pretrained networks and, in particular, we proposed a simple baseline. The baseline is composed by a feature extractor and a knn-like prototype memory. The baseline is crafted to be performant in practical scenarios achieving optimal results with a memory overhead of few KBytes. Moreover we discussed its possible extension to the realm of unsupervised continual learning. We then linked this preliminary discussion with the exploration of visual categories.

To do so we introduce another work tackling unsupervised pattern discovery. In fact, the notion of *discovery* is naturally included into the notion of lifelong learning: an agent capable of lifelong learning, surely should possess the ability to autonomously discover new knowledge. We do so by introducing a new unsupervised algorithm to perform unsupervised semantic segmentation at different semantic scales.

## Further Developings

With the several studies proposed, we want to highlight the directions where it might be more fruitful to investigate further to build better Continual Learning agents.

*A first warning we raised regards the dataset usage to assess the performance of CL algorithms.* In particular, with Section 4.1 we see that extreme levels of buffer data resize still provide good results in rehearsal systems, suggesting that, perhaps, more realistic datasets should be included to devise more useful solutions. This finding is also supported by Section 4.3 which shows that tackling these benchmarks with a pretrained backbone is sufficient to overcome quasi-optimally continual learning scenarios on 5 different datasets. This also suggests that pretraining could be a great advantage, in the generalization ability of the model, when building new CL algorithms.

To tackle the aforementioned point, the community can focus more on unsupervised continual learning which is a natural and more challenging problem extension. While keeping the same datasets we can now also leverage pretrained backbones. While being appealing on its own, following this line is also greatly encouraged by the fact that there are virtually no works on such a topic.

With the study proposed in Section 4.2 we show that ViTs are naturally inclined continual learners. We suspect that the less inductive bias carried by such models might be the key that allows such models to perform better in incremental scenarios. On another side, we see that the results obtained without pretraining have difficulty achieving CNN performances so easily (we can compare the results of Section 4.1 and Section 4.2). This calls for the need to build less data-hungry models in line with the world's fast-paced data generation. Within Section 4.2 we also propose a new way to assess Continual Learning methods. We think that *the community still lacks of a principled way to measure the stability-plasticity trade-off*. With our introduction of the two curves, we proposed an initial tentative to monitor the performance of a system.

Last but not least, with the work of Section 4.4 we stress that autonomously discovering new patterns should be a core ability of an intelligent system. In fact, if an agent can explore the real world and find hierarchies of knowledge without help, all it has to do to incrementally learn is to store such knowledge in some kind of long-term memory repository which translates into a *compression problem*.

# Bibliography

- Networks of spiking neurons: The third generation of neural network models. *Neural Networks*, 1997.
- Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurélien Lucchi, Pascal Fua, and Sabine Süsstrunk. SLIC superpixels compared to state-of-the-art superpixel methods. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2012.
- Tameem Adel, Han Zhao, and Richard E. Turner. Continual learning with adaptive weights (CLAW). In *International Conference on Learning Representations (ICLR)*, 2020.
- Abien Fred Agarap. Deep learning using rectified linear units (relu). *ArXiv*, 2018.
- Rishabh Agarwal, Dale Schuurmans, and Mohammad Norouzi. An optimistic perspective on offline reinforcement learning. In *ICML*, 2020.
- Hongjoon Ahn, Jihwan Kwak, Su Fang Lim, Hyeonsu Bang, Hyojun Kim, and Taesup Moon. Ss-il: Separated softmax for incremental learning. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 824–833, 2021.
- Rahaf Aljundi, Punarjay Chakravarty, and Tinne Tuytelaars. Expert gate: Lifelong learning with a network of experts. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.
- Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.
- Rahaf Aljundi, Eugene Belilovsky, Tinne Tuytelaars, Laurent Charlin, Massimo Caccia, Min Lin, and Lucas Page-Caccia. Online continual learning with maximal interfered retrieval. In *Advances in Neural Information Processing Systems (NeurIPS)*. Curran Associates, Inc., 2019a.
- Rahaf Aljundi, Klaas Kelchtermans, and Tinne Tuytelaars. Task-free continual learning. In *CVPR*. Computer Vision Foundation / IEEE, 2019b.

- Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online 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 (NeurIPS)*, 2019c.
- Elahe Arani, Fahad Sarfraz, and Bahram Zonooz. Learning fast, learning slow: A general continual learning method based on complementary learning system. *ICLR*, 2022.
- S. Banerjee, Vinay Kumar Verma, Toufiq Parag, Maneesh Kumar Singh, and Vinay P. Namboodiri. Class incremental online streaming learning. *NeurIPS*, 2021.
- Eden Belouadah, Adrian Popescu, and Ioannis Kanellos. A comprehensive study of class incremental learning algorithms for visual tasks. *Neural Networks*, 2021.
- Edward L. Bennett, Marian Cleeves Diamond, David Krech, and Mark Richard Rosenzweig. Chemical and anatomical plasticity of brain changes in brain through experience, demanded by learning theories, are found in experiments with rats. *Science*, 1964.
- Alessandro Betti, Marco Gori, Simone Marullo, and Stefano Melacci. Developing constrained neural units over time. In *International Joint Conference on Neural Networks, IJCNN*. IEEE, 2020.
- Magdalena Biesialska, Katarzyna Biesialska, and Marta R. Costa-jussà. Continual lifelong learning in natural language processing: A survey. In *Proceedings of the 28th International Conference on Computational Linguistics*. International Committee on Computational Linguistics, 2020.
- 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.
- Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark experience for general continual learning: a strong, simple baseline. 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.
- Gail A. Carpenter and Stephen Grossberg. The art of adaptive pattern recognition by a self-organizing neural network. *Computer*, 1988.
- Fabio Cermelli, Massimiliano Mancini, Samuel Rota Buló, Elisa Ricci, and Barbara Caputo. Modeling the background for incremental learning in semantic segmentation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.
- Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.
- Arslan Chaudhry, Marc’Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with A-GEM. In *International Conference on Learning Representations, (ICLR)*, 2019a.
- Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K. Dokania, Philip H. S. Torr, and Marc’Aurelio Ranzato. On tiny episodic memories in continual learning, 2019b.
- Hui Chen, Chao Tan, and Zan Lin. Ensemble of extreme learning machines for multivariate calibration of near-infrared spectroscopy. *Spectrochimica Acta Part A: Molecular and Biomolecular Spectroscopy*, 2020.
- Xiangning Chen, Cho-Jui Hsieh, and Boqing Gong. When vision transformers outperform resnets without pretraining or strong data augmentations. *ICLR*, 2022.
- Ming-Ming Cheng, Fang-Lue Zhang, Niloy J. Mitra, Xiaolei Huang, and Shi-Min Hu. Repfinder: finding approximately repeated scene elements for image editing. *ACM Trans. Graph.*, 2010.
- Riccardo Del Chiaro, Bartłomiej Twardowski, Andrew D. Bagdanov, and Joost van de Weijer. RATT: recurrent attention to transient tasks for continual image captioning. 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.
- Ondrej Chum and Jiri Matas. Unsupervised discovery of co-occurrence in sparse high dimensional data. In *The Twenty-Third IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2010*, 2010.

- Tarin Clanuwat, Mikel Bober-Irizar, A. Kitamoto, Alex Lamb, Kazuaki Yamamoto, and David Ha. Deep learning for classical japanese literature. *ArXiv*, 2018.
- Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). In Yoshua Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR*, 2016.
- Cyprien de Masson d’Autume, Sebastian Ruder, Lingpeng Kong, and Dani Yogatama. Episodic memory in lifelong language 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 (NeurIPS)*, 2019.
- Matthias Delange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Greg Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.
- Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. Imagenet: A large-scale hierarchical image database. In *Society Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2009.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT*. Association for Computational Linguistics, 2019.
- Prithviraj Dhar, Rajat Vikram Singh, Kuan-Chuan Peng, Ziyang Wu, and Rama Chellappa. Learning without memorizing. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR*, 2021.
- Briar Doty, Stefan Mihalas, Anton Arkhipov, and Alex T. Piet. Heterogeneous ‘cell types’ can improve performance of deep neural networks. *bioRxiv*, 2021.

