Dual memory system to overcome catastrophic forgetting

Miguel Angel Solinas

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ GRENOBLE ALPES

Spécialité : CIA - Ingénierie de la Cognition, de l'interaction, de l'Apprentissage et de la création

Arrêté ministériel : 25 mai 2016

Présentée par

Mliguel Angel SOLINAS

Thèse dirigée par **Martial MERMILLOD**, Université Grenoble Alpes et co-encadrée par **Marina REYBOZ**,ingénieur chercheur, CEA Grenoble

préparée au sein du **Laboratoire Laboratoire de Psychologie et Neuro Cognition**
dans **l'École Doctorale Ingénierie pour la Santé la Cognition et l'Environnement**

Modèle de mémoire double pour de l'apprentissage incrémental

Dual memory system to overcome catastrophic forgetting

Thèse soutenue publiquement le **9 décembre 2021**,
devant le jury composé de :

Monsieur MARTIAL MERMILLOD
Professeur des Universités, UNIVERSITE GRENOBLE ALPES, Directeur de thèse
Monsieur RUFIN VANRULLEN
Directeur de recherche, CNRS DELEGATION OCCITANIE OUEST, Rapporteur
Monsieur DAMIEN QUERLIOZ
Chargé de recherche HDR, CNRS DELEGATION ILE-DE-FRANCE SUD, Rapporteur
Madame ANNE GUERIN-DUGUE
Professeur des Universités, UNIVERSITE GRENOBLE ALPES, Présidente
Madame LAËTITIA MATIGNON
Maître de conférences, UNIVERSITE LYON 1 - CLAUDE BERNARD, Examinatrice

Acknowledgments

I would like to thank Stéphane Rousset for agreeing to review much of the thesis. Thank you for the excellent discussions we had in the working group. I very much enjoyed the research talks we had, your advice and observations were always very useful. Special thanks to Marina Reyboz and Martial Mermillod for guiding me through the thesis to a fruitful achievement.

Thanks to the members of the CEA and LPNC for their good humour, their solidarity and all the moments of relaxation outside working hours. Thanks in particular to Ivan Panades, Carolynn Bernier, Anca Molnos Diego Puschini for their wise advice and comments.

I would like to thank the interns I was able to manage Luca, Theo, Houssem, Julie, Gianfranco, Loreley, Martin and Caro. To a large extent their work has allowed me to further my knowledge.Thanks for the advice I received from my colleagues Yannick, Romain and Marion during the thesis.

A mi familia y en especial a Margot

Basic mechanisms of learning and forgetting in artificial systems

One of the main characteristics that make human beings unique is their ability to learn continually. It is part of individual development and it is vital to progress and to avoid stagnation. In order to evolve, human beings need to gain experience and acquire competencies to broaden their skills continually. Artificial neural networks lack the capacity to store memories and to learn continually. Indeed, artificial neural networks suffer from catastrophic forgetting of old experiences as new experiences are learned.

Deep learning has yielded remarkable results in many applications; however, artificial neural networks continue to forget. Modeling true continual learning, as humans do, remains a challenge and requires finding appropriate solutions to this problem. For almost three decades, researchers have been dealing with the problem of catastrophic forgetting by studying the neurogenesis of the brain, synaptic consolidation and replay systems.

First, neurogenesis-based approaches evolve the neural network architecture to adapt to different training experiences using independent sets of parameters. Second, synaptic consolidation-based approaches limit the changes in important parameters of previously learned experiences. Thus, new experiences employ neurons that are less useful for previous experiences. Third, replay-based approaches overcome catastrophic forgetting by replaying an amount of previously learned experiences. It is therefore possible to replay previously learned information in two ways: with real samples (rehearsal) or with synthetic samples (pseudo-rehearsal).

Rehearsal approaches replay examples from dedicated memory buffers. Alternatively, pseudo-rehearsal approaches generate pseudo-samples to emulate previously learned data, alleviating the need for dedicated buffers. Revisiting what has been previously learned through examples or pseudo-samples while learning a new tasks allows adapting the global set of parameters for past and new tasks. In this way, it is possible to overcome catastrophic forgetting similarly to classical deep learning training when the entire dataset is present. Since replay methods often rely on limited memory buffers or on roughly generative models, their biggest challenge is to represent correctly and globally what has been previously learned.

This thesis brings together contributions on continual learning, on the properties of autoencoders and knowledge transfer. First, we make a distinction between continual learning and catastrophic forgetting. We highlight certain limitations concerning the settings used to evaluate continual learning approaches and we draw future research tracks. Second, we introduce an auto-associative memory module and a sampling method to generate synthetic samples for capturing and transferring knowledge that replay-based approaches can employ in continual learning. Third, we propose a continual learning model when privacy issues exist. We improve and extend this model by combining pseudo-rehearsal and rehearsal approaches to provide an efficient and competitive solution that improves state-of-the-art results. Finally, in a comprehensive investigation, we attempt to determine which pseudo-samples to use in replay-based approaches to alleviate catastrophic forgetting. We detail methodological aspects of each contribution and we provide evidence for our contributions on datasets such as MNIST, SVHN, CIFAR-10 and CIFAR-100.

Keywords: continual learning – incremental learning – lifelong learning – catastrophic forgetting – catastrophic inference.

Contents

1

Introduction

Humans and animals continuously adapt to their environments by learning from previous experiences and by constantly changing their perceptions of reality. This functional dynamic behaviour is possible due to continuous brain rewiring and due to learning of new concepts based on previous ones. This main feature allows humans and animals to evolve and survive in the non-stationary reality of our world.

The analogous behaviour in artificial intelligence would be an autonomous agent endowed with the ability to evolve. Such an agent should process and adapt to tons of new data generated every day (e.g. images with new tags collected from clinical patients or new customer trends). The new data contains new observations and patterns that can be very different from what has been learned before; for instance, in the medical field, new patient folders, new images of a patient, details of the evolution of the sickness of the patient and so on. This makes it crucial for artificial intelligence to provide agents with deep learning models capable of evolving by considering previous experiences. In other words, such artificial agents would be engaged in continuous operations and they would require continual learning models that gradually expand the experiences acquired for future decision-making.

State-of-the-art deep learning models are capable of outperforming individual tasks (e.g. object recognition, image classification) that were thought to be achievable only by humans (Badia et al. (2020); Senior et al. (2020); Silver et al. (2016)). However, the static models involved behind are incapable of expanding and updating previously learned tasks when new data becomes available or when a different task must be learned. Due to the catastrophic forgetting phenomenon, previously acquired tasks are forgotten as new tasks are learned (McCloskey and Cohen (1989)). Therefore, every time new data is available, classic deep

learning models need to be updated and the whole training process must start from scratch to include the new and the old data. This practice becomes impracticable in real-life time-constrained scenarios where data is non-stationary, it changes over time and it can not be stored because of memory footprints or privacy issues. Artificial neural network models cannot yet rely on previously learned tasks over time, which means they are not yet ready for our non-stationary world.

