Continual learning for hierarchical classification, few-shot recognition, and multi-modal learning

Kai Wang

Universitat Autònoma
de Barcelona

Continual learning for hierarchical classification, few-shot recognition, and multi-modal learning

A dissertation submitted by **Kai Wang** to the Universitat Autònoma de Barcelona in fulfilment of the degree of **Doctor of Philosophy** in the Departament de Ciències de la Computació.

Bellaterra, March 16, 2022

Director	**Dr. Joost van de Weijer** Centre de Visió per Computador Universitat Autònoma de Barcelona **Dr. Luis Herranz** Centre de Visió per Computador Universitat Autònoma de Barcelona
Thesis committee	**Dr. Vincenzo Lomonaco** Department of Computer Science University of Pisa **Dr. Trzciński Tomasz** Faculty of Electronics and Information Technology Warsaw University of Technology **Dr. Jorge Bernal Del Nozal** Centre de Visió per Computador Universitat Autònoma de Barcelona

This document was typeset by the author using LaTeX 2ε.

ISBN: 978-84-124793-2-4

Printed by Ediciones Gráficas Rey, S.L.

Acknowledgements

First, I have to appreciate all the help and education from my supervisor Joost van de Weijer and Luis Herranz, without whom I cannot grow up to be a researcher. Thanks to all their patience to supervise me. Moreover, Xialei Liu and Andy Bagdanov are also playing the role of supervisors in my PhD career. They helped me to quickly accumulate my knowledge in the computer vision area.

Second, I need to say thanks to my family. The support from my parents (Xiulan Meng, Minquan Wang) makes me far away from any worries. Also Qiuyue Yang, who accompanied me for these happy four years. Then also other family members (Minjie Wang family, Haitao Meng family, Guizhi Wang family, Xiuli Meng family, Xiuhong Meng family, Guique Wang family, Guimin Wang family, Guique Wang family, Fangyong Wang family. Sorry I cannot list all your names here), who also cast me quite a lot of concerns especially in the hardest Covid-19 quarantine time.

Third, thanks to all co-authors who have surely helped me in the past years. And also thanks to the Huawei company, which is always supporting our projects. This gave us more opportunities and probabilities in the past three years. Fourth, I'm lucky since my home mates in these years are all nice: Wenjie Qian, Changyong Lu, Jiarui Li, Kaidi Hu, Xu Han, Ting Zhang.

Fifth, the support from CVC guys cannot be neglected, their job makes the PhD journey smoother and easier: Montse, Gigi, Kevin, Laura, Carola, Clara, Eva, Mireia and so on. And also the accompanying from these friends from CVC makes life colorful and happy: Fei Yang, Jose Luis Gomez, Shiqi Yang, Yi Xiao, Yixiong Yang, Danna Xue, Yaxing Wang, Xialei Liu, Lu Yu, Hector, Sanket, Ali, Mohamod, Idoia and so on.

Sixth, in SAF I also made quite a lot enthusiastic friends who helped me to improve my powerlifting level and enrich my knowledge on sports: Enric, Mikel, Dimitrii, Daniel, David, Toni, Lilia, Nati, Alex, Eric, Kuai Yu, Yunong Liu, Jinge Yang, Ziwen Wang, Shishuang Xiang. Also, I would never forget the happy time each weekend in the basketball court of Cerdanyola del Valles with all of you: Wenchao

Duan, Jing Li, Jose et al.

Last but not least, thanks to Jilin University and CSC committee: Hongwei Zhao, Mo Dong, Zhen Wang and so on. Without your support I cannot be luckily to come to Spain. And also it was an enjoyable time to learn Spanish in Beijing Language and Culture University under the instructions of Min Pan and Yiyun Hu.

Abstract

Deep learning has drastically changed computer vision in the past decades and achieved great success in many applications, such as image classification, retrieval, detection, and segmentation thanks to the emergence of neural networks. Typically, for most applications, these networks are presented with examples from all tasks they are expected to perform. However, for many applications, this is not a realistic scenario, and an algorithm is required to learn tasks sequentially. Continual learning proposes theory and methods for this scenario.

The main challenge for continual learning systems is called catastrophic forgetting and refers to a significant drop in performance on previous tasks. To tackle this problem, three main branches of methods have been explored to alleviate the forgetting in continual learning. They are regularization-based methods, rehearsal-based methods, and parameter isolation-based methods. However, most of them are focused on image classification tasks. Continual learning of many computer vision fields has still not been well-explored. Thus, in this thesis, we extend the continual learning knowledge to meta-learning, we propose a method for the incremental learning of hierarchical relations for image classification, we explore image recognition in online continual learning, and study continual learning for cross-modal learning.

In this thesis, we explore the usage of image rehearsal when addressing the incremental meta-learning problem. Observing that existing methods fail to improve performance with saved exemplars, we propose to mix exemplars with current task data and episode-level distillation to overcome forgetting in incremental meta-learning. Next, we study a more realistic image classification scenario where each class has multiple granularity levels. Only one label is present at any time, which requires the model to infer if the provided label has a hierarchical relation with any already known label. In experiments, we show that the estimated hierarchy information can be beneficial in both the training and inference stage.

For the online continual learning setting, we investigate the usage of intermediate feature replay. In this case, the training samples are only observed by the model only one time. Here we fix the memory buffer for feature replay and compare the effectiveness of saving features from different layers. Finally, we investigate

multi-modal continual learning, where an image encoder is cooperating with a semantic branch. We consider the continual learning of both zero-shot learning and cross-modal retrieval problems.

Key words: *continual learning, zero-shot learning, image recognition, cross-modal retrieval, few-shot learning*

Resumen

El aprendizaje profundo ha cambiado drásticamente la visión por computador en las últimas décadas y ha logrado un gran éxito en muchas aplicaciones, como la clasificación, recuperación, detección y segmentación de imágenes gracias al surgimiento de las redes neuronales. Por lo general, para la mayoría de las aplicaciones, a estas redes se les enseñan ejemplos de todas las tareas que se espera que realicen. Sin embargo, para muchas aplicaciones, este no es un escenario realista y se requiere que un algoritmo aprenda tareas de forma secuencial. El aprendizaje continuo propone teoría y métodos para este escenario.

El principal desafío para los sistemas de aprendizaje continuo se denomina olvido catastrófico y se refiere a una degradación significativa del rendimiento en las tareas anteriores. Para abordar este problema, se han explorado tres ramas principales de métodos para aliviar el olvido en el aprendizaje continuo. Estos son métodos basados en regularización, métodos basados en ensayos y métodos basados en aislamiento de parámetros. Sin embargo, la mayor parte de estos métodos están enfocados a tareas de clasificación de imágenes. El aprendizaje continuo en muchos campos de visión por computador aún no se ha explorado en profundidad. Por lo tanto, en esta tesis, extendemos el aprendizaje continuo a métodos de meta-aprendizaje, proponemos un método para el aprendizaje incremental de relaciones jerárquicas para la clasificación de imágenes, exploramos el reconocimiento de imágenes en el aprendizaje continuo en línea y estudiamos el aprendizaje continuo dentro del aprendizaje intermodal.

En esta tesis, exploramos la repetición de ejemplares para abordar el problema del meta-aprendizaje incremental. Al observar que los métodos existentes no logran mejorar el rendimiento con ejemplos guardados, proponemos mezclar ejemplos con datos de tareas actuales y destilación a nivel de episodio para superar el olvido en el meta-aprendizaje incremental. A continuación, estudiamos un escenario de clasificación de imágenes más realista, donde cada clase tiene múltiples niveles de granularidad. Sólo hay una etiqueta presente en todo momento, lo que requiere que el modelo deduzca si la etiqueta proporcionada tiene una relación jerárquica

con alguna otra etiqueta ya conocida. En los experimentos, mostramos que la información de la jerarquía estimada puede ser beneficiosa tanto en la etapa de entrenamiento como en la de inferencia.

Para el caso del aprendizaje continuo en línea, investigamos la repetición de características intermedias. En este caso, el modelo sólo observa las muestras de entrenamiento una sola vez. Aquí fijamos el tamaño del búfer de memoria para la repetición de características y comparamos la efectividad de guardar características de diferentes capas. Finalmente, investigamos el aprendizaje continuo multimodal, donde un codificador de imágenes coopera con una rama semántica. Consideramos el aprendizaje continuo en problemas tanto de aprendizaje sin ejemplos como de recuperación multimodal.

Palabras clave: *aprendizaje continuo, aprendizaje sin ejemplos, reconocimiento de imágenes, recuperación multimodal, aprendizaje con pocos ejemplos*

Resum

L'aprenentatge profund ha canviat dràsticament la visió per computador en les darreres dècades i ha aconseguit un gran èxit en moltes aplicacions, com ara la classificació, recuperació, detecció i segmentació d'imatges gràcies al sorgiment de les xarxes neuronals. En general, per a la majoria de les aplicacions, aquestes xarxes mostren exemples de totes les tasques que s'espera que realitzin. No obstant això, per a moltes aplicacions, aquest no és un escenari realista i cal que un algoritme aprengui tasques de forma seqüencial. L'aprenentatge continu proposa teoria i mètodes per a aquest escenari.

El principal desafiament per als sistemes d'aprenentatge continu s'anomena oblit catastròfic i fa referència a una degradació significativa del rendiment en les tasques anteriors. Per abordar aquest problema, s'han explorat tres branques principals de mètodes per alleujar l'oblit en l'aprenentatge continu. Aquests són mètodes basats en regularització, mètodes basats en assaigs i mètodes basats en aïllament de paràmetres. No obstant això, la major part d'aquests mètodes estan enfocats a tasques de classificació d'imatges. L'aprenentatge continu en molts camps de visió per computador encara no ha estat explorat en profunditat. Per tant, en aquesta tesi extenem l'aprenentatge continu a mètodes de meta-aprenentatge, proposem un mètode per a l'aprenentatge incremental de relacions jeràrquiques per a la classificació d'imatges, explorem el reconeixement d'imatges en l'aprenentatge continu en línia i estudiem l'aprenentatge continu dins de l'aprenentatge intermodal.