- Petr Douthek, Jiri Matas, Michal Perdoch, and Ondrej Chum. Image matching and retrieval by repetitive patterns. In *20th International Conference on Pattern Recognition, ICPR 2010*, 2010.
- Arthur Douillard and Timothée Lesort. Continuum: Simple management of complex continual learning scenarios, 2021.
- Arthur Douillard, Matthieu Cord, Charles Ollion, Thomas Robert, and Eduardo Valle. Podnet: Pooled outputs distillation for small-tasks incremental learning. In *Proceedings of the IEEE European Conference on Computer Vision (ECCV)*, 2020.
- Arthur Douillard, Alexandre Ramé, Guillaume Couairon, and Matthieu Cord. Dytox: Transformers for continual learning with dynamic token expansion. *CoRR*, abs/2111.11326, 2021.
- Sayna Ebrahimi, Mohamed Elhoseiny, Trevor Darrell, and Marcus Rohrbach. Uncertainty-guided continual learning with bayesian neural networks. In *International Conference on Learning Representations, (ICLR)*, 2020.
- Spyros Gidaris and Nikos Komodakis. Dynamic few-shot visual learning without forgetting. In *Conference on Computer Vision and Pattern Recognition, (CVPR)*. IEEE, 2018.
- Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*, 2014.
- Pankaj Gupta, Yatin Chaudhary, Thomas A. Runkler, and Hinrich Schütze. Neural topic modeling with continual lifelong learning. In *International Conference on Machine Learning, (ICML)*. PMLR, 2020.
- Raia Hadsell, D. Rao, Andrei A. Rusu, and Razvan Pascanu. Embracing change: Continual learning in deep neural networks. *Trends in Cognitive Sciences*, 2020.
- Tyler L. Hayes and Christopher Kanan. Lifelong machine learning with deep streaming linear discriminant analysis. In *CVPR*, 2020.
- Tyler L. Hayes, Nathan D. Cahill, and Christopher Kanan. Memory efficient experience replay for streaming learning. In *International Conference on Robotics and Automation, ICRA 2019, Montreal, QC, Canada, May 20-24, 2019*. IEEE, 2019.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition, (CVPR)*. IEEE, 2016.

- Felix Hill, Adam Santoro, David G. T. Barrett, Ari S. Morcos, and Timothy P. Lillicrap. Learning to make analogies by contrasting abstract relational structure. In *7th International Conference on Learning Representations, ICLR*. OpenReview.net, 2019.
- Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *ArXiv*, 2015.
- Sepp Hochreiter. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 1998.
- Guillaume Hocquet, Olivier Bichler, and Damien Querlioz. Ova-inn: Continual learning with invertible neural networks. In *2020 International Joint Conference on Neural Networks (IJCNN)*, 2020.
- Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.
- Yen-Chang Hsu, Y. Liu, and Z. Kira. Re-evaluating continual learning scenarios: A categorization and case for strong baselines. *ArXiv*, abs/1810.12488, 2018.
- Guang-Bin Huang, Lei Chen, and Chee Kheong Siew. Universal approximation using incremental constructive feedforward networks with random hidden nodes. *IEEE Trans. Neural Networks*, 2006.
- Inbar Huberman and Raanan Fattal. Detecting repeating objects using patch correlation analysis. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 2016.
- Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Francis R. Bach and David M. Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning, ICML, JMLR Workshop and Conference Proceedings*. JMLR.org, 2015.
- Khurram Javed and Faisal Shafait. Revisiting distillation and incremental classifier learning. In *ACCV*, 2018.
- W. Johnson. Extensions of lipschitz mappings into hilbert space. *Contemporary mathematics*, 1984.
- K. J. Joseph, Jathushan Rajasegaran, Salman Hameed Khan, Fahad Shahbaz Khan, Vineeth N. Balasubramanian, and Ling Shao. Incremental object detection via meta-learning. *IEEE transactions on pattern analysis and machine intelligence TPAMI*, 2021.

- Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In Yoshua Bengio and Yann LeCun, editors, *2nd International Conference on Learning Representations, (ICLR)*, 2014.
- Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017*. OpenReview.net, 2017.
- 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. *Proceedings of the National Academy of Sciences*, 2017.
- Jeremias Knoblauch, Hisham Husain, and Tom Diethe. Optimal continual learning has perfect memory and is np-hard. In *ICML*, 2020.
- Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.
- Ramesh Kumar Lama, Jeonghwan Gwak, Jeong-Seon Park, and Sang-Woong Lee. Diagnosis of alzheimer’s disease based on structural mri images using a regularized extreme learning machine and pca features. *Journal of healthcare engineering*, 2017.
- Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998.
- Soochan Lee, Junsoo Ha, Dongsu Zhang, and Gunhee Kim. A neural dirichlet process mixture model for task-free continual learning. In *8th International Conference on Learning Representations, ICLR*, 2020.
- Louis Lettry, Michal Perdoch, Kenneth Vanhoey, and Luc Van Gool. Repeated pattern detection using CNN activations. In *2017 IEEE Winter Conference on Applications of Computer Vision, WACV 2017*, 2017.
- Thomas K. Leung and Jitendra Malik. Detecting, localizing and grouping repeated scene elements from an image. In *Computer Vision - ECCV’96, 4th European Conference on Computer Vision, Proceedings, Volume I*. Springer, 1996.
- Duo Li, Guimei Cao, Yunlu Xu, Zhanzhan Cheng, and Yi Niu. Technical report for iccv 2021 challenge sslad-track3b: Transformers are better continual learners. *arXiv preprint arXiv:2201.04924*, 2022.

- Fei-Fei Li, Robert Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Comput. Vis. Image Underst.*, 2007.
- Zhizhong Li and Derek Hoiem. Learning without forgetting. *transactions on pattern analysis and machine intelligence (PAMI)*, 2017.
- Seppo Linnainmaa. Taylor expansion of the accumulated rounding error. *BIT Numerical Mathematics*, 1976.
- Jingchen Liu and Yanxi Liu. GRASP recurring patterns from a single view. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2013.
- Xialei Liu, Chenshen Wu, Mikel Menta, Luis Herranz, Bogdan Raducanu, Andrew D Bagdanov, Shangling Jui, and Joost van de Weijer. Generative feature replay for class-incremental learning. In *Conference on Computer Vision and Pattern Recognition Workshops CVPRW*, 2020a.
- Yanxi Liu, Robert T. Collins, and Yanghai Tsin. A computational model for periodic pattern perception based on frieze and wallpaper groups. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2004.
- Yaoyao Liu, Anan Liu, Yuting Su, Bernt Schiele, and Qianru Sun. Mnemonics training: Multi-class incremental learning without forgetting. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020b.
- Yun Liu, Ming-Ming Cheng, Xiaowei Hu, Jia-Wang Bian, Le Zhang, Xiang Bai, and Jinhui Tang. Richer convolutional features for edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.
- Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *Proc. ICCV*, 2021.
- Vincenzo Lomonaco and Davide Maltoni. Core50: a new dataset and benchmark for continuous object recognition. In Sergey Levine, Vincent Vanhoucke, and Ken Goldberg, editors, *Proceedings of the 1st Annual Conference on Robot Learning*, Proceedings of Machine Learning Research. PMLR, 2017.
- Vincenzo Lomonaco, Karan Desai, Eugenio Culurciello, and Davide Maltoni. Continual reinforcement learning in 3d non-stationary environments. In *Conference on Computer Vision and Pattern Recognition, (CVPR)*. IEEE, 2020.
- Vincenzo Lomonaco, Lorenzo Pellegrini, Pau Rodríguez López, Massimo Caccia, Qi She, Yu Chen, Quentin Jodelet, Ruiping Wang, Zheda Mai, David Vázquez,