However, the biological capacity of natural neural networks to store memories and build on previous experiences encourages a better understanding of the catastrophic forgetting phenomenon in artificial neural networks. Although natural cognitive systems can gradually forget previously learned information, catastrophic forgetting is rarely observed in such systems (e.g. fragments of erased memories remain latent in the nervous system Perez et al. (2018)). The French psychologist Ribot (1882) was the first to suggest that memories could be reorganized and be gradually forgotten over time and this later led to the current complementary learning model for consolidation, which commonly represents the hippocampal-neocortical system (McClelland et al. (1995); O'Reilly et al. (2014)). Consequently, natural cognitive systems do not forget "catastrophically" and humans tend to learn sequentially one pattern after another and so on. Although some of the early learned patterns may be seen again, humans do not need to see all the previously learned patterns to remember them. However, much less is known about how these patterns are transformed into lifelong memories in the brain. Several studies suggest that humans can consolidate old memories due to three related mechanisms: the dual hippocampal-neocortex network Barry and Maguire (2019), neurogenesis in the brain Akers et al. (2014) and synaptic consolidation Yang et al. (2014). While adult neurogenesis occurs at a very slow rate Frankland and Bontempi (2005), these studies suggest that patterns of neuronal activity replayed during sleep are likely to be the main mechanisms of brain consolidation that may underlie synaptic consolidation. Some of these biological mechanisms have inspired, to some extent, several research efforts to mitigate catastrophic forgetting in artificial neural networks.

The challenge now is to build resilient artificial systems capable of updating their previous experiences over time when new data becomes available. These systems must be agnostic in terms of what they have learned before and what they will learn. They must also be generic in terms of the problem to be solved. In general, these systems must be capable of continual learning and must be immune to catastrophic forgetting. The construction of such systems has been the main objective guiding this PhD.

1.1 Catastrophic forgetting in Continual Learning

Artificial neural networks learn to perform a *task* (e.g. the process of categorizing a given set of data into classes) by finding an "optimal" point in the parameter space. When ANNs subsequently learn a new task (e.g. the process of categorizing a new set of data into a new class), their parameters will move to a new solution point that allows the ANNs to perform the new task. Catastrophic forgetting McCloskey and Cohen (1989); French (1999) arises when the new set of parameters is completely inappropriate for the previously learned tasks. The latter is mainly a consequence of not taking into consideration previously learned tasks. The gradient descent algorithm adapts, unless regularized in some way, all parameters of an artificial neural network to the new task without considering previous tasks. Catastrophic forgetting is related to the stability-plasticity dilemma French (1997); Abraham and Robins (2005) which is a more general problem in neural networks. Learning models require both plasticity to learn new tasks and stability to prevent forgetting previously learned tasks.

In continual learning, the objective is to overcome the catastrophic forgetting problem by looking for a trade-off between stability and plasticity. For instance, a fully stable system could transform all new information into lifelong memories and learn new things until a memory budget is filled. Therefore, it is necessary to distinguish valuable memories from useless ones and, consequently, the plasticity would help to forget what is not crucial. There are very rare cases of an "unlimited" memory budget in humans who can remember an uncannily large number of experiences; not without adverse effects Van Bree (2016).

The catastrophic forgetting problem in ANNs has been addressed in cognitive sciences since the early 90's McCloskey and Cohen (1989); Robins (1995). The latest development of deep neural networks has led to a higher interest in this field. This challenge is now addressed, with no particular distinction, as continual learning Shin et al. (2017); Parisi et al. (2019), sequential learning McCloskey and Cohen (1989); Aljundi et al. (2018), lifelong learning Rannen et al. (2017); Aljundi et al. (2017); Chaudhry et al. (2018b) and incremental learning Rebuffi et al. (2017); Chaudhry et al. (2018a). In general lines, all of them aim at learning new tasks from a continuous stream of data without forgetting previous tasks. For clarity and simplicity, we will use the expression *continual learning*. The final goal in continual learning is to employ the tasks learned in the past to help future problem-solving.

The easiest way to overcome catastrophic forgetting in continual learning is to learn new training samples jointly with old ones to avoid forgetting previously seen patterns. In this way, the best and most straightforward solution is to store all the previously seen samples; however, this solution is unrealistic for three main reasons: i. large memory footprint requirements are often impracticable for real applications or edge devices, ii. privacy issues are usually a concern when storing raw proprietary data, iii. complete retraining of each new set of incoming data can be infeasible on large scales. Although it is possible to overcome catastrophic forgetting by replaying only a certain amount of previously learned examples, the amount of stored examples plays a critical role and privacy issues remain a problem. Alternatively, it is also possible to replay synthetic samples from a data generator instead of storing examples. Regardless of the provenance of the old samples, a more challenging and open question is *which samples should be replayed to improve the performance in continual learning*.

Early steps to alleviate catastrophic forgetting have mainly focused on replaying old activity patterns while learning new data. The hippocampal-neocortical system has, to some extent, inspired this general concept. It consists in replaying what the neural network might have learned in the past using as input (i) an input stimulus and as output (o) the activation of the network stimulus. The input-output activation patterns represent the *knowledge* of the neural network in a stable state (i.e. before being updated). In addition, the replay of the input-output activation patterns contains enough information to prevent the ANN from catastrophically forgetting previously learned tasks (Robins (1995); Ans and Rousset (1997); French (1997); Li and Hoiem (2017); Rebuffi et al. (2017); Buzzega et al. (2020)). These early attempts to reduce catastrophic forgetting have shown that the replaying of the input-output activation patterns can successfully alleviate catastrophic forgetting not only in classification tasks but also in the continual learning of mathematical operations Ans and Rousset (2000).

Following the resurgence of neural networks in 2012, the problem of catastrophic forgetting in continual learning has received increased attention. Consequently, these early strategies have given rise to several current research efforts that focus primarily on improving the process of knowledge retrieval from input-output activation patterns (Liu et al. (2020); Shim et al. (2020)). Despite all the current advances and broader strategies, there is still an important open question in this area that was flagged up several years ago French (1999):

"How best to optimize the input stimulus used to recover information. Are there ways to improve 'quality' of the input stimulus so that they better reflect the originally learned regularities in the environment?". This thesis attempts to answer this question through an extensive study.

1.2 Focus of the thesis

It is well established that replaying, as previously acquired knowledge, what the neural network have learned through an input stimulus and its corresponding activation output helps to alleviate catastrophic forgetting. Among the strategies employed in the literature, it is possible to identify three widely extended ones to acquire the input stimuli. The first one is a buffer strategy that stores a portion of learned examples to be used later to capture old knowledge through input-output activation patterns. The second strategy consists in generating the synthetic samples by modeling the input distribution. Both strategies implicitly assume that samples that resemble the input distribution are necessary for optimal activation patterns. The third strategy consists in capturing the input-output activation patterns through random stimuli. However, only a few works have focused on an optimal model to improve the knowledge retrieval process through input-output activation patterns. Without focusing specifically on the structure of input stimuli, this work concentrates on finding a model with an optimal architecture to improve the knowledge retrieval process in continual learning tasks.

1.3 Content of the thesis

The main goal of this thesis is to study the catastrophic forgetting problem in artificial neural networks and to propose strategies to alleviate this problem. The primary motivation is to allow artificial neural networks to accumulate previously learned tasks over time and to forget what is not crucial. This is achieved by considering the mapping function of the input-output activation patterns (i.e. knowledge) as a basis for building a "memory" without relying solely on what ANNs have previously seen. The second objective of this thesis is to obtain a model that is agnostic with respect to the dataset and with respect to what is and will be learned. That is to say, a model that does not depend on external information but simply on acquired knowledge. In parallel, our goal is to improve the knowledge retrieval process of the input-output activation patterns to optimally consolidate the learned patterns into lifelong memories.