En aquesta tesi, explorem la repetició d'exemplars per abordar el problema del meta-aprenentatge incremental. En observar que els mètodes existents no aconsegueixen millorar el rendiment amb exemples desats, proposem barrejar exemples amb dades de tasques actuals i destil·lació a nivell d'episodi per superar l'oblit en el meta-aprenentatge incremental. A continuació, estudiem un escenari de classificació d'imatges més realista on cada classe té múltiples nivells de granularitat. Només hi ha una etiqueta present en tot moment, cosa que requereix que el model dedueixi si l'etiqueta proporcionada té una relació jeràrquica amb alguna altra etiqueta ja coneguda. En els experiments, mostrem que la informació de la jerarquia estimada pot ser beneficiosa tant a l'etapa d'entrenament com a la d'inferència.

Per al cas de l'aprenentatge continu en línia, investiguem la repetició de ca-

racterístiques intermitges. En aquest cas, el model només observa les mostres d'entrenament una sola vegada. Aquí fixem la mida del búfer de memòria per a la repetició de característiques i comparem l'efectivitat de guardar característiques de diferents capes. Finalment, investiguem l'aprenentatge continu multimodal on un codificador d'imatges coopera amb una branca semàntica. Considerem l'aprenentatge continu en tant problemes d'aprenentatge sense exemples com de recuperació multimodal.

Paraules clau: *aprenentatge continu, aprenentatge sense exemples, reconeixement d'imatges, recuperació multimodal, aprenentatge amb pocs exemples*

Contents

List of Figures

List of Tables

1 Introduction

Over the last couple of decades the availability of digital data has seen an explosive growth. This growth has been accompanied with a desire to understand, learn, analyze this data in a wide range of applications. Artificial intelligence has since had a considerable impact on a wide range of applications in the life of human beings: such as face recognition [36, 92, 118], recommendation systems [109], autonomous driving [41], robotics [84], game AI [157, 185], medical analysis [66, 100].

Especially, deep learning has much contributed to the current AI revolution [82, 85]. The original techniques and theory underlying deep learning had already been designed in the 80s [86, 147]. The availability of large labelled datasets, and breaktroughs in hardware (GPUs), combined with improved optimization techniques allowed deep learning to outperform traditional hand-crafted features combined with classifiers by a large margin. Since then, deep learning has been applied to the vast majority of computer vision tasks.

However, artificial intelligence based on deep learning [85] highly relies on a huge amount of data to obtain optimal performance. For example, image classification pretraining on ImageNet [82], or text-image alignment model CLIP [138] pretrained on 14M images. Thus, data collection, annotation and relevant privacy issues have been a bottleneck in the development of real-world applications. Due to these reasons, updating the deep learning models while incrementally inputting data has been a choice to solve the data bottleneck.

Different from deep learning models, humans are good at learning incrementally during their lifetime and do not suffer from much forgetting; we refer to this phenomenon as lifelong learning. For example, after we have learned how to distinguish *dogs* and *cats* during our childhood, we would not forget this during our whole life while learning many new tasks. However, this is not true for a robot, which we expect to behave similarly to a human in the future, as is illustrated in Fig. 1.1.

To make the robots and deep learning models have the same lifelong learning ability as human beings, various methods have been explored to overcome forgetting during the learning procedure. These methods belong to the computer vision subarea named *Continual Learning* [89, 107]. More specifically, in this thesis, we aim to solve continual learning in few-shot recognition, hierarchical classification and multi-model learning problems.

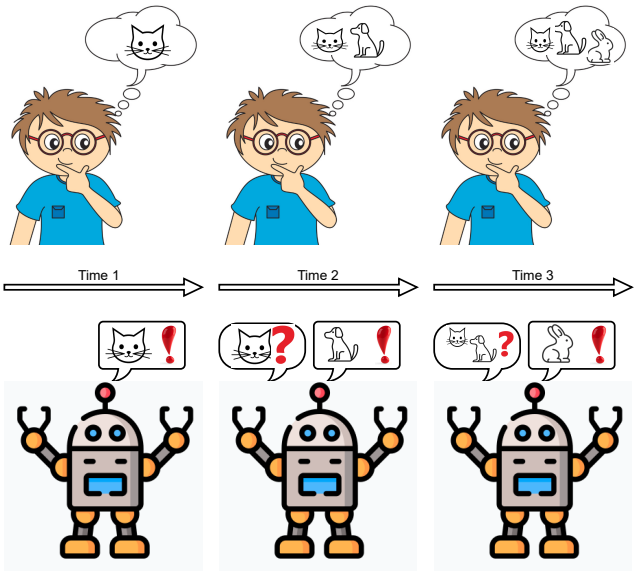


Figure 1.1 – Comparison between human and robot lifelong learning. Robots are prone to forget the previous knowledge, while humans not. Continual learning algorithms aim to make algorithms behave more similar to humans, to be able to learn new knowledge while preserving knowledge previously learned.

1.1 Continual learning

In most deep learning research, the training data is considered to be present jointly. Thus, the learner can process the whole dataset several times to get an optimal model for a specific problem. However, this situation cannot be always guaranteed in many real-world application scenarios. Due to privacy issues or storage restrictions, the learner could be restricted to only have access to the training data of a single task at a time. This scenario is referred to as *continual learning* (or *incremental learning* or *lifelong learning*). The main challenge in this scenario is learning a model from the current data (referred to as current task), while preventing forgetting knowledge of previous tasks. As a baseline, which applies finetuning to the model always suffers from a considerable decrease in performance on the previous tasks since the learner is trained to be optimal for the current task due to the adaptation to the current data. This phenomenon is called *catastrophic forgetting [76, 117]*. The field of continual learning aims at studying methods which prevent forget-

ting [33, 114, 142, 184]. Continual learning methods can be categorized into three main groups [33]).

Regularization-based continual learning. The first group is based on regularization. Based on the importance estimation of the parameters in the model, these methods apply a regularization loss term to the final loss function. In this way, the knowledge from previous tasks can be kept by constraining the parameter changes. The difference among these methods mainly lies in how to estimate the importance. From these differences, these methods can be further divided into data-focused [94] and prior-focused [76]. Data-focused methods, including LwF [94], LFL [71], EBLL [141] and DMC [202] use knowledge distillation [58] from previous models. Prior-focused methods, including EWC [76], IMM [88], SI [198], R-EWC [104] and MAS [4], focus on obtaining prior knowledge by estimating the importance of model parameters.

Rehearsal-based continual learning. These methods overcome forgetting by replay the previous data (real or synthetic). This branch can be divided into two main strategies: exemplar-rehearsal [63,184], which are methods that save a small portion of previous task training data (named exemplars); and pseudo-rehearsal [54, 183], which are methods that where generative models are trained to generate previous data at the image or feature level. Then, the replayed data is used to balance the bias for joint classifiers or to regularize the gradients in model updating.

Parameter isolation methods. This family focuses on allocating different model parameters to each task. They begin with a simple architecture and update continually with new neurons to allow additional capacity needed for the new task. Depending on whether the network is fixed or dynamic from the beginning, this branch can be further divided into two groups: fixed network and dynamic network. The fixed network group, including Piggyback [111], PackNet [112] and HAT [154], learns a mask for each task, which is applied to the weights or activations in the networks. This group is further developed to the case where no forgetting is allowed in [114]. However, this group is limited to the task-aware setting. The dynamic network group, represented by Expert Gate [7], avoids this problem by learning an autoencoder gate to obtain the task oracle.

Based on an analysis of the state-of-the-art of continual learning, we have identified five research directions that we pursue in this thesis, and which we will outline in the following sections.

1.1.1 Incremental meta learning

Meta-learning is a learning paradigm, where a model gains experience over a series of sampled episodes. In this way, the model gains the ability to generalize to unseen

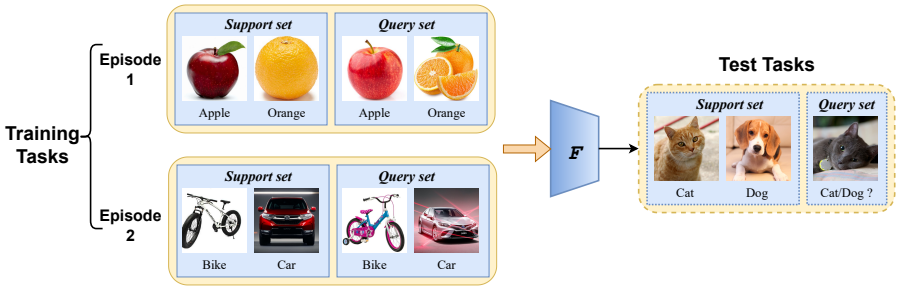


Figure 1.2 – Illustration of *few-shot learning*. The task at test time is to correctly classify between a set of classes based on only a few training samples from these classes. During training the network is optimized to generalize well to unseen classification problems. In this thesis, we investigate this problem in a continual learning setting.

tasks [61]. Meta-learning is a promising technique to make models generalize to new tasks and environments which are absent during training. This ability is considered to be crucial for future AI systems to make them behave more human-like. Few-shot learning has been considered as the paradigm-of-choice to test and evaluate meta-learning algorithms (see Figure 1.2). It aims at learning models from just few samples, and particularly meta-learning applied to few-shot image classification has attracted interest in recent years [16, 90, 163, 193]. Few-shot learning can be categorized into three types of approaches: data augmentation, model enhancement and algorithm-based methods [179].

However, most few-shot learning methods are limited by their application scenarios, since they require a large number of training data for meta-learning. This will lead to poor performance in continual learning scenarios where the training data arrives incrementally and there will not be sufficient categories at any time stage to learn a good generalization model. Therefore continual learning [33,76,142,152,197] research on this topic is crucial to solve this dilemma. To address both the problem of incremental learning and meta-learning, *incremental meta-learning* was proposed as a way of performing few-shot learning in incremental learning scenarios [99].