- German Ignacio Parisi, Nikhil Churamani, Marc Pickett, Issam H. Laradji, and Davide Maltoni. Cvpr 2020 continual learning in computer vision competition: Approaches, results, current challenges and future directions. *Artif. Intell.*, 2022.
- David Lopez-Paz and Marc’Aurelio Ranzato. Gradient episodic memory for continual learning. 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 (NeurIPS)*, 2017.
- Ilya Loshchilov and Frank Hutter. SGDR: stochastic gradient descent with warm restarts. In *International Conference on Learning Representations, (ICLR)*, 2017.
- Andrew L. Maas, Awni Y. Hannun, and Andrew Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *in ICML Workshop on Deep Learning for Audio, Speech and Language Processing*, 2013.
- Zheda Mai, Ruiwen Li, Jihwan Jeong, David Quispe, Hyunwoo J. Kim, and Scott Sanner. Online continual learning in image classification: An empirical survey. *Neurocomputing*, 2022.
- 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*. Computer Vision Foundation / IEEE Computer Society, 2018.
- David McCaffary. Towards continual task learning in artificial neural networks: current approaches and insights from neuroscience. *ArXiv*, 2021.
- James L. McClelland, Bruce L. McNaughton, and Randall C. O’Reilly. Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory. *Psychological review*, 1995.
- Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*. Elsevier, 1989.
- Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 1943.
- Sanket Vaibhav Mehta, Darshan Patil, Sarath Chandar, and Emma Strubell. An empirical investigation of the role of pre-training in lifelong learning. *ArXiv*, abs/2112.09153, 2021.

- Umberto Michieli and Pietro Zanuttigh. Incremental learning techniques for semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops (ICCV-W)*, 2019.
- Seyed Iman Mirzadeh, Arslan Chaudhry, Dong Yin, Timothy Nguyen, Razvan Pascanu, Dilan Gorur, and Mehrdad Farajtabar. Architecture matters in continual learning. *arXiv preprint arXiv:2202.00275*, 2022.
- Katelyn Morrison, Benjamin Gilby, Colton Lipchak, Adam Mattioli, and Adriana Kovashka. Exploring corruption robustness: Inductive biases in vision transformers and mlp-mixers. *CoRR*, abs/2106.13122, 2021.
- Martin Mundt, Yong Won Hong, Iuliia Pliushch, and Visvanathan Ramesh. A wholistic view of continual learning with deep neural networks: Forgotten lessons and the bridge to active and open world learning. *ArXiv*, abs/2009.01797, 2020.
- Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, 2010.
- Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. Feature visualization. *Distill*, 2017. doi: 10.23915/distill.00007. <https://distill.pub/2017/feature-visualization>.
- Yuki Ono, Eduard Trulls, Pascal Fua, and Kwang Moo Yi. Lf-net: Learning local features from images. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 31, NeurIPS*, 2018.
- German Ignacio Parisi, Ronald Kemker, Jose L. Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 2019.
- Minwoo Park, Kyle Brocklehurst, Robert T. Collins, and Yanxi Liu. Deformed lattice detection in real-world images using mean-shift belief propagation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2009.
- Sayak Paul and Pin-Yu Chen. Vision transformers are robust learners. *arXiv preprint arXiv:2105.07581*, 2021.
- Jary Pomponi, Simone Scardapane, Vincenzo Lomonaco, and Aurelio Uncini. Efficient continual learning in neural networks with embedding regularization. *Neurocomputing*, 2020.
- Mozhgan Pourkeshavarz and M. Sabokrou. Zs-il: Looking back on learned experiences for zero-shot incremental learning. *ICLR*, 2022.