In Chapter 2, we provide a brief introduction to neural networks and autoencoders. In particular, we present the main concepts of generative models that allow us to distinguish between our contributions and the generative models. Then, we present one of the transversal tools of our work: the process of knowledge distillation and transfer. Next, we present some trends in continual learning and the corresponding jargon. Finally, we describe learning workflows, frequently used evaluation metrics and the datasets used in this thesis.

In Chapter **??**, we take a comprehensive look at the state-of-the-art of catastrophic forgetting since its inception until today. We place continual learning approaches in an Atlas trying to identify clusters of approaches with similar solutions. We analyze 12 clusters found with seven characteristics of continual learning and we detail their limitations, challenges and utilities. We point out some side effects that have emerged in recent years when questioning the difference between catastrophic forgetting, continual learning and their evaluation methods. Next, we outline possible avenues for exploring continual learning.

In Chapter **??**, we demonstrate that an autoencoder can remember and generate what it has learned. We also discuss its subsequent linkage to an auto-associative memory. In particular, we provide mathematical proofs and empirical evidence to show the memory capacity of autoencoders. Then, we present an iterative sampling process called *reinjection* that allows us to sample the training distribution learned by the autoencoder. We build a workflow with its algorithm to exploit the generated samples to transfer knowledge from a trained neural network to an untrained one. Finally, we extend the autoencoder property to a hybrid model that replicates and classifies a given input. We empirically demonstrate that the hybrid model retains the autoencoder property and generates input-output activation patterns (i.e. samples with corresponding labels) useful for knowledge transfer and continual learning.

In Chapter **??**, we present a dual-memory system for continual learning. Specifically, we pair two hybrid models; the first one learns a new task while the second one generates and captures the previous knowledge with input-output patterns. During the continual learning of the new task, the generated input-output activation patterns are replayed and serve to alleviate forgetting. This dual memory framework provides a specific data-free system to alleviate catastrophic forgetting in ANNs with pseudo-samples (i.e. synthetic samples). This solution is suitable for applications where privacy is essential. After analyzing its strengths and limitations, we propose to endow this model with a memory buffer that yields a Combined replay solution. Combined replay combines the rehearsal and the pseudo-rehearsal methods by exploiting the strengths of the hybrid model. In this way, the knowledge retrieval (i.e. the generation of the input-output patterns) process is improved and the forgetting is effectively reduced. This chapter also shows the difficulties in adequately improving the knowledge retrieval process in ANNs and it proposes an effective solution for continual learning problems.

In Chapter **??**, we present a study that is the culmination of the experiences launched throughout this thesis and the previous chapters. During the consolidation process, knowledge is often captured with samples from the distribution, but it is unclear what kind of samples to use. This chapter is a longitudinal study on the question *where is the knowedge in continual learning?* which we try to answer empirically. Precisely, this chapter consists in using several different sources of samples during the consolidation step based on five hypotheses. In trying to answer what information is beneficial to consolidate and to capture during the consolidation process, we point out which samples are beneficial for continual learning.

In Chapter **??**, we conclude with the main contributions and the perspectives for extending the results and understanding of this thesis.

1.3.1 List of publications and patents

PUBLICATIONS The results presented in these doctoral theses and complementary works are summarised in the following contributions.

1.3.1.a Published

- Generalization of iterative sampling in autoencoders Solinas et al. (2020).
- Beneficial effect of combined replay for continual learning Solinas et al. (2021).
- Impact of Spatial Frequency Based Constraints on Adversarial Robustness Bernhard et al. (2021).

1.3.1.b Accepted

- Dream Net: a privacy-preserving continual learning model for facial emotion recognition. (Workshop in International Conference on Affective Computing & Intelligent Interaction 2021).
- Impact of reverberation through deep neural networks on adversarial perturbations (IEEE International Conference on Machine Learning and Applications 2021).

1.3.1.c Submitted

- Beneficial effects of reinjections for continual learning. (SN Computer Science)
- Where is the knowledge in continual learning? (Work completed, pending approval by the patent committee).

Brevets

- Brevet_1 : A data-free transfer knowledge mechanism for neural networks
- Brevet_2 : A data-free continual learning mechanism to alleviate catastrophic forgetting
- Brevet_3 :Iterative sampling for an anomaly detection mechanism

Ongoing work

- Continual learning survey: past, present and future.
- Brevet_4 : Improving the generation of synthetic data for continual learning

2

Background

2.1 Background

Artificial neural networks (ANNs) are mathematical models conceived at the beginning to study, to some extent, the behavior of the human brain. These mathematical models are called neural networks because biological neurons inspire them as McCulloch and Pitts (1943); Hebb and Hebb (1949) formalized it. Artificial neural networks are generally formed by neuron circuits and interconnections between the neurons, known as synapses. They are called connectionist systems because the information is encoded in the parameters of the connections between units.

In the beggining there is no sharp division between connectionism and computational neuroscience, the main difference is that connectionist systems focus on high-level cognitive processes such as recognition, memory, comprehension and reasoning rather than on the specific details of neural functioning. In the 1980s, connectionism experienced a strong

revitalization by formalizing a generic learning rule known as backpropagation LeCun et al. (1988). Backpropagation, which remains the "ace in the hole" of contemporary connectionist research, allows a wide range of ANN models to learn a given mapping function from input to output.

Today, connectionism is mainly encompassed in deep learning as it has led to remarkable advances in broader applications. It is still not clear whether artificial neural networks are limited in some important aspects or not (e.g. catastrophic forgetting and adversarial attack problems); however, it is clear that deep learning models are powerful and flexible enough to cope with many problems.

In this chapter, we briefly introduce background materials related to the current work. We first introduce neural networks and their two-step procedure (i.e. inference and training). We then introduce classifiers, auto-encoder, and some other models and techniques essential to understanding this work. Finally, we give the literature jargon of continual learning and the central evaluation metrics to asses continual learning approaches.

2.2 Neural Networks

In supervised learning, a training dataset $D = (x_i, y_i)$ represents a mapping function ($F : x_i \rightarrow y_i$) defined by observations x_i and their corresponding labels y_i, which are sampled i.i.d. from a distribution $P_{x,y}$. Then, a neural network is employed to learn a mapping function f that approximates F. The objective of a neural network is to correctly match inputs x_i to a target output y_i by adapting its parameters θ. For example, in a digit classification problem, x_i consists of digit images while y_i corresponds to the digit category.

In two steps, a neural network classifier learns a mapping function $y = f(x; \theta)$ and adapts parameters θ that minimize the error between model predictions and the ground truth.