To address the incremental meta-learning problem, Liu et al. [99] proposed the Indirect Discriminant Alignment (IDA) as a solution to incrementally accumulate the knowledge for the meta-learner. In IDA, class prototypes from previous tasks are represented by *anchors*, which are computed as the average of all images belonging to a specific class. Then these prototypes are used to distill knowledge from previous

models. As shown in their paper, this strategy greatly reduces forgetting for short sequences of tasks. Based on IDA, they proposed EIML as an extension of IDA with exemplars. But surprisingly, their results showed that EIML fails to outperform IDA, even with increasing numbers of exemplars. This is counter-intuitive in continual learning, since exploiting exemplars generally boosts performance in incremental learning [17, 113, 142]. One of these probable reasons for this is that in EIML, exemplars are only used to recompute the class centers for previous tasks, which underestimate the usage of exemplars. *Therefore, in this thesis, we will explore how to correctly use exemplars to increase the efficiency of the knowledge transfer in incremental meta-learning.*

1.1.2 The incremental learning of hierarchical knowledge

In human learning, the association of new concepts to old concepts is accumulated over time. People construct a hierarchy of knowledge to better consolidate information. For example, as a child one might learn the concept of animal first, whereas later one might learn about various animals such as birds and monkeys. Even later one would be able to distinguish several monkeys like for example gorillas and chimpanzees. Typically, image classification problems in computer vision have a flat structure and ignore this hierarchical nature of knowledge. However, there have been a number of works that consider hierarchical work [156], but these works do not consider the incremental learning of the hierarchical knowledge. An example of the hierarchical knowledge graph is provided in Figure 1.3.

Recently, the IIRC (Incremental Implicitly-Refined Classification) setup [1] has been proposed as a novel extended benchmark to evaluate lifelong learning methods in a realistic setting where the construction of hierarchical knowledge is key. In the IIRC benchmark, each class has multiple granularity levels. But only one label is present at any time, which requires the model to infer whether the related labels have been observed in previous tasks. This setting is much closer to real-life learning, where a learner gradually improves its knowledge of objects.

Based on the IIRC benchmark, Abdelsalam et al. [1] adapted and evaluated several state-of-the-art continual learning methods to solve this problem, including iCaRL [142], LUCIR [63], and AGEM [27]. However, their work does not propose an effective solution specifically designed for the IIRC problem. In particular, they do not incrementally learn the hierarchical knowledge, which is important to correctly label the data in this setting. In addition, at inference time they do not use the hierarchical knowledge to verify the plausibility of the label predictions. For example, an image that is classified as a 'bird' on one level of the hierarchy should not be classified as a 'chimpanzee' at the other level. *Therefore, in this thesis, we aim to propose a continual learning method tailored for the learning of hierarchical*

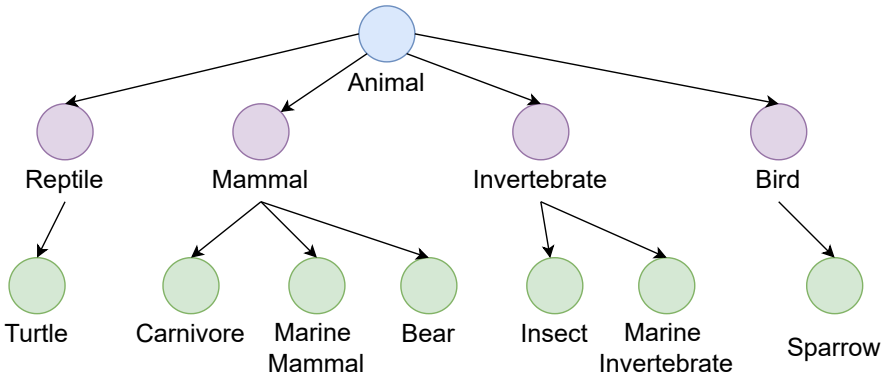


Figure 1.3 – Partial representation of the hierarchical information of the Imagenet dataset [148]. We investigate how hierarchical relations between classes can be learned incrementally.

classification tasks.

1.1.3 Online continual learning with compressed feature replay

One of the challenging settings in continual learning is *online continual learning* of non-iid data streams [6, 27, 108]. Under the online continual learning setting, each image can only be seen one time during training (except exemplars in storage). The application scenarios mainly exist in resource constrained devices, such as mobile phones, robots and other smart devices. The majority of methods in continual learning allow for multiple cycles over the training data [33]. Thus, these continual learning methods cannot be directly operated on this challenging online continual learning setting. Moreover, they take longer to train for several epochs. In this chapter, we focus on online continual learning setup.

Among the continual learning approaches, some of the best performance is obtained with rehearsal-based methods [54, 142, 155]. The rehearsal of images from the previous tasks is one of the most popular strategies [27, 63, 142, 184]. However, this strategy leads to memory increasing and the imbalance problem of training data between the previous tasks and the current one. An alternative way to generate images is using generative models (e.g. GANs) [155, 183]. However, image generation itself is still a hard problem and may require complex generative models, which would also be continually learned, making this method not very practical for continual learning on complex datasets.

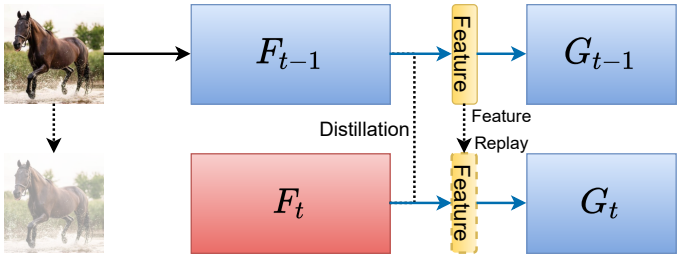


Figure 1.4 – Illustration of feature replay. Features from previous tasks are replayed when training the current task to prevent forgetting. In this thesis, we study compressed feature replay that allows performing feature replay at deeper layer in neural networks. As a result, it can obtain improved plasticity.

To overcome the shortcomings of image replay, recent papers have paid attention to feature replay (as shown in Figure 1.4) in continual learning [54, 106]. GFR [106] trains a generator to replay compact features of the images (after the last average pooling layer of a ResNet-18). REMIND [54] saves feature exemplars since it is very efficient and requires therefore less memory per image. To further reduce the memory usage, REMIND [54] applies product quantization [65] to compress the features.

However, replaying on the feature level does not allow much training of the feature extractor before the replay layer. As a consequence, if this backbone is not optimal enough for future tasks, the performance is sub-optimal for continual learning. This significantly limits the learning of representations that can discriminate between all tasks. *Therefore, we propose to investigate compression of the replay features which would allow it to move to deeper layers for replay and consequently for more plasticity during training.*

1.1.4 Multi-modal continual learning

One important direction in deep learning is multi-modal learning where multiple data modalities are considered [125, 168]. Humans when learning use multiple senses, including vision, hearing, taste, touch and smell. In computer vision, there has been much attention for the joint learning of semantic text and vision embeddings [95, 138]. However, this research field has not yet been much investigated within the context of non-stationary training data as the case in continual learning setups. We have identified two aspects of continual multi-modal training that we investigate in depth in this thesis.

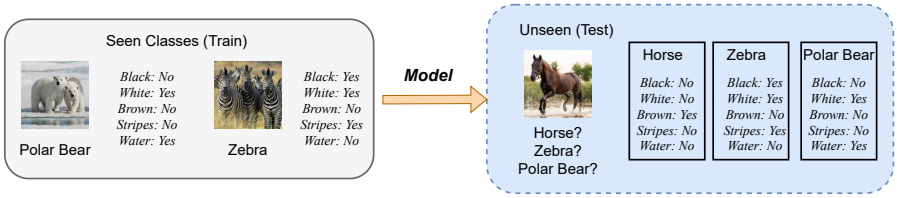


Figure 1.5 – Illustration of *zero-shot learning*. Based on knowledge from the mapping from attributes to classes, zero-shot learning is able to predict the presence of previously unseen classes. We investigate combining this characteristics with continual learning in this thesis.

The first area of investigation is zero-shot learning (ZSL) [188,189,203] (as shown in Figure 1.5), where semantic concepts are represented by meaningful feature vectors instead of one-hot labels (i.e. all classes are equally similar and dissimilar to each other). Zero-shot learning enables the recognition of (visually) unseen classes via a semantic model that describes them in connection to the seen classes. We can also observe that zero-shot learning has an implicit temporal static structure, which means that the class descriptions are learned first then the visual classification model is updated on seen classes thus to obtain the ability to recognize unseen classes. *In this thesis, we aim to investigate systems that combine the capabilities of zero-shot learning and can predict unseen classes, with the capabilities of continual learning, i.e., learn new classes without forgetting previous ones.*

A second application of multi-modal learning that we investigate in more depth is cross-modal retrieval [28,31,38,176,180]. In continual learning, new tasks can happen at different points in time. In a retrieval scenario, we must also consider the indexing operation, thus special attention should be given to the role of this additional stage. One of the advantages of embedding networks compared to classification networks is that the operation is done in a single space shared by all tasks, so we can retrieve data regardless whether the setup is task-aware or task-agnostic. Also, the retrieval performance in the continual learning scenario is affected by how much the embedded space is distorted. Finally, catastrophic forgetting affects differently to indexed and query data. *In this thesis, we aim to investigate the continual learning of cross-modal representations for retrieval.*

1.2 Objectives and approach

In this thesis, we aim to explore the continual learning problem in various computer vision problems, including meta learning, implicit-refined classification, online continual learning and multi-modal tasks. Here we set out to define our objectives and approach to solve the problems identified in the previous section.

1.2.1 Incremental meta learning

Most meta-learning approaches assume the existence of a very large set of labeled data available for episodic meta-learning of base knowledge. This contrasts with the more realistic continual learning paradigm in which data arrives incrementally in the form of tasks containing disjoint classes. In this chapter, we consider this problem of incremental meta-learning in which classes are presented incrementally in discrete tasks. The existing method [99] fails to exploit exemplars to improve performance, we therefore define the following objective:

> **Exemplar usage for incremental meta learning:** Propose a rehearsal-based method for the problem of incremental meta-learning. Carefully study knowledge distillation within the context of rehearsal-based incremental meta learning.