- Ameya Prabhu, Philip H. S. Torr, and Puneet K. Dokania. Gdumb: A simple approach that questions our progress in continual learning. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision - (ECCV)*, Lecture Notes in Computer Science. Springer, 2020.
- James Pritts, Ondrej Chum, and Jiri Matas. Rectification, and segmentation of coplanar repeated patterns. In *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR, 2014*.
- Muhammad Naveed Iqbal Qureshi, Beomjun Min, Hang Joon Jo, and Boreom Lee. Multiclass classification for the differential diagnosis on the adhd subtypes using recursive feature elimination and hierarchical extreme learning machine: structural mri study. *PloS one*, 2016.
- Maithra Raghu, Thomas Unterthiner, Simon Kornblith, Chiyuan Zhang, and Alexey Dosovitskiy. Do vision transformers see like convolutional neural networks? *ArXiv*, 2021.
- Jathushan Rajasegaran, Munawar Hayat, Salman H. Khan, Fahad Shahbaz Khan, and Ling Shao. Random path selection 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*.
- Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. icarl: Incremental classifier and representation learning. In *Conference on Computer Vision and Pattern Recognition, (CVPR)*. IEEE, 2017.
- Maximilian Riesenhuber and Tomaso A. Poggio. Hierarchical models of object recognition in cortex. *Nature Neuroscience*, 1999.
- Mark Bishop Ring et al. *Continual learning in reinforcement environments*. PhD thesis, 1994.
- Hippolyt Ritter, Aleksandar Botev, and David Barber. Online structured laplace approximations for overcoming catastrophic forgetting. In *Advances in Neural Information Processing Systems*, 2018.
- Carlos Rodríguez-Pardo, Sergio Suja, David Pascual, Jorge Lopez-Moreno, and Elena Garces. Automatic extraction and synthesis of regular repeatable patterns. *Comput. Graph.*, 2019.

- David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy P. Lillicrap, and Gregory Wayne. Experience replay 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.
- Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 1958.
- Aurko Roy, Mohammad Saffar, Ashish Vaswani, and David Grangier. Efficient content-based sparse attention with routing transformers. *Trans. Assoc. Comput. Linguistics*, 2021.
- David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 1986.
- Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *Int. J. Comput. Vis.*, 2015.
- Andrei A. Rusu, Neil C. Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *ArXiv*, 2016.
- Sara Sabour, Nicholas Frosst, and Geoffrey E. Hinton. Dynamic routing between capsules. 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 NIPS*, 2017.
- Frederik Schaffalitzky and Andrew Zisserman. Geometric grouping of repeated elements within images. In John N. Carter and Mark S. Nixon, editors, *Proceedings of the British Machine Vision Conference 1998, BMVC*.
- Jürgen Schmidhuber. Deep learning in neural networks: An overview. *CoRR*, abs/1404.7828, 2014.
- Eli Shechtman and Michal Irani. Matching local self-similarities across images and videos. In *2007 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2007)*. IEEE Computer Society, 2007.
- Dongsub Shim, Zheda Mai, Jihwan Jeong, Scott Sanner, Hyunwoo Kim, and Jongseong Jang. Online class-incremental continual learning with adversarial shapley value. *AAAI*, 2021.

- Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. 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 NIPS, 2017*.
- Shagun Sodhani, Sarath Chandar, and Yoshua Bengio. Toward training recurrent neural networks for lifelong learning. *Neural Computation*, 2020.
- David Stutz, Alexander Hermans, and Bastian Leibe. Superpixels: An evaluation of the state-of-the-art. *Comput. Vis. Image Underst.*, 2018.
- Fan-Keng Sun, Cheng-Hao Ho, and Hung-Yi Lee. LAMOL: language modeling for lifelong language learning. In *International Conference on Learning Representations, (ICLR)*, 2020.
- Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 2015.
- Chuanqi Tan, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu. A survey on deep transfer learning. In Vera Kurková, Yannis Manolopoulos, Barbara Hammer, Lazaros S. Iliadis, and Ilias Maglogiannis, editors, *International Conference on Artificial Neural Networks (ICANN)*, Lecture Notes in Computer Science. Springer, 2018.
- Bosiljka Tasic, Zizhen Yao, Lucas T. Graybuck, Kimberly A. Smith, Thuc Nghi Nguyen, Darren Bertagnolli, Jeff Goldy, Emma Garren, Michael N. Economo, Sarada Viswanathan, Osnat Penn, Trygve E Bakken, Vilas Menon, Jeremy A. Miller, Olivia Fong, Karla E. Hirokawa, Kanan Lathia, Christine Rimorin, Michael Tieu, Rachael Larsen, Tamara Casper, Eliza Barkan, Matthew Kroll, Sheana E. Parry, Nadiya Shapovalova, Daniel Hirschstein, Julie Pendergraft, Heather Anne Sullivan, Tae Kyung Kim, Aaron Szafer, Nick Dee, Peter A. Groblewski, Ian R. Wickersham, Ali H. Cetin, Julie A. Harris, Boaz P. Levi, Susan M. Sunkin, Linda J. Madisen, Tanya L. Daigle, Loren L. Looger, Amy Bernard, John W. Phillips, Ed S. Lein, Michael J. Hawrylycz, Karel Svoboda, Allan R. Jones, Christof Koch, and Hongkui Zeng. Shared and distinct transcriptomic cell types across neocortical areas. *Nature*, 2018.
- Sebastian Thrun. *Explanation-based neural network learning a lifelong learning approach*. PhD thesis, University of Bonn, Germany, 1995a.