The first step is called feedforward propagation or inference. Input information x flows through the neural network, being multiplied by the intermediate computations to the output. In most cases, the mapping $f(x) : x \rightarrow y$ is a function aggregation described as follows: $f(x) : g_3 \circ g_2 \circ g_1(x)$ where g_l are intermediate layers connected in a chain $y = f(x) = g_3(g_2(g_1(x)))$. The intermediate layers are parametrized by a weight vector θ_l that is employed to weigh the input before being transformed by an activation function. At each hidden layer, the activation function transforms the weighted sum of the inputs into an output. The hidden layers can also contain bias parameters usually employed to shift the weighted sum of the input. A hidden layer is often represented as $g_l(x) = \varphi(\theta_l, x)$ where θ_l is the paremeter of the layer and φ is the activation function of the layer. A neural network comprises an input layer g_1, which is the first layer of the network, an output layer g_3 and several intermediate layers. The last layer, the output layer, is where the prediction is expected. The total length of the chain gives the depth of the model and it is where the term *deep learning* comes from.

The training data provided comprises observations of F evaluated at different points. Each sample x_i is accompanied by a label $y_i = F(x_i)$ that represents the desired output for the output layer. Altogether, the label specifies what the output layer must do at each observation. However, the output behavior of the intermediate layers is not directly specified by the training data, so they are called hidden layers.

The second step is called training and consists in backpropagating an error signal over the model parameters Riedmiller and Braun (1993). A loss functions is used to compute the error made by the model about their predictions; that is, the loss is high when the model is doing a poor job and low when it is performing well. The error signal is the value of the

difference between the predicted labels and the desired true labels. Then, the error value is used to adjust the model parameters to reduce the prediction discrepancy. The gradient of the loss function with respect to the previous layers is employed as the update rule to update the parameters of each layers. The learning process is repeated until convergence, for which an optimizer iteratively computes the gradient on batches randomly sampled from the training set. When a neural network finds an "optimal" set of parameters, it is ready to be deployed and and to make predictions on unseen examples.

A wide range of features allows ANNs to learn f to find a valid mapping between inputs and outputs. For example, to evade the linear constraints, non-linear functions φ are used in the hidden layers instead of linear transformations because they provides higher degrees of freedom that enable models "to understand" the non-linear relationships between the examples x and the corresponding labels y. Non-linear transformations allow transforming non-linear separable problems into linear separable ones. More specifically, non-linear functions allow the input space to be folded so that space can be divided into small linear regions Pascanu et al. (2013). The non-linear transformation, the right loss function and an appropriate number of parameters in the hidden layers allow a neural network to approximate any mapping function. This is the origin of the term universal approximator Hornik et al. (1989). For a detailed introduction to neural networks, please refer to Goodfellow et al. (2016).

The learned mapping function $f(\theta, x)$ is continually updated and evaluated over time in continual learning. When evaluating the performance of the classifier, what is judged behind the scenes is the degree of degradation of the mapping function concerning the original F mapping functions. Ideally, the mapping function would not degrade over time, but it does so due to catastrophic forgetting. Therefore, metrics showing the evolution of $f(\theta, x)$ over learning steps allow us to identify how well a continual learning approach maintains the desired mapping function.

2.2.1 Autoencoders

While a classifier neural network learns a mapping function defined by the observations and their corresponding labels, autoencoders aim at replicating the input at the output layer. The mapping function is self-defined by the input samples and the autoencoder aims to obtain an output similar to its input. An autoencoder is often comprised of an encoder part $e(x)$ that maps the input x into a code z and a decoder part $d(x)$ that maps the code c into the replicated input. Therefore, an autoencoder is usually represented as with two mapping functions $\hat{x} = d(e(x))$, first from input to code $z = e(x)$ and then from code to replications $\hat{x} = d(z)$ where $\hat{x}$ is the replicated vector. In this way, autoencoders are neural networks that minimize the following loss function over the input data and its replication as in Equation 2.1.

$$\mathcal{L}(x, \hat{x}) = -(x\, log(d(e(x))) + (1 - x)\, log(1 - d(e(x)))) \tag{2.1}$$

where x is the input vector of the training distribution and $\hat{x}$ is the predicted output of the autoencoder. Note that in the simplified equation, vector values are evaluated, so 1 represents a vector value of the same dimension as the input. Depending on the input distribution and the model architecture, autoencoders can be trained with several loss functions (e.g. mean squared error). Unless indicated otherwise, along this work, we employ the binary cross-entropy to train the autoencoders.

There is not a unique recipe for autoencoders and its utilization is broader and covers many applications. For example, regarding the dimension of the latent code z on the au-

toencoders, it is possible that the latent code defines compaction or dilation of the input information. Some regularized autoencoders also include prior knowledge to structure the shape of the latent code while improving the replications and increasing generative capabilities. However, autoencoders are not limited to utilizing a latent code or to the classic encoding-decoding behaviour. As long as neural networks replicate inputs into outputs and the loss function is valuated between inputs and replications, it is possible to consider an auto-encoder neural network.

In chapter 3, we revisit some other characteristics of autoencoders that go beyond the encoding-decoding process. These characteristics are exploited later in chapters 4 to build a continual learning solutions.

2.2.1.a Generative Auto-Encoder

In machine learning, real-world data may not be accessible for various reasons (e.g. privacy, lack of data, memory footprint issues, etc.). In these situations, generative models can then be trained to model a given training distribution and be used to synthesize artificial data as variational auto-encoders Kingma and Welling (2013) and adversarial auto-encoder Goodfellow et al. (2014a) do. In this section we present variational auto-encoders and adversarial auto-encoders, two popular generative autoencoders that are quickly revisited in chapter 3.

Variational Auto-Encoder aims to regularize the latent space of the auto-encoder by encouraging the latent space to shape a target distribution (e.g. Gaussian distribution $\mathcal{N}(0,1)$). In this way, the input data is densely located in specific areas of the latent space. An interesting feature of VAEs is that the latent space is continuous and complete Spinner et al. (2018) and it allows sampling of the latent space data to generate samples through the decoding part.

A variational auto-encoder is an auto-encoder with a major difference in the encoder's part. While the auto-encoder simply encodes the input data into a latent variable z, the variational auto-encoder encodes the input data into a prior distribution (i.e. the target distribution). To do so, the encoder part of the variational auto-encoder is trained to minimize the Kullback-Leiber divergence between an encoded distribution Q and the target distribution P. Given the two distributions P and Q defined on the same probability space X, the KL divergence indicates the amount of information that is lost when using Q to represent P as in Equation 2.2.

$$D_{KL}(Q\|P) = \sum_{x=\mathcal{X}} Q(x) \log \frac{Q(x)}{P(x)} \tag{2.2}$$

To set up the encoded distribution with mean and variance parameters, the encoder outputs two vectors: a mean vector μ and the standard deviation vector σ. To sample z from μ and σ, the encoder uses a *scale transformation* of the distribution: it randomly samples ξ from $\mathcal{N}(0,1)$ and computes z from ξ, μ and σ as in Equation 2.3.

$$z = \sigma.\xi + \mu \tag{2.3}$$

While the KL divergence between Q and P encourages every data sample to fit the target distribution, the reconstruction error allows the model to differentiate the data points from different zones. For instance, all the samples in the latent space will overlap into the target distribution if the reconstruction loss is not minimized. In this way, a right balance between reconstruction and regularization allows a variational auto-encoder to compress the

information in a structured latent space and to act as a generative model. A variational auto-encoder is usually trained to minimize the loss described in Equation 2.4.

$$\mathcal{L}_{VAE} = BCE(x, \hat{x}) + D_{KL}(\mathcal{N}(\mu(x), \sigma(x)) \| \mathcal{N}0, 1) \tag{2.4}$$

where BCE corresponds to Equation 2.1 used in a classic autoencoder. In this particular case, the target distribution is a $\mathcal{N}(0, 1)$; however, it can be replaced by a multivariate normal distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ or more complex distributions.