To address the challenges of incremental meta-learning, we propose our method *Episodic Replay Distillation* (ERD) as a replay-based continual learning solution. Different from IDA [99], where the exemplars are only used to compute the class centers, we explore the usage of exemplars to contribute in incremental meta-learning. Firstly, we propose a *cross-task meta-learning loss*, which allows meta-learning to benefit from the larger number of classes stored in the exemplar buffer; Secondly, we propose *Episodic Replay Distillation* that also uses exemplars in distillation to improve the efficiency of knowledge transfer from the previous model to the current one.

1.2.2 The incremental learning of hierarchical knowledge

Human beings learn and accumulate hierarchical knowledge over their lifetime. This knowledge is associated with previous concepts for consolidation and hierarchical construction. However, current incremental learning methods lack the ability to build a concept hierarchy by associating new concepts to old ones. A more realistic setting tackling this problem is referred to as Incremental Implicitly-Refined Classification (IIRC) [1], which simulates the recognition process from

coarse-grained categories to fine-grained categories. However, in their work they evaluate existing methods for this new setting, but do not propose a new method especially designed to acquire hierarchical knowledge incrementally. We therefore define the following objective:

> **Incremental hierarchical knowledge acquisition:** We aim to propose a method that incrementally acquires the hierarchical knowledge from a sequence of tasks. This knowledge should then be exploited to prevent forgetting and improve the prediction capability of the method.

To overcome catastrophic forgetting under the IIRC setup, we propose a new module named Hierarchy-Consistency Verification (HCV), with which we aim to learn the implicit hierarchical knowledge. When learning new classes, we aim to automatically discover their hierarchical relations with the classes seen in previous tasks. When learning new tasks with new superclasses and subclasses, we automatically discover relations among these superclasses and subclasses. We aim to show that this information can be exploited to boost the performance in IIRC setup both at inference time and training time.

1.2.3 Online continual learning with compressed feature replay

Online continual learning aims to learn from a non-IID stream of data from a number of different tasks, where the learner is only allowed to consider data once. Methods are typically allowed to use a limited buffer to store some of the images in the stream. Recently, it was found that feature replay, where an intermediate layer representation of the image is stored (or generated) leads to superior results than image replay, while requiring less memory. Quantized exemplars can further reduce the memory usage. However, a drawback of these methods is that they use a fixed (or very intransigent) backbone network. This significantly limits the learning of representations that can discriminate between all tasks. We therefore pursue the following objective:

> **Compressed feature replay:** The main drawback of feature replay is the lack of plasticity of the network located before the feature replay layer. Therefore, we propose compressed feature replay that allows us to efficiently perform feature replay at deeper layers in the network while still adhering to low memory restrictions.

To address the limitation of feature replay, we propose an auxiliary classifier autoencoder module named ACAE, which allows for compressed feature replay at in-

termediate layers of the network, to enhance the REMIND method [55]. Note that, compared to the current feature replay methods which are focusing on replaying features in the last layers, we allow for more feature replay options. Furthermore, we also addresses an important problem of feature replay methods, namely the sub-optimal performance due to fixing the feature extractor backbone. We evaluate our method under the *online continual learning* setting.

1.2.4 Multi-modal continual learning

In contrast to humans, conventional models of visual recognition assume a static and a semantic world. These models cannot predict classes whose instances were not observed during training. Zero-shot learning (ZSL) and continual learning (CL) relax these two assumptions. However, both these research fields have not yet been combined into a single learning system. We therefore define the following objective:

> **Bookworm continual learning:** We propose a new challenging setup for future AI system, where the semantic model remains fixed while the visual model is updated continuously. In addition, we aim to develop a system that can address this setup. It should continually learn without forgetting previous knowledge, and simultaneously predict unseen classes.

Bookworm continual learning can be seen as a generalization of continual learning which is limited by lacking explicit semantic models, and zero-shot learning which is not continual. One important challenge for bookworm continual learning is the effective integration of semantic models and continual learning. Additionally, we propose a framework via feature generation. Essentially, a generative model (a conditional VAE in our case) learns how to generate features of the previous classes and also future classes (in zero-shot learning via the semantic model), thus generating synthetic features of all classes to train a joint classifier.

The second topic of multi-modal learning that we investigate is cross-modal retrieval. We focus on cross-modal retrieval between language and visual representations. Multimodal representations and continual learning are two areas closely related to human intelligence. The former considers the learning of shared representation spaces where information from different modalities can be compared and integrated. The latter studies how to prevent forgetting a previously learned task when learning a new one. While humans excel in these two aspects, deep neural networks still have clear limitations. We therefore propose the following objective:

> **Cross-modal continual learning:** We aim to investigate cross-modal retrieval. The aim is to effectively perform retrieval in known and unknown

domains. Special attention will be dedicated to studying catastrophic forgetting in this setup.

For cross-modal continual learning, we identify and study the different factors that lead to forgetting in cross-modal embeddings and retrieval. Addressing those factors, we study modifications in the retrieval framework, network architecture and regularization that can help to alleviate them.

2 Incremental Meta-Learning via Episodic Replay Distillation for Few-Shot Image Recognition*

2.1 Introduction

Meta-learning, commonly referred to as "learning to learn", is a learning paradigm in which a model gains experience over a sequence of learning episodes.† This experience is optimized so as to improve the model's future learning performance on unseen tasks [13, 43, 61, 127, 161, 165, 169]. Meta-learning is one of the most widely applied techniques to achieve models that generalize, like humans, to new tasks and environments that have not been seen during training – a capability generally considered to be crucial for future AI systems. Few-shot learning, which aims to learn from very limited numbers of samples, has emerged as the paradigm-of-choice to test and evaluate meta-learning algorithms. Meta-learning applied to few-shot image recognition problems has attracted increased attention in recent years [16, 30, 49, 90, 91, 98, 163, 193].

However, most few-shot learning methods are limited in that they require a large pool of training data, with a large number of classes and a large number of samples per class, for meta-learning. This can lead to poor performance in practical incremental learning situations where the training tasks arrive in a continual way and there are insufficient categories at any single time to learn a performant and general model via meta-learning. The study of learning from data that arrives in such a sequential manner is called *continual* or *incremental learning* [33, 76, 142, 152, 197]. Catastrophic forgetting is the main challenge to building incremental learning systems [117]. This refers to the phenomenon in which knowledge from previous tasks is forgotten when updating the learner with knowledge from new ones. To address both the challenges of incremental and meta-learning, *incremental meta-learning* (illustrated in Figure 2.1) was recently proposed as a way of performing few-shot learning in such incremental learning scenarios [99].

To address the incremental meta-learning problem, Liu et al. [99] proposed the Indirect Discriminant Alignment (IDA) approach. In this method, class centers from previous tasks are represented by *anchors*, which are used to distill knowledge

*This chapter is based on a publication in the 3rd CLVISION workshop in CVPR 2022 [173]

†To avoid ambiguities, we use the term *episode* in the meta-learning sense rather than how it is used in continual learning. We use *task* in the continual learning sense to refer to a disjoint group of new classes.

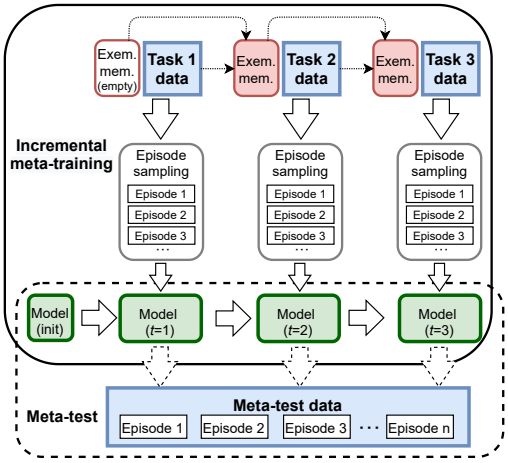


Figure 2.1 – Incremental meta-learning with optional exemplar memories [99]. Data from the previous tasks, unless in the exemplar memory, is unavailable in successive ones. Conventional meta-learning assumes a large number of base classes available for episodic training, while *incremental* meta-learning requires that the meta-learner updates incrementally when a new set of classes (a new *task*) arrives.

from previous models. They show that this greatly reduces forgetting for short sequences of tasks. They proposed EIML, an extended version of IDA with exemplars[‡] from old tasks. But their results surprisingly showed that EIML fails to outperform IDA without exemplars. This seems counter-intuitive, since exploiting exemplars generally boosts performance in incremental learning [17, 113, 142]. The probable reason for this is that EIML only uses exemplars to recompute the class centers of previous tasks, which are then used for distillation.

To perform incremental meta-learning a number of challenges must be addressed: (i) due to the incremental nature of the learning process, knowledge from previous tasks (often represented by class centers) suffers from semantic drift and consequently catastrophic forgetting; (ii) meta-learning benefits from training on a large variety of classes (from which episodes are sampled) to learn representations that can generalize to unseen problems [16, 90, 163], which can be problematic during incremental learning where we only have access to classes present in the

[‡]Exemplars refer to a small buffer of data from previous tasks that can be used during the training of new tasks. It is one of the most effective strategies to counter catastrophic forgetting.

current task; and (iii) when performing knowledge distillation for incremental meta-learning, Liu et al. [99] do not exploit exemplars to perform distillation. In this chapter, we observe that exemplars can also be used to increase the efficiency of the knowledge transfer.

To address the challenges of incremental meta-learning, we propose a replay-based method to address incremental meta-learning. We call our method *Episodic Replay Distillation* (ERD). We use a small exemplar memory from previous classes to prevent catastrophic forgetting. Differently from the descussed work [99] that only uses exemplars for the computation of class prototypes, we show that exemplars can be used in two other steps during incremental meta-learning. Namely, to address the reduced efficiency of meta-learning on datasets with fewer classes (as would be the case when we only perform meta-learning on the classes within a single task) we propose a *cross-task meta-learning loss* which allows meta-learning to directly benefit from the larger number of classes which are present in the exemplar memory. In addition, to improve the efficiency of knowledge distillation we propose *Episodic Replay Distillation* that also uses the exemplars to distill information from the previous task model to the current one.