- Sebastian Thrun. Is learning the  $n$ -th thing any easier than learning the first? In David S. Touretzky, Michael Mozer, and Michael E. Hasselmo, editors, *Advances in Neural Information Processing Systems, (NIPS)*. MIT Press, 1995b.
- Sebastian Thrun and Tom M. Mitchell. Lifelong robot learning. *Robotics Auton. Syst.*, 15(1-2):25–46, 1995.
- Akihiko Torii, Josef Sivic, Masatoshi Okutomi, and Tomas Pajdla. Visual place recognition with repetitive structures. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2015.
- Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herve Jegou. Training data-efficient image transformers & distillation through attention. In Marina Meila and Tong Zhang, editors, *ICML, 2021*.
- Gido M. van de Ven and A. Tolia. Three scenarios for continual learning. *ArXiv*, 2019.
- Gido M. van de Ven and Andreas Savas Tolia. Generative replay with feedback connections as a general strategy for continual learning. *ArXiv*, 2018.
- Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew W. Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. In *The 9th ISCA Speech Synthesis Workshop*. ISCA, 2016.
- 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 NIPS*, 2017.
- Hongxing Wang, Gangqiang Zhao, and Junsong Yuan. Visual pattern discovery in image and video data: a brief survey. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.*, 2014.
- Yuqing Wang, Zhaoliang Xu, Xinlong Wang, Chunhua Shen, Baoshan Cheng, Hao Shen, and Huaxia Xia. End-to-end video instance segmentation with transformers. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.
- Ross Wightman. Pytorch image models. <https://github.com/rwightman/pytorch-image-models>, 2019.

- Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, Zhengyou Zhang, and Yun Raymond Fu. Incremental classifier learning with generative adversarial networks. *ArXiv*, 2018.
- Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Raymond Fu. Large scale incremental learning. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- Tete Xiao, Mannat Singh, Eric Mintun, Trevor Darrell, Piotr Dollár, and Ross B. Girshick. Early convolutions help transformers see better. *CoRR*, abs/2106.14881, 2021.
- Ju Xu and Zhanxing Zhu. Reinforced continual learning. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems, (NeurIPS)*, 2018.
- Jianwei Yang, Chunyuan Li, Pengchuan Zhang, Xiyang Dai, Bin Xiao, Lu Yuan, and Jianfeng Gao. Focal self-attention for local-global interactions in vision transformers. *ArXiv*, abs/2107.00641, 2021.
- Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong learning with dynamically expandable networks. In *6th International Conference on Learning Representations, ICLR*. OpenReview.net, 2018.
- Lu Yu, X. Liu, and J. Weijer. Self-training for class-incremental semantic segmentation. *ArXiv*, 2020a.
- Lu Yu, Bartłomiej Twardowski, Xialei Liu, Luis Herranz, Kai Wang, Yongmei Cheng, Shangling Jui, and Joost van de Weijer. Semantic drift compensation for class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition CVPR*, 2020b.
- Pei Yu, Yinpeng Chen, Ying Jin, and Zicheng Liu. Improving vision transformers for incremental learning. *ArXiv*, abs/2112.06103, 2021.
- Sangdoon Yun, Dongyoon Han, Sanghyuk Chun, Seong Joon Oh, Youngjoon Yoo, and Junsuk Choe. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *IEEE/CVF International Conference on Computer Vision, (ICCV)*. IEEE, 2019.
- Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *Proceedings of the 34th International Conference on Machine Learning*, Proceedings of Machine Learning Research. PMLR, 2017.