Another way of obtaining a structured latent space is through an adversarial model. The combination of an auto-encoder with an adversarial loss function is known as Adversarial Auto-Encoder (AAE). Both VAEs and AAEs follow the same objective and implement variational inference; however, they differ in how they impose a prior on the latent space. In the case of AAEs, the prior is learned through an adversarial model.

2.2.2 Adversarial examples

Adversarial examples are a particular case of samples that poses a significant problem addressed by a very active research community as shown by the recent explosion in the number of published articles in the field of adversarial machine learning. According to Goodfellow et al. (2014b), adversarial examples are "inputs formed by applying small but intentionally worst-case perturbations to examples from the dataset, such that the perturbed input results in the model outputting an incorrect answer with high confidence".

When training a classifier neural network, is easy to think of the mapping function F as task decision boundaries given by the dataset $(x_i, y_i) \in D$. As a classifier, it must correctly classify the samples of the dataset and it can do so by building decision boundaries that approximate the task boundaries (i.e. $f(x) \approx F(x)$). A trained neural network yield an optimal solution when the task decision boundaries and the model decision boundaries meet enough to correctly classify inputs into the right classes. One common explanation for adversarial examples is based on the usually imperfect matching between task decision boundaries and model decision boundaries. Adversarial examples would be crafted to take advantage of this imprecision of the classifiers. Note that, even if the imprecision is small, it always exists; so, it is always possible to craft adversarial examples if an attacker has access to the model. Although such a crafted sample may not be perceived as modified, the model treats it as a sample of an incorrect class.

Adversarial attacks aim to find a tiny perturbation ϵ, often constrained by a norm (e.g. l_∞, l_2, l_1), and then add that perturbation to a legitimate input $x_i \in D = \{x, y\}_{i=1}^{N}$ to craft an adversarial example. The adversarial example is $x'_i = x_i + \epsilon$ that, in terms of distances, it is quite close to the legitimate input x_i but it is classified differently by the neural network $f(x_i) \neq f(x'_i)$. For instance, a simple adversarial example can be obtained by employing the following perturbation of Equation 2.5.

$$x'_i = x_i + \lambda \, sign(\nabla \mathcal{L}(f(x_i), y_{aux})) \tag{2.5}$$

where x'_i denotes the adversarial example, x_i denotes the legitimate example and λ denotes the strength of the perturbation. $\mathcal{L}(f(x_i), y_{aux})$ denotes the loss function of the classifier (e.g. $\mathcal{L}$ can be the cross entropy loss) given a legitimate input x_i and its desired label y_{aux}.

The process of generating an adversarial example can be done in a targeted or untargeted fashion. The attack is targeted when it is expected that x' belongs to a specific class. In this particular case, the desired label y_{aux} is provided and the sign in the equation is inverted $x'_i = x_i - \lambda \, sign(\nabla \mathcal{L}(f(x_i), y_{aux}))$. It is equivalent to minimize the loss between

the model output and the desired class in contrast to untargeted attacks where the loss is maximized and y_{aux} corresponds to the true label of x_i (i.e. $y_{aux} = y_i$). In the targeted case, the example is modified to approach a class while in the untargeted case, the example is modified to move away from its class.

In Chapter 6, we generate targeted samples on specific regions of the input training distributions to evaluate some continual learning solutions. These crafted samples are exploited to study to what extent the continual learning approaches can be improved with this specific set of samples.

2.2.3 Feature extraction

One of the most important abilities of the deep learning model comes from extensive feature engineering. Deep models disentangle as much of the input data as possible to find common patterns that allow them to establish similarities between seen and unseen samples. Deep neural networks can be employed to extract these high-level representations of the input data. Among the neural networks employed to extract features, convolutional artificial neural networks (CNNs) are a specialized kind of neural networks that allow extracting abstract representations of data LeCun et al. (1989). CNNs are neural networks that use convolutional and pooling operations in place of the general fully connected layers in at least one of their layers. The CNNs are often seen as feature maps because the output features of the CNNs are invariant to small local translations. The values (i.e. features) of most CNN layer outputs do not change for local input translations making CNNs very attractive. For example, to determine whether an image contains a face, a CNN does not learn the location of the eyes with a pixel-perfect match. It only needs to learn that, if there is one eye on the left and one on the right, it is most likely a face. Interestingly, deeper feature maps encode high-level features like "trousers" or "humane eyes" while low-level feature maps detect edges and shapes from the raw input data. Therefore, the deeper feature maps of a CNN contain common and invariant patterns of the image class which are more informative than the raw input data; consequently, they facilitate many downstream tasks (e.g. object recognition and identification).

The feature extraction property of CNNs is not limited to CNNs, but is often a general property of deep models. For example, Transformers Vaswani et al. (2017) neural networks are an attention-based mechanism with similar capabilities to CNNs in disentangling common patterns from input data. Moreover, it has recently been observed that fully connected neural networks can obtain competitive results in large-scale image classification tests, indicating their usefulness as feature extractors Tolstikhin et al. (2021). This work employs deep pre-trained deep models as an auxiliary tool to facilitate some experiments in Chapters 4 and 5.

2.2.4 Distillation

We employ the terminology introduced in Hinton et al. (2015), where the mapping from input vectors to output vectors defines the knowledge of a trained ANN. This abstract view of the knowledge is free from any particular ANN implementation and from any input vector (i.e. knowledge can be captured from any model and with different input vectors).

Knowledge distillation is a technique that aims to capture what a model has previously learned in order to transfer it to another model. This technique was originally designed to capture the knowledge of a large model that has learned a mapping function from a large dataset and to transfer it, later, to a smaller model that is lighter and easier to deploy.

However, the technique implementation is much broader and it is often used to resolve different problems such us model compression 'Hinton et al. (2015), knowledge transfer Yuan et al. (2020) and continual learning Rebuffi et al. (2017).

The real samples(i.e. examples) and their output activations (i.e. pseudo labels *logits*), which are inferred by the trained ANN classifier, are employed to transfer knowledge from a trained ANN classifier to an untrained ANN classifier. The inferred labels, the soft labels, correspond to the relative class probabilities delivered by the trained classifier whereas the ground-truth labels correspond to those given by the real dataset. Intuitively, the output activation pattern of the large model can convey richer information than the true labels of the samples. For instance, when an ANN classifier infers a label value for a given input sample, what the classifier delivers is the probabilities of each class. These outputs expose the decision boundaries learned by the model and the similarities and dissimilarities between the different classes. A new classifier can build similar decision boundaries by learning the examples from the dataset and their corresponding model activation outputs (i.e. input-output activation patterns).

There exist several ways of employing the knowledge of a trained neural classifier (*source*) to train a new one (*target*). In this work, we present the forms that are commonly used in continual learning.