The main contributions of this chapter are:

- **Cross-task meta-learning**: we apply a *cross-task meta loss* which explicitly uses the exemplars during the meta-learning. This loss results in higher quality feature representations and better generalization to new few-shot recognition problems.

- **Episodic replay distillation**: we exploit the exemplars to efficiently transfer the knowledge from the previous to the current model. Since the exemplars are closer to the class prototypes of previous tasks this results in more efficient knowledge distillation. We are the first to show how exemplar replay can be used for incremental meta-learning, as the previous attempt at this only showed marginal improvements with exemplars [137].

- **Experimental evaluation**: we are the first to evaluate incremental meta-learning on long task sequences (evaluation is increased from just 3 tasks in [137] to 16 tasks in our work). Our method significantly outperforms the state-of-the-art using both Prototypical Networks and Relation Networks.

2.2 Related work

In this section we briefly review the work from the literature most related to our proposed approach.

2.2.1 Few-shot learning

Few-shot learning can be categorized into three main classes of approaches according to which aspect is enhanced using prior knowledge: data augmentation, model enhancement and algorithm-based methods [179]. Among them, few-shot learning based on metrics or optimization-based approaches are the main streams in current research.

Metric-based methods. These approaches use embeddings learned from other tasks as prior knowledge to constrain the hypothesis space. Since samples are projected into an embedding subspace, the similar and dissimilar samples can be easily discriminated. Among these techniques, ProtoNets [161], RelationNets [165], MatchNets [169] and TADAM [131] are the most popular. ProtoNets [161] computes the prototypes for each class in the support set and classify the query images by the nearest-centroid method. RelationNets [165] generalizes this framework by introducing a relation module to train the feature representations. MatchNets [169] adopts memory module and attention mechanism to merge the information in each task. TADAM [131] proposes to use a task conditioned metric which leads to different metric spaces for different tasks.

Optimization-based methods. These use prior knowledge to search for the model parameters which best approximate the hypothesis in search space and use prior knowledge to alter the search strategy by providing good initialization or guiding optimization steps. Representative methods are MAML [43], MAML++ [12], Reptile [127] and MetaOptNet [87]. MAML [43] models the weights of the network as a function of the initial network weights. Based on MAML, MAML++ [12] introduces various modifications to improve the stabilization, convergence speed and computation. Reptile [127] improves MAML by ignoring second-order derivatives. MetaOptNet [87] learns the feature representations generalizing well for SVM classifiers.

2.2.2 Continual learning

Continual learning methods can be divided into three main categories [33]: replay-based, regularization-based and parameter-isolation methods. Since parameter-isolation methods are restricted to the task-aware settings [33], we only discuss the first two categories which are relevant to our method.

Replay methods. These prevent forgetting by including data (real or synthetic) from previous tasks, stored either in an episodic memory or via a generative model. There are two main strategies: exemplar replay [27, 63, 142, 184] and pseudo-replay [155, 183]. The former store a small number of training samples (called exemplars) from previous tasks. The latter uses generative models learned from pre-

vious data distributions to synthesize data. However, pseudo-replay by generating high-resolution images with generative model is itself a very hard problem [155,183] since it requires complex generative models which would also need to be continually learned. An alternative variant is feature replay [55, 106]. However, feature replay imposes limitations on updating backbone models, which is unacceptable for few-shot learning since the purpose of few-shot learning is to learn a good backbone for future updates. Thus, for the Incremental Meta-Learning problem, conducting exemplar replay is a more reasonable choice.

The classification model in continual learning is a joint classifier, the exemplars are used to correct the bias [184] or regularize the gradients [108]. However, in Incremental Meta-Learning there is no joint classifier (only a temporary classifier for each episode). Thus, those exemplar-based continual learning methods need adaptation to the Incremental Meta-Learning. Replay for Incremental Meta-Learning should be at the episode level instead of the image level.

Regularization-based methods. These approaches add a regularization term to the loss function which impedes changes to the parameters deemed relevant to previous tasks. The difference depends on how to estimate relevance, and these methods can be further divided into data-focused [94] and prior-focused [76]. Data-focused methods use knowledge distillation from previously-learned models. Prior-focused methods estimate the importance of model parameters as a prior for the new model.

Distillation methods in continual learning are trying either to align the outputs at the feature level [106, 183] or the predicted probabilities after a softmax layer [94]. However, aligning at the feature level has been observed to be not ineffective [99] and the lack of a unified classifier makes it impossible to align in probabilities level. Thus, we also must adapt distillation to the Incremental Meta-Learning setting.

2.2.3 Meta-learning for continual learning

In addition to the Incremental Meta-Learning setting, there are a few works on continual learning that exploit meta-learning, such as La-MAML [51], iTAML [140] and OSAKA [21]. These methods focus on improving model performance on task-agnostic incremental classification. There is also some work focusing on dynamic, few-shot visual recognition systems [46, 143, 195], which aim to learn novel categories from only a few training samples while at the same time not forgetting the base categories. This setting can be regarded as a variant of few-shot learning where the objective is to maintain good performance on the original base categories when acquiring new ones.

Another related setting is FSCIL [2,116,166,200], where the authors constrain the

continual learning tasks using a few labeled samples (excluding the base task, which has numerous classes and abundant images to enable learning a strong pretrained model). However, this setting concentrates mostly on learning a task-agnostic joint classifier for all the classes that have been observed up to the current time step. It is a variant of the incremental learning setting with some constraints on the amount of samples in new tasks.

Different from these variants, the incremental meta-learning setting adopts the original objective of few-shot learning: to make the model generalize to unseen tasks even when training over incremental tasks, where each task contains a significant amount of data (and is therefore more similar to standard class-incremental learning) on which we train our meta-learner. Since the seen classes are increasing, the model should gain more generalization ability instead of over-fitting to the current task. The meta-learning can then perform few-shot classification on unseen classes – something which is not considered in few-shot class-incremental learning (FSCIL). And since conventional continual learning methods are not suitable for incremental meta-learning, we propose our approach specifically for this setting.

2.3 Methodology

Future learning systems will aim to continually integrate new tasks without requiring joint training over all previously seen data [76, 167]. Specifically the combination of incremental learning with meta-learning is relevant, since at test-time new problems with unseen classes are evaluated. Therefore it is important to develop incremental learning theory on how this new information can be absorbed by the learner to further improve its performance on future tasks. Furthermore, incremental learning does not require the learner to be trained from scratch every time new data arrives (which is also more sustainable), and it can be applied in settings where it is prohibited to retain all past data due to privacy concerns or governmental legislation. A practical example of incremental meta-learning is a robot which must continue to function – with minimal labelling effort – in new scenarios where it must manipulate previously unseen objects. At the same time, it should incrementally improve its model to increase performance in future scenarios. Other scenarios include Lifelong Person Re-identification [137], and incremental few-shot drug discovery [9].

In this section, we start by defining the standard few-shot learning formulation and then introduce the *incremental meta-learning* setup. Then in sections 2.3.2 and 2.3.3 we describe our approach to Incremental Meta-Learning and its application to few-shot image recognition.

2.3.1 Few-shot and meta-learning

In this section we first introduce the standard formulation of few-shot learning, and then describe the incremental meta-learning approach as applied to few-shot classification.

Conventional few-shot learning. An approach to standard, non-incremental classification is to learn a parametric approximation $p(y|x;\theta)$ of the posterior distribution of the class y given the input x. Such models are trained by minimizing a loss function over a dataset D (e.g. the empirical risk). Few-shot learning, however, presents extra difficulties since the number of samples available for each class y is very small (as few as one). In the meta-learning paradigm, training is divided into two phases: meta-training, in which the model learns how to learn few-shot recognition, and meta-testing where the meta-trained model is evaluated on unseen few-shot recognition tasks.

Meta-training for few-shot learning consists of H *episodes* (meta-training task in few-shot learning terminology), where each episode D^τ is drawn from the train split of the entire dataset. Few-shot recognition problems consisting of N classes with K training samples per class are referred to N-way, K-shot recognition problems. Each episode is divided into support set S and query set Q: $D^\tau = (S, Q)$, where $S = \{(x_i, y_i)\}_{i=1}^{NK}$ consists of N training classes each with K images, and $Q = \{(\hat{x}_i, \hat{y}_i)\}_{i=1}^{NK^Q}$ is a set of K^Q images for each of the N selected classes in the episode.

More specifically, we formulate our method based on *ProtoNets* in this section, and discuss its extension to Relation Networks later. ProtoNets consist of an embedding module f_θ and a classifier module g. First, the support set S is fed into the embedding module f_θ to obtain class prototypes $\mathbf{c}_k$:

$$\mathbf{c}_k = \frac{1}{K} \sum_{(x_i,k)\in S} f_\theta(x_i). \tag{2.1}$$

Then, an episode-specific classifier is applied to the query set, where the prediction for class k of query image $\hat{x}$ is:

$$g_k(f_\theta(S), f_\theta(\hat{x})) = p(y = k|\hat{x};\theta) = \frac{\exp(-d(f_\theta(\hat{x}), \mathbf{c}_k))}{\sum_{k'} \exp(-d(f_\theta(\hat{x}), \mathbf{c}_{k'}))} \tag{2.2}$$

where the summation in the denominator is over all classes k' in the support set and d is the Euclidean distance as in ProtoNets. Then the meta-loss for updating θ is:

$$L_{meta}(\theta; S, Q) = -\sum_{(\hat{x},\hat{y})\in Q} [\log g_{\hat{y}}(f_\theta(S), f_\theta(\hat{x}))]. \tag{2.3}$$

Incremental Meta-Learning. When performing incremental meta-learning, data arrives as a sequence of disjoint tasks: $X_1, ..., X_t, ..., X_T$, where T denotes the number of tasks , and t the current training session. The aim of Incremental Meta-Learning is to incrementally learn the parameters θ_t for task t from the disjoint tasks:

$$\theta_t^* = \arg\min_{\theta_t} L(\theta_t; \theta_{t-1}, S_t, Q_t), \tag{2.4}$$

Depending on whether we store exemplars from previous tasks, the support set S_t and query set Q_t can be constructed differently using samples in the current task and exemplars from previous tasks. These query and support sets are described in detail in the next section.