Let us assume the logits with $z = h_\theta(x)$ and the corresponding probability distribution over the classes $f_\theta(x) \triangleq softmax(h_\theta(x))$. Given an input set of samples x, suppose that $\hat{y} = f_s(x)$ is the output of the source classifier and $y^* = f_t(x)$ is the output of the target model. Then, the knowledge distillation loss is defined as in Equation 2.6

$$\mathcal{L}(y^*, \hat{y}) = -KL\left\langle z^*, \hat{z}\right\rangle \tag{2.6}$$

where KL denotes the divergence between the probability distribution of a source model z^* and the distribution over all the classes of a target model $\hat{z}$ (see Equation 2.2). Note that KL can be replaced by $\langle z^*, ., \log(\hat{z})\rangle$ where log is operated entry-wise, by the mean squared error $MSE(z^*, \hat{z})$ or $BCE(z^*, \hat{z})$.

Unless indicated otherwise, when distilling with binary cross-entropy or mean squared error losses, the logits in Equation 2.6 are the output of the last layer $z = h_\theta(x)$ before the softmax activation. When distilling with Kullback–Leibler divergence (KL), the pseudo-probabilities for a softmax output ($softmax(h_\theta(x))$) is obtained using a parameter t called temperature, which for standard *softmax* is set to 1. The temperature generates a smoother output distribution of the soft-targets as it is presented in Equation 2.7.

$$z_i = \frac{y_i^{1/t}}{\sum_j y_j^{1/t}} \tag{2.7}$$

Knowledge distillation is intensively used throughout this work to mitigate catastrophic forgetting and not to deviate the behaviour of the network too much from the optimal one (i.e. for old and new tasks) when learning new data.

2.3 Continual learning

In this section, we present the jargon from the literature and the main concepts of continual learning. Note that there may be different names for similar concepts in the continual learning community. To avoid misunderstandings with parallel works, we present the main concepts used in this paper with a short definition.

2.3.1 Multi-head vs Single-head settings

There exist two main trends in continual learning to alleviate and evaluate catastrophic forgetting. The first trend is called multi-head because this setting assigns a separate output layer (head) for each new task as it is illustrated in Figure 2.1. The models in a multi-head setting only need to classify labels within a head task Chaudhry et al. (2018a). The major drawback of this setting is that it requires additional supervisory signals and a testing time to select the corresponding head task. The task identity (i.e task-id) is provided by an oracle that indicates which head the model must see to get the output prediction. Consequently, this setting can not be employed when task-ids are not delivered.

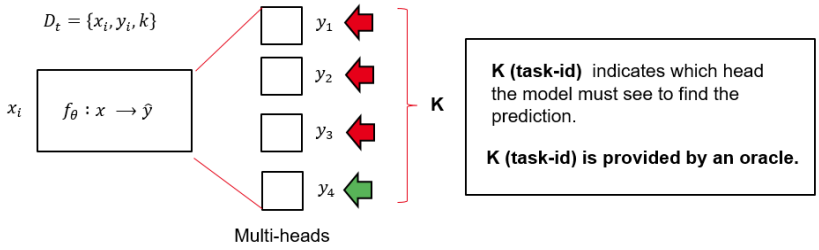

Figure 2.1 – Multi-head setting.

Alliteratively, the single-head setting is employed by default in more realistic scenarios when no oracle is available. Unlike multi-head, the single-head setting does not employ heads or a task-id for each new task and the models learn to classify all labels without a task identity Masana et al. (2020). Figure 2.2 ilustrate the single head setting.

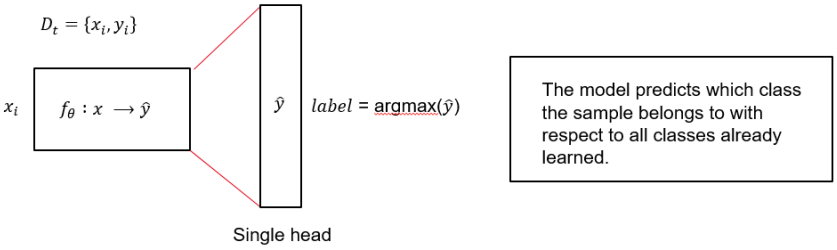

Figure 2.2 – Single-head setting.

It is possible to formulate these settings in the following way: let $D_k = (x_i^k, y_i^k)_{i=1}^{n_k}$ the dataset under test with inputs $x_i^k \in X$, labels $y_i^k \in Y^k$ and task-ids $k \in \mathbb{N}$. Thus, for a single-head setting, the task-id is unknown and the models dispose of the classic tuple (X, Y) whereas for a multi-head setting, the task-id is known and the model disposes of a three-tuple (X, Y, k). For instance, for a dataset of 10 classes resolved in 5 incremental tasks, the multi-head setting will learn to predict a class out of two labels for each of the 5 incremental tasks. In a single-head setting, the model will learn to predict a label out of all the ten classes. A trivial and non-realistic scenario is when there are as many classes as tasks (for instance, 10 classes and its corresponding 10 task-ids). In such a case, the multi-head setting would not need to learn to classify the samples because it receives the task-id for each class, whereas the single-head setting would learn how to predict the 10 labels incrementally from the ten classes.

Throughout this paper, the single-headed evaluation scenario is chosen to evaluate and train our continual learning approaches. Although this scenario is more challenging than a multi-head setting, it is also more realistic and agnostic because it does not rely on a task-id oracle and requires less supervisory signals Van de Ven and Tolias (2019).

2.3.2 Scenarios

Continuous learning solutions seek strategies to accumulate tasks over time. To evaluate the continual learning approaches, the datasets under test are often grouped, split or permuted to form a continual learning scenario. There are two typical ways of creating a continual learning scenario which are described below.

- Sample aggregation: Given a dataset, a model must learn to classify the dataset samples in a first task T_1. Then, a sequence of T tasks is composed with new samples of the already learned classes as it is shown in Figure 2.3. The objective is to incorporate new samples into the model without performance losses on previously learned samples. In this scenario, the task structure does not change but the problem is the changing input distribution. A real example could be the evolution of facial aging or the X-ray image evolution from a patient. Although the owner of the X-ray image or face may change over time, they belong to the same owner in both cases. In this scenario, the task may always be the same but the concept drift (i.e. input distribution changes) leads to forgetting previous samples. Motivated by the sample aggregation scenario, ramdom permutations of the dataset samples are often employed to create more samples. However, it has already been pointed out the shortcomings of sample random permutation due to its simplistic and unrealistic implementation Farquhar and Gal (2018); Hsu et al. (2018).

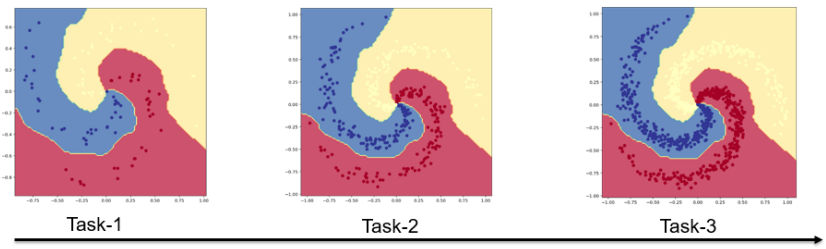

Figure 2.3 – Sample aggregation in continual learning.

- Class aggregation: Given a dataset with several classes, these are grouped into multiple class splits, that form a sequence of tasks $[T_1, T_2, ..., T_n]$. The aim is to incorporate into the model new samples from a new unlearned class without losing performance in previously learned classes, as shown in Figure 2.4. A real example could be to incorporate a new member's facial identity into a facial recognition system or the aggregation of a new skin disease into the hospital's prediction model. In this scenario, the task label and the samples change for each incremental step, so the model aims to exploit and preserve the previously acquired experience to maximize its continual learning performance.

Sample aggregation and class aggregation constitute the majority of the cases used everyday to evaluate and deploy continual learning solutions. The sample aggregation scenario

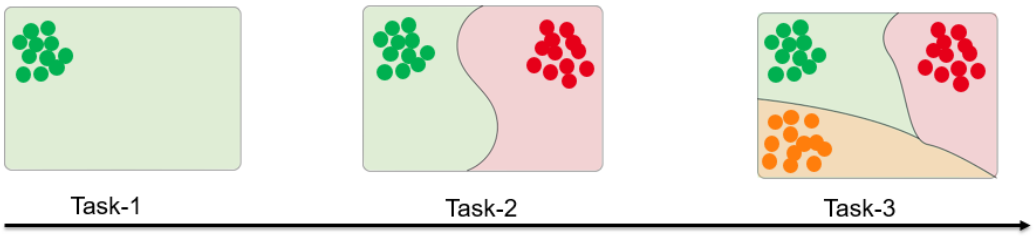

Figure 2.4 – Class aggregation in continual learning.

must to start from an initial training because the idea is to incorporate more samples into previously learned classes. Thus, the models are first pre-trained with samples of all the classes and then new samples are aggregated to the already learned classes. In class aggregation, the model can learn new classes from scratch or learn from previous tasks. While in the first case the objective is to evaluate the performance of an untrained model to incrementally learn new classes from scratch, in the second case, the objective is to maintain the performance of the model on the old tasks and on the new tasks.

This work concentrates its efforts on class aggregation because it is the most challenging one and in it, the neural network has to iteratively modify the class boundaries to correctly classify the new learned classes without forgetting old classes Van de Ven and Tolias (2019).

2.3.3 Metrics

In continual learning, evaluation metrics are broad and encompass various aspects of continuous learning approaches, such as multi-head and single-head evaluation metrics. Without loss of generality, we focus on two simple but essential measures throughout this manuscript. Accuracy (Equation 2.8) and forgetting (Equation 2.9) Chaudhry et al. (2018a) are described as follows: note that the performance of all our experiments are measured with a single-head evaluation metric and we do not use a task identifier.

Accuracy: Let $a_{k,j} \in [0, 1]$ be the accuracy (fraction of correctly classified data from tasks 1 to k after learning the task i). The higher the value of a_k the better the model performance on the classification task.

$$A_T = \frac{1}{T}\sum_{j=1}^{T} a_{T,j} \tag{2.8}$$

Forgetting: Let $f_i \in [-1, 1]$ be the forgetting on task i. It measures the gap between the maximum accuracy obtained in the past and the current accuracy about the same task. The lower the forgetting, the better the model performance.

$$F_T = \frac{1}{T-1}\sum_{j=1}^{T-1} (max \underset{l\in 1,\ldots,i-1}{} a_{l,j}) - a_{i,j} \tag{2.9}$$

2.3.4 Terminology

The following List 2.1 is a brief definition of each of the terms most commonly used in the continual learning community.

Term	Description
Knowledge	Represents what a neural network has previously learned. It is represented by a mapping from input vectors to output vectors or simply by pair between an input stimulus and its activation pattern.
Mapping function	Defines a special type of relationship between a domain element and a second element. A mapping shows how the elements are matched.
Input distribution	Indicates a unique distribution of data generation from a given task.
Task	Based on a problem with predefined inputs and outputs (i.e. the available data), it defines a specific type of prediction or inference. For example, a classification task assigns data to categories.
Catastrophic forgetting	Also known as catastrophic inference, it is the complete loss of previously acquired expertise once a new ones is learned.
Replay	During incremental learning of new classes or new samples, the learner revisits previous tasks through real or synthetic samples.
Rehersal	Performs incremental learning steps while replaying samples already seen from previous tasks (i.e. replaying real samples).
Pseudo-rehersal	Performs incremental learning steps while replaying synthetic samples that mimic previously learned samples from earlier tasks.
Single-head	The model has only one output layer in which all classes are predicted.
Multi-head	The model has as many output layers as tasks learned.

Table 2.1 – Terminology: short description of the main terms used in this work.

2.4 Datasets

The datasets used in the experiments of this thesis are presented below, accompanied by a brief description.

MNIST LeCun et al. (2010): The MNIST dataset of handwritten digits has a training set of 60,000 examples and a test set of 10,000 examples. The MNIST subset belongs to a larger set available called NIST. The digits consist of grayscale images that have been normalized in size and centered on a fixed-size image of 28x28x1 (height, width and channels, respectively). This dataset contains the digits from 0 to 9 providing a dataset of 10 classes.

CIFAR-10 Krizhevsky et al. (2009): The CIFAR-10 dataset consists of 60,000 32x32x3 (height, width and channels, respectively) color images of 10 classes, with 6,000 images per class. For each class, there are 5,000 training images and 1,000 test images.

CIFAR-100 Krizhevsky et al. (2009): CIFAR-100 dataset is the same as CIFAR-10 in terms of image size (32x32x3), except that it has 100 classes in total instead of 10 classes. Each class contains 600 images, where 500 images are training images and 100 images are test images. The 100 classes of CIFAR-100 are grouped into 20 superclasses. Each image has a

"thin" label (the class it belongs to) and a "thick" label (the superclass it belongs to).

SVHN Netzer et al. (2011): SVHN is a real-world image dataset for the development of machine learning algorithms with the minimal pre-processing requirement. It can be considered similar to MNIST (e.g., the images are of cropped small digits); however, it contains more labeled data. It contains 10 classes with approximately 10,000 images per class. SVHN is derived from house numbers in Google Street View and it contains 73,257 images for training, 26,032 images for testing.

ImageNet Deng et al.: ImageNet is an image database organized according to the WordNet hierarchy, which contains 1,000 classes with approximately 1,000 images per class. This dataset is associated with the ILSVRC challenge (ImageNet Large Scale Visual Recognition Challenge). Models usually are pre-trained on ImageNet and then employed for different downstream tasks (e.g. feature extraction).

Bibliography

W. C. Abraham and A. Robins. Memory retention–the synaptic stability versus plasticity dilemma. *Trends in neurosciences*, 28(2):73–78, 2005.

K. G. Akers, A. Martinez-Canabal, L. Restivo, A. P. Yiu, A. De Cristofaro, H.-L. L. Hsiang, A. L. Wheeler, A. Guskjolen, Y. Niibori, H. Shoji, et al. Hippocampal neurogenesis regulates forgetting during adulthood and infancy. *Science*, 344(6184):598–602, 2014.

R. Aljundi, P. Chakravarty, and T. Tuytelaars. Expert gate: Lifelong learning with a network of experts. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3366–3375, 2017.

R. Aljundi, M. Rohrbach, and T. Tuytelaars. Selfless sequential learning. *arXiv preprint arXiv:1806.05421*, 2018.

B. Ans and S. Rousset. Avoiding catastrophic forgetting by coupling two reverberating neural networks. *Comptes Rendus de l'Académie des Sciences-Series III-Sciences de la Vie*, 320(12):989–997, 1997.