2.3.2 Cross-task episodic training

Keeping exemplars from previous tasks is a successful approach to avoid catastrophic forgetting in conventional incremental learning [17, 113, 142]. However, it is not obvious how to leverage exemplars for incremental *meta*-learning. We propose a novel way of using exemplars for this specific setup.

To fully exploit exemplars E_t from previous tasks, for each episode during meta-learning we construct two sets of support and query images (see Figure 2.2-(a)). Each episode is broken down into two sets of few-shot problems:

- In each episode we construct a *cross-task sub-episode* by sampling N classes from the current task with probability $1-P$ and from previous tasks with probability P. It means for each of the N classes in the episode, a Bernoulli trial with probability P determines if the class is drawn from a past task. Thus, we have on average $N \times (1-P)$ classes from the current task and $N \times P$ from the past. Then, for each class we randomly sample K images as support set S^m and K_Q images as query set Q^m (m denotes that we *mix* the exemplars with current task samples here).
- We also construct an *exemplar sub-episode* by sampling N classes from only the exemplars from previous tasks, each with $K+K_Q$ images to form a support set S^e and query set Q^e. Note that this episode is only composed of *exemplars* from previous tasks.

The reason we sample cross-task sub-episodes with probability P is that exemplars are normally much fewer than samples in the current task, and thus the exemplars are not expected to be as varied as the samples from the current task. With a probability P, we can control the balance between current and previous classes in the cross-task sub-episode. And it doesn't influence the update of the memory buffer.

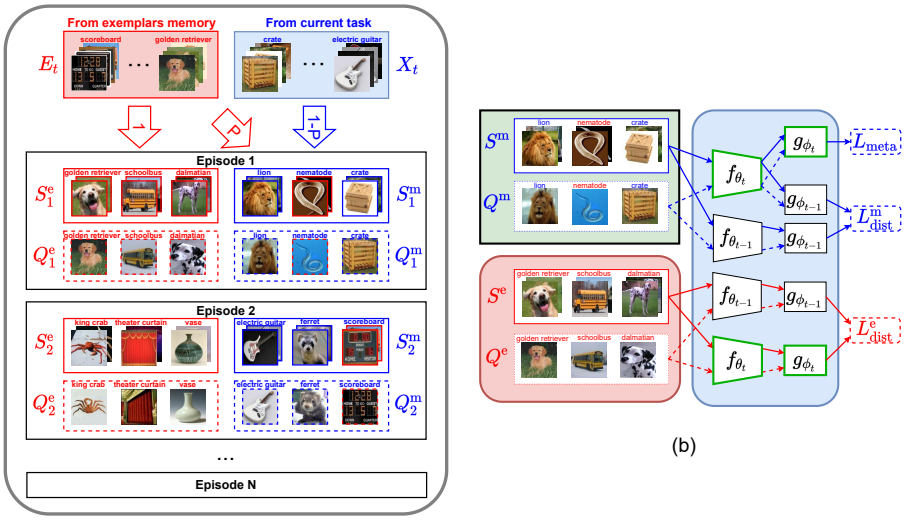


Figure 2.2 – (a) Proposed episode sampling. During episodic meta-learning we build two sets of few-shot problems: *exemplar sub-episodes* based on only exemplars from previous tasks (S_i^e and Q_i^e) and *cross-task sub-episodes* with a mix of exemplars from previous tasks and samples from the current task (S_i^m and Q_i^m). (b) Proposed Episodic Replay Distillation framework. Modules in green are the current embedding model, which are updated with both cross-task and exemplar sub-episodes. Red lines and blue lines are data flows for exemplar sub-episode and cross-task sub-episode, respectively. Solid lines and dotted lines indicate the data flows from support set and query set respectively. When computing loss for ProtoNets, g is a parametric-free operation, while for Relation Networks, g consists of a set of parameters ϕ.

Given S^m, Q^m, the cross-task meta-training loss is defined as:

$$L_{\text{meta}}(\theta_t; S^m, Q^m) = -\sum_{(\hat{x},\hat{y})\in Q^m} \log g_{\hat{y}}(f_{\theta_t}(S^m), f_{\theta_t}(\hat{x})). \tag{2.5}$$

This loss is only computed over S^m and Q^m since in S^m we have samples from the previous and current tasks.

The intra-task meta-loss used in [99] only performs the meta-learning on the data of the current task. The quality of the meta-learner is expected to improve with when learned on wide variety of classes [16, 90, 163]. Only considering the classes within the current task is therefore expected to limit its generalization. In conclusion, we propose to also exploit the replay memory during the meta-learning by performing the meta-learning on both the cross-task and exemplar sub-episodes.

For saving exemplar to E_t, we consider two widely used strategies. The first strategy stores N_{ex} exemplars for each class of each previous task, which is standard in replay-based continual learning methods (UCIR, PODNet, etc.). In this case, the buffer is linearly increased by training sessions. The second strategy fixes the maximum buffer size to M exemplars. We apply both settings in the ablation study and will use the increasing buffer strategy as default.

2.3.3 Episodic Replay Distillation (ERD)

In addition to cross-task episodic training, multiple distillation losses are applied to avoid forgetting when we update the current model (see Figure 2.2-(b)). We first explore distillation using *exemplar sub-episodes*. This is computed as:

$$L^e_{\text{dist}}(\theta_t; \theta_{t-1}, S^e, Q^e) = \sum_{\hat{x}\in Q^e} \text{KL}\big[g(f_{\theta_{t-1}}(S^e), f_{\theta_{t-1}}(\hat{x})) \,||\, g(f_{\theta_t}(S^e), f_{\theta_t}(\hat{x}))\big], \tag{2.6}$$

where $f_{\theta_{t-1}}$ is the embedding network from the previous task with parameters θ_{t-1}. During training, only the current model f_{θ_t} is updated and $f_{\theta_{t-1}}$ is frozen.

Next, similar to Eq. 2.6, we also propose a distillation loss using *cross-task sub-episodes*. It is computed according to:

$$L^m_{\text{dist}}(\theta_t; \theta_{t-1}, S^m, Q^m) = \sum_{\hat{x}\in Q^m} \text{KL}\big[g(f_{\theta_{t-1}}(S^m), f_{\theta_{t-1}}(\hat{x})) \,||\, g(f_{\theta_t}(S^m), f_{\theta_t}(\hat{x}))\big]. \tag{2.7}$$

The only difference between this distillation loss function and Eq. 2.6 is the inputs.

Finally, θ_t is updated by minimizing:

$$L(\theta_t; \theta_{t-1}, S^e, Q^e, S^m, Q^m) = L_{\text{meta}} + \lambda_m L^m_{\text{dist}} + \lambda_e L^e_{dist}, \tag{2.8}$$

where λ_m and λ_e are trade-off parameters.

The proposed distillation strategy, proposed in this section, more efficiently exploits the exemplars for knowledge transfer than the ERD method proposed [99]. ERD performs meta distillation over all classes by exemplar replay, while EIML (the variant of ERD that uses exemplars [99]) only distills the new classes with the old class prototypes. Since the old classes are drifting due to forgetting in EIML and distillation is more efficient when data used for distillation is more similar to previous task data, aligning the new classes (which can be far away in embedding space) to old prototypes further impedes the learning of the meta model. In ERD, however, we also use the exemplars to distill the knowledge (defined by a few-shot problem on both old and new classes). We argue (and will later experimentally verify) that this leads to more efficient knowledge transfer.

We compare between EIML and ERD (with only the cross-task sub-episode) in Figure 2.3. As a summarization, EIML is under-utilizing exemplars only using them to compute the prototype means. ERD also uses the exemplars during the meta-learning as well as for improved distillation. Moreover, we further enhance our ERD model with extra exemplar sub-episodes to improve the results, which is absent in EIML. In addition, a full scheme of our method ERD based on ProtoNets is shown in Algorithm. 1.

2.3.4 Extension to Relation Networks

Episodic Replay Distillation is not limited to ProtoNets. It can also be extended to Relation Networks [165], which consist of a relation module with parameters ϕ. Losses introduced in previous sections are adapted as:

$$L_{\text{meta}}(\theta_t, \phi_t; S^m, Q^m) = \sum_{(x,y)\in S^m, (\hat{x},\hat{y})\in Q^m} \left[g_{\phi_t}(\mathscr{C}(f_{\theta_t}(x), f_{\theta_t}(\hat{x}))) - \mathbf{1}(y=\hat{y}) \right]^2, \tag{2.9}$$

where $\mathscr{C}$ is the concatenation of support set and query set embeddings, $\mathbf{1}$ is the Boolean function returning 1 when its argument is true and 0 otherwise. Distillation losses are updated as:

$$\begin{aligned} L^m_{\text{dist}}(\theta_t, \phi_t; \theta_{t-1}, S^m, Q^m) &= \sum_{x\in S^m, \hat{x}\in Q^m} \left[g_{\phi_{t-1}}(\mathscr{C}(f_{\theta_{t-1}}(x), f_{\theta_{t-1}}(\hat{x}))) - g_{\phi_{t-1}}(\mathscr{C}(f_{\theta_t}(x), f_{\theta_t}(\hat{x}))) \right]^2 \\ L^e_{\text{dist}}(\theta_t, \phi_t; \theta_{t-1}, \phi_{t-1}, S^e, Q^e) &= \sum_{x\in S^e, \hat{x}\in Q^e} \left[g_{\phi_{t-1}}(\mathscr{C}(f_{\theta_{t-1}}(x), f_{\theta_{t-1}}(\hat{x}))) - g_{\phi_t}(\mathscr{C}(f_{\theta_t}(x), f_{\theta_t}(\hat{x}))) \right]^2. \end{aligned} \tag{2.10}$$

Although Relation Networks and ProtoNets adopt different ways to calculate the prediction probabilities for given query images, they share similar network architectures with embedding and classification modules. This type of architecture

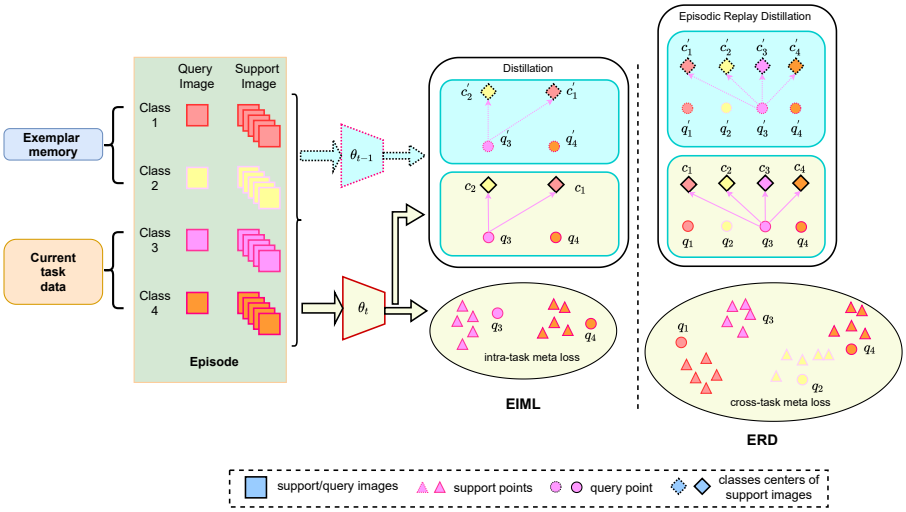


Figure 2.3 – Illustration of novelty of our method ERD with respect to previous state-of-the-art method EIML [99]. We sample an episode that is passed through the previous model θ_{t-1} and current model θ_t. EIML only uses exemplars to compute prototypes used during distillation. While, in ERD, we additionally use exemplars to improve meta-learning by using a cross-task meta-learning loss, and we also apply them in our improved knowledge distillation approach called Episodic Replay Distillation.

is widely used in metric-based few-shot learning and we believe that our method can be easily adapted to other methods with similar architectures.

2.4 Experimental Results

In this section we report on a range of experiments to quantify the contribution of each element of the proposed approach and to compare our performance against the state-of-the-art in continual few-shot image classification.

2.4.1 Experimental setup

Here we describe the datasets and experimental protocols used in our experiments.

Datasets. We evaluate performance on four datasets: Mini-ImageNet [169], CI-

Algorithm 1: Episodic Replay Distillation based on ProtoNets

Input : New task data X_t, exemplar memory E_{t-1}, old model $f_{\theta_{t-1}}$.
Output : New model f_{θ_t}, new exemplar set E_t

Initialize new model: $f_{\theta_t} \leftarrow f_{\theta_{t-1}}$.
Sample exemplar sub-episodes from E_{t-1} and cross-task episodes from X_t and E_{t-1} (Figure 2.2a).
for *all sampled cross-task and exemplar sub-episodes* **do**
    Compute L_{meta} (Eq. 2.5) and L^m_{dist} (Eq. 2.7) from cross-task sub-episode.
    Compute L^e_{dist} (Eq. 2.6) from exemplar sub-episode.
    Update f_{θ_t} using final loss: $L = L_{\text{meta}} + \lambda_m L^m_{\text{dist}} + \lambda_e L^e_{dist}$ (Eq. 2.8)
end
Select new exemplars from X_t using nearest to center criterion and merge them with E_{t-1} to form E_t.
while $|E_t| > M$ **do**
    Randomly remove exemplars from the class with the most.
end
return f_{θ_t}, E_t

FAR100 [81], CUB-200-2011 [170] and Tiered-ImageNet [144]. Mini-ImageNet consists of 600 84×84 images from 100 classes. We propose a split with 20 of these classes as meta-test set unseen during training sessions. The other 80 classes are used to form the incremental meta-training set which is split into 4 or 16 tasks with equal numbers of classes for incremental meta-learning. Each class in each task is then divided into a meta-training split with 500 images, from which support and query sets are sampled for each episode, and a test split with 100 images that is set aside for task-specific evaluation. We select $N_{ex} = 20$ exemplars per class before proceeding to the next task.

	IML training tasks				Meta-test Images
Task #:	1	2	...	16	
Classes per task:	5	5		5	20
Images in train split:	500	500	...	500	600
Images in test split:	100	100		100	

Table 2.1 – The proposed 16-task split of Mini-ImageNet and CIFAR100 datasets for incremental few-shot learning.

CIFAR100 also contains 100 classes, each with 600 images, so we use the same splitting criteria as for Mini-ImageNet. The CUB dataset contains 11,788 images of 200 birds species. We split 160 classes into an incremental meta-training set and the other 40 are kept as a meta-test set of unseen classes. We divide the 160 classes into 4 or 16 equal incremental meta-learning tasks. Since there are fewer images per class, we choose $N_{ex} = 10$ images per class as exemplars for each previous task and 20 images as test split for each class in each task. On Tiered-ImageNet, We keep the same test split (8 categories, 160 classes) as in the original setup, then split the training and validation classes (26 categories, 448 classes) into 16 equally-sized tasks. We select $N_{ex} = 20$ exemplars for each class and 300 images per class as the test split. Since in this case each task contains more classes than in the other datasets, we only test it in the 16-task scenario.

Implementation details. We use ProtoNets as our main meta-learner, but also validate ERD using Relation Networks. We evaluate both the 4-Conv [161] and ResNet-12 [57] backbones as feature extractors. We sample $H = 200$ (4-task) or $H = 50$ (16-task) episodes per task in each training epoch. We train each meta-learning task for 200 epochs using Adam [75] with a learning rate as 0.001.

We evaluate on two widely used few-shot learning scenarios: 1-shot/5-way and 5-shot/5-way. We include results on both incremental training tasks and the unseen meta-test set. For each task (including the unseen set), we randomly construct N_{ep} episodes to obtain the final performance of the meta-learner, which is computed as the mean classification accuracy across the N_{ep} episodes. N_{ep} is 10000 for the 4-task and 1000 for the 16-task scenario. For exemplar selection for ProtoNets, we use the Nearest-To-Center (NTC) criterion to select samples closest to the class mean. If the exemplar memory of size M is full, we iteratively remove exemplars from the class with the most exemplars until there remain only M total exemplars. For Relation Networks, since the image embeddings are feature maps instead of feature vectors, we cannot obtain class prototypes and therefore use random selection. By default we set $\lambda_m = \lambda_e = 0.5$ and $P = 0.2$.

All reported results are an average of three runs under one fixed, randomly-generated class order for each dataset[§]. To measure the influence of class orderings, we also conduct experiments on CIFAR100 with 10 random orders and report the average accuracy and variance (see Section 2.4.2, Figure 2.6).

Compared methods. We compare our method with a finetuning baseline (FT), IDA [99], and a variant of IDA with N_{ex} exemplars per class (EIML). The meta-test upper bounds are obtained by jointly training on all training tasks and testing on the unseen meta-test split (i.e. the standard setting in non-incremental, few-shot

[§]We do not report confidence intervals to improve readability of the tables. We found the 95% confidence intervals to be stable and relatively small (ranging from 0.15-0.25%).

learning). We evaluate on two sets of tasks separately for comparison. At each training session, we evaluate on previously *seen* classes as a way of measuring forgetting. This we call *mean accuracy on seen classes*. Performance on the *meta-test set* (all unseen classes) instead measures the generalization ability to new few-shot recognition problems.

2.4.2 Results on long task sequences.

In this section we report on experiments performed on 16-task incremental few-shot learning scenarios.

Ablation on distillation and cross-task meta losses. We start by ablating the two main novel contributions: the use of exemplars in the cross-task meta-loss and our proposed knowledge distillation method. This ablation study is performed on CIFAR100 in the 16-task 1-shot/5-way scenario with 4-Conv as the backbone. We focus on *meta-test accuracy* to compare among variants since this is the most important evaluation metric in incremental meta-learning.

In Table 2.2 we vary the way the distillation and cross-task meta losses are computed. In EIML, the meta loss is computed only with new classes and the distillation loss aligns new classes with old classes. In ERD the meta loss is computed over all classes and the distillation loss aligns all classes. From the comparison among these five variants, we observe that both components contribute to the final performance gain with respect to EIML. The underlying reason is that EIML ignores the importance of mixing old and new classes at incremental meta-training time. In EIML, exemplars are only used to compute *previous* task prototypes, which under-exploits their potential. Furthermore, a proper distillation loss over exemplars is very important. We incrementally construct the classifier using two sub-episodes: the exemplar sub-episode ensures good performance on old classes, and the cross-task sub-episode ensures discrimination of all classes. The ablation results show that the use of both sub-episodes during distillation helps to significantly improve the results on unseen classes.

Comparison with the state-of-the-art. Here we report results on 16-task 1-shot/5-shot 5-way incremental meta-learning on all four datasets. For CIFAR100 and Mini-ImageNet, the first and third rows in Figure 2.4 show mean accuracy on previous tasks. It is clear that we achieve significantly less forgetting compared to other methods. In the second and fourth rows, the meta-test accuracy for ERD increases with more meta-training tasks due to seeing more diverse classes, while for IDA and EIML the performance drops significantly in some training sessions. This might be due to forgetting on previous tasks and overfitting to the current one. Notably, for our method the meta-test accuracy after the last task is much closer to the joint

Datasets: CIFAR100, Learner: ProtoNets, Backbone: 4-Conv									
Exemplar usage				Method	Training sessions				
Meta loss	Distillation data	Old Prototype	Exemplar SE		2	4	8	16	avg
No	No	Yes	No	EIML	39.7	40.2	44.9	42.0	43.1
No	Yes	Yes	No	-	38.9	40.6	43.3	43.7	43.3
Yes	No	Yes	No	-	39.2	42.6	45.4	45.5	45.2
Yes	Yes	Yes	No	ERD	39.6	43.6	49.0	47.6	46.8
Yes	Yes	Yes	Yes	ERD	**40.1**	**45.0**	**50.0**	**50.5**	**48.1**

Table 2.2 – Meta-test accuracy under the 1-shot/5-way 16-task setting. Here we ablate the meta loss and distillation loss. *Exemplar SE* is brief for *Exemplar sub-episode*.

training upper bound, which is learned using all training tasks.