B. Ans and S. Rousset. Neural networks with a self-refreshing memory: knowledge transfer in sequential learning tasks without catastrophic forgetting. *Connection science*, 12(1): 1–19, 2000.

A. P. Badia, B. Piot, S. Kapturowski, P. Sprechmann, A. Vitvitskyi, Z. D. Guo, and C. Blundell. Agent57: Outperforming the atari human benchmark. In *International Conference on Machine Learning*, pages 507–517. PMLR, 2020.

D. N. Barry and E. A. Maguire. Remote memory and the hippocampus: A constructive critique. *Trends in cognitive sciences*, 23(2):128–142, 2019.

R. Bernhard, P.-A. Moellic, M. Mermillod, Y. Bourrier, R. Cohendet, M. Solinas, and M. Reyboz. Impact of spatial frequency based constraints on adversarial robustness. *arXiv preprint arXiv:2104.12679*, 2021.

P. Buzzega, M. Boschini, A. Porrello, D. Abati, and S. Calderara. Dark experience for general continual learning: a strong, simple baseline. *arXiv preprint arXiv:2004.07211*, 2020.

A. Chaudhry, P. K. Dokania, T. Ajanthan, and P. H. Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 532–547, 2018a.

A. Chaudhry, M. Ranzato, M. Rohrbach, and M. Elhoseiny. Efficient lifelong learning with a-gem. *arXiv preprint arXiv:1812.00420*, 2018b.

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database.

S. Farquhar and Y. Gal. Towards robust evaluations of continual learning. *arXiv preprint arXiv:1805.09733*, 2018.

P. W. Frankland and B. Bontempi. The organization of recent and remote memories. *Nature reviews neuroscience*, 6(2):119–130, 2005.

R. M. French. Pseudo-recurrent connectionist networks: An approach to the'sensitivity-stability'dilemma. *Connection Science*, 9(4):353–380, 1997.

R. M. French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135, 1999.

I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio. *Deep learning*, volume 1. MIT press Cambridge, 2016.

I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial networks. *arXiv preprint arXiv:1406.2661*, 2014a.

I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014b.

D. O. Hebb and D. Hebb. *The organization of behavior*, volume 65. Wiley New York, 1949.

G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. *NIPS Deep Learning and Representation Learning Workshop*, 2015.

K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

Y.-C. Hsu, Y.-C. Liu, A. Ramasamy, and Z. Kira. Re-evaluating continual learning scenarios: A categorization and case for strong baselines. *arXiv preprint arXiv:1810.12488*, 2018.

D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Y. LeCun, D. Touresky, G. Hinton, and T. Sejnowski. A theoretical framework for back-propagation. In *Proceedings of the 1988 connectionist models summer school*, volume 1, pages 21–28, 1988.

Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

Y. LeCun, C. Cortes, and C. Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

Z. Li and D. Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

Y. Liu, Y. Su, A.-A. Liu, B. Schiele, and Q. Sun. Mnemonics training: Multi-class incremental learning without forgetting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12245–12254, 2020.

M. Masana, X. Liu, B. Twardowski, M. Menta, A. D. Bagdanov, and J. van de Weijer. Class-incremental learning: survey and performance evaluation. *arXiv preprint arXiv:2010.15277*, 2020.

J. L. McClelland, B. L. McNaughton, and R. 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*, 102(3):419, 1995.

M. McCloskey and N. J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier, 1989.

W. S. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. 2011.

R. C. O'Reilly, R. Bhattacharyya, M. D. Howard, and N. Ketz. Complementary learning systems. *Cognitive science*, 38(6):1229–1248, 2014.

G. I. Parisi, R. Kemker, J. L. Part, C. Kanan, and S. Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, 2019.

R. Pascanu, G. Montufar, and Y. Bengio. On the number of response regions of deep feed forward networks with piece-wise linear activations. *arXiv preprint arXiv:1312.6098*, 2013.

L. Perez, U. Patel, M. Rivota, I. E. Calin-Jageman, and R. J. Calin-Jageman. Savings memory is accompanied by transcriptional changes that persist beyond the decay of recall. *Learning & Memory*, 25(1):45–48, 2018.

A. Rannen, R. Aljundi, M. B. Blaschko, and T. Tuytelaars. Encoder based lifelong learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1320–1328, 2017.

S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010, 2017.

T. Ribot. *Diseases of memory: An essay in the positive psychology*, volume 43. D. Appleton, 1882.

M. Riedmiller and H. Braun. A direct adaptive method for faster backpropagation learning: The rprop algorithm. In *IEEE international conference on neural networks*, pages 586–591. IEEE, 1993.

A. Robins. Catastrophic forgetting, rehearsal and pseudorehearsal. *Connection Science*, 7 (2):123–146, 1995. doi: 10.1080/09540099550039318. URL https://doi.org/10.1080/09540099550039318.

A. W. Senior, R. Evans, J. Jumper, J. Kirkpatrick, L. Sifre, T. Green, C. Qin, A. Žídek, A. W. Nelson, A. Bridgland, et al. Improved protein structure prediction using potentials from deep learning. *Nature*, 577(7792):706–710, 2020.

D. Shim, Z. Mai, J. Jeong, S. Sanner, H. Kim, and J. Jang. Online class-incremental continual learning with adversarial shapley value. *arXiv e-prints*, pages arXiv–2009, 2020.

H. Shin, J. K. Lee, J. Kim, and J. Kim. Continual learning with deep generative replay. *arXiv preprint arXiv:1705.08690*, 2017.

D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

M. Solinas, C. Galiez, R. Cohendet, S. Rousset, M. Reyboz, and M. Mermillod. Generalization of iterative sampling in autoencoders. In *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 877–882. IEEE, 2020.

M. Solinas, S. Rousset, R. Cohendet, Y. Bourrier, M. Mainsant, A. Molnos, M. Reyboz, and M. Mermillod. Beneficial effect of combined replay for continual learning. In *ICAART (2)*, pages 205–217, 2021.

T. Spinner, J. Körner, J. Görtler, and O. Deussen. Towards an interpretable latent space: an intuitive comparison of autoencoders with variational autoencoders. In *IEEE VIS 2018*, 2018.

I. Tolstikhin, N. Houlsby, A. Kolesnikov, L. Beyer, X. Zhai, T. Unterthiner, J. Yung, D. Keysers, J. Uszkoreit, M. Lucic, and A. Dosovitskiy. Mlp-mixer: An all-mlp architecture for vision, 2021.

T. Van Bree. Digital hyperthymesia-on the consequences of living with perfect memory. 2016.

G. M. Van de Ven and A. S. Tolias. Three scenarios for continual learning. *arXiv preprint arXiv:1904.07734*, 2019.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.

G. Yang, C. S. W. Lai, J. Cichon, L. Ma, W. Li, and W.-B. Gan. Sleep promotes branch-specific formation of dendritic spines after learning. *Science*, 344(6188):1173–1178, 2014.

F. Yuan, L. Shou, J. Pei, W. Lin, M. Gong, Y. Fu, and D. Jiang. Reinforced multi-teacher selection for knowledge distillation. *arXiv preprint arXiv:2012.06048*, 2020.

List of Figures

List of Tables