Note also that EIML works much better than IDA after around 4 tasks. In the original IDA paper, the authors report similar results for both IDA and EIML, which might simply be due to only evaluating on very short sequences of two or three tasks. This is likely caused by anchor drift in IDA and the fact that in EIML exemplars could be used to re-calibrate them. In general. All methods work better in the 5-shot evaluation. The underlying reason for this is that 1-shot recognition is more complex than 5-shot.

In Figure 2.5, we show results on Tiered-ImageNet and CUB. For Tiered-ImageNet, the trends are similar to CIFAR100 and Mini-ImageNet, but the performance difference between EIML and ERD is smaller since there are more classes in each task on Tiered-ImageNet. For CUB, we generally see similar trends as in the other three datasets. However, since CUB is a fine-grained dataset, the forgetting in *mean accuracy on seen classes* is not as serious as for the other three coarse-grained datasets. Instead, we observe increasing *mean accuracy on seen classes*, which could be because the new tasks benefit from the accumulated knowledge from the old ones.

Finally, in Figures 2.6a and 2.6b we report the average over 10 random orders on CIFAR100 to show the robustness of our model to changing task order.

2.4.3 Results on short task sequences

We also compare our method with others on short sequences. In Tables 2.3 and 2.4, we first evaluate our model with a 4-Conv backbone on 1-shot/5-shot 5-way few-shot on three datasets. We see that FT suffers from catastrophic forgetting, and that meta-test accuracy drops dramatically and exhibits overfitting to the current task. IDA is not able to improve meta-test accuracy on Mini-ImageNet, but improves performance on CIFAR100 and CUB. As for EIML, with exemplars it shows large

1-shot/5-way ***4-task*** setting												
Learner:	ProtoNets											
Dataset:	Mini-ImageNet				CIFAR100				CUB			
Backbone:	4-Conv											
	Upper bound: 53.2				Upper bound: 55.4				Upper bound: 61.1			
Sessions:	1	2	3	4	1	2	3	4	1	2	3	4
FT	43.8	44.1	42.4	37.9	44.6	45.1	48.0	45.5	45.1	54.6	54.9	58.8
IDA	43.8	48.3	47.2	42.3	44.6	48.0	51.3	47.6	45.1	**54.7**	54.9	58.7
EIML	43.8	48.8	49.4	47.5	44.6	48.0	52.0	51.7	45.1	53.4	55.0	58.9
ERD	43.8	**51.1**	**52.3**	**53.0**	44.6	**49.5**	**53.6**	**55.1**	45.1	53.9	**58.3**	**60.8**
Backbone:	ResNet-12											
	Upper bound: 59.9				Upper bound: 61.8				Upper bound: 74.8			
Sessions:	1	2	3	4	1	2	3	4	1	2	3	4
FT	45.7	45.9	42.1	37.7	47.0	45.0	51.0	44.6	53.4	64.0	63.7	66.8
IDA	45.7	53.0	53.7	47.6	47.0	53.6	59.2	54.8	53.4	64.4	68.8	73.3
EIML	45.7	53.2	56.5	55.8	47.0	53.3	58.3	57.8	53.4	62.8	69.1	73.3
ERD	45.7	**55.2**	**58.2**	**59.3**	47.0	**55.6**	**61.3**	**61.4**	53.4	**66.1**	**72.4**	**74.1**

Table 2.3 – Meta-test accuracy by training session in the 4-task setting. We evaluate *1-shot* and *5-way* few-shot recognition on three datasets using two different backbones.

improvement compared to IDA. However, our method ERD outperforms EIML by a large margin after learning all four tasks. These results further confirm the observations on the 16-task setting. ERD not only achieves the best performance with less forgetting, but also gets closer to the upper bound after the last task. Note also that CIFAR100 and Mini-ImageNet are coarse-grained datasets, compared to CUB, which makes few shot classification much harder due to intra-class variability.

Finally, we consider ResNet-12 as a backbone to show that ERD can be applied to different network architectures. Our method achieves consistently better performance over others with much higher accuracy than using the 4-Conv backbone.

2.4.4 Comparison with standard Continual Learning methods

In Figure 2.7, we compare our method with three state-of-the-art CL methods: iCaRL [142], PODNet [37] and UCIR [63]. For the evaluation on seen classes, we follow the same protocol as IDA where the average classification accuracy is calculated over N_{ep} episodes. Note that these methods were not designed for incremental meta-learning and cannot be directly applied to this scenario. To adapt these method to incremental meta-learning, we use them to continually learn representations and then evaluate them with a nearest-centroid classifier for few-shot learning. Observe how on seen classes UCIR works better than iCaRL and PODNet, however

5-shot/5-way ***4-task*** setting												
Learner:	ProtoNets											
Dataset:	Mini-ImageNet				CIFAR100				CUB			
Backbone:	4-Conv											
	Upper bound: 75.1				Upper bound: 76.5				Upper bound: 82.5			
Sessions:	1	2	3	4	1	2	3	4	1	2	3	4
FT	63.4	64.1	65.2	62.1	67.0	68.2	71.2	67.5	69.4	73.4	74.5	76.4
IDA	63.4	68.5	68.1	66.0	67.0	70.3	72.8	69.6	69.4	75.4	76.0	78.6
EIML	63.4	69.1	70.3	70.2	67.0	70.7	73.6	72.7	69.4	75.2	78.2	79.0
ERD	63.4	**69.4**	**71.4**	**72.2**	67.0	**71.2**	**74.4**	**73.9**	69.4	**75.9**	**78.7**	**80.4**
Backbone:	ResNet-12											
	Upper bound: 81.9				Upper bound: 81.0				Upper bound: 91.2			
Sessions:	1	2	3	4	1	2	3	4	1	2	3	4
FT	66.2	65.6	62.7	61.4	69.1	68.5	70.4	66.4	76.3	80.7	80.3	84.2
IDA	66.2	73.1	75.5	74.5	69.1	75.5	77.9	78.8	76.3	81.8	84.4	86.7
EIML	66.2	74.6	77.5	78.3	69.1	76.8	78.6	80.3	76.3	83.2	86.1	88.3
ERD	66.2	**74.7**	**77.7**	**80.0**	69.1	**77.2**	**79.4**	**80.8**	76.3	**83.4**	**86.6**	**89.6**

Table 2.4 – Meta-test accuracy by training session in the 4-task setting. We evaluate *5-shot* and *5-way* few-shot recognition on three datasets using two different backbones.

Datasets: CIFAR100, Learner: ProtoNets, Backbone: 4-Conv										
Evaluation:	Meta-test accuracy					Mean accuracy on seen classes				
1-shot/5-way ***16-task*** setting										
Sessions:	2	4	8	16	avg	2	4	8	16	avg
FT	37.3	37.6	40.0	34.4	38.1	44.8	44.1	40.9	37.8	42.5
IDA	39.8	39.3	42.2	35.9	40.3	50.6	46.2	44.1	39.6	44.7
EIML	39.7	40.2	44.9	42.0	43.1	51.1	48.9	46.8	46.7	48.6
ERD	**40.1**	**45.0**	**50.0**	**50.5**	**48.1**	51.0	**52.2**	**53.7**	**54.8**	**53.9**
iCaRL	39.0	42.0	43.4	45.2	43.1	50.1	47.2	46.5	48.0	48.5
UCIR	35.1	36.3	39.5	42.2	39.3	**53.6**	50.5	50.1	51.9	52.4
PODNet	36.0	37.0	37.1	36.4	37.0	52.9	43.8	41.0	41.1	44.6
5-shot/5-way ***16-task*** setting										
Sessions:	2	4	8	16	avg	2	4	8	16	avg
FT	53.6	55.7	59.4	50.7	56.1	61.0	59.8	60.1	55.2	60.2
IDA	58.6	60.2	62.3	54.9	59.1	75.2	66.6	64.3	58.7	64.8
EIML	57.3	**64.1**	67.1	67.7	65.2	**76.8**	**73.3**	72.4	70.2	73.0
ERD	**58.7**	63.9	**68.6**	**71.2**	**66.9**	75.7	73.2	**74.1**	**74.6**	**74.9**
iCaRL	52.6	57.3	60.1	62.1	59.1	68.9	65.9	65.8	67.2	67.4
UCIR	45.2	48.7	55.3	60.8	54.4	68.9	68.5	70.7	73.7	71.7
PODNet	48.7	50.2	51.4	50.9	50.7	71.0	59.1	57.1	57.5	60.5

Table 2.5 – Meta-test accuracy and mean accuracy as a function of the number of training sessions on the 16-task setting using ProtoNets as the meta learner. We evaluate on CIFAR-100 to compare with standard continual learning methods.

under meta-test evaluation, iCaRL works the best among the standard CL methods. PODNet performs similarly to the FT baseline in both cases. Our method, that is especially tailored for incremental meta-learning, outperforms the standard CL methods by a large margin – especially for few-shot evaluation on unseen classes, where our 1-shot meta-test accuracy outperforms iCaRL by around 8.5% after 16 tasks.

2.4.5 Additional ablation studies

Here we show ablation studies on CIFAR100 in the 16-task 1-shot/5-way scenario with 4-Conv as the backbone. We report *meta-test accuracy* to compare among variants.

Ablation on P with $\lambda_m = \lambda_e = 0.5$. As shown in Figure 2.8a, ERD obtains the best performance with $P = 0.2$. This is what we use by default for all previous experiments. When $P = 0$, it means there are no previous classes in the *cross-task sub-episode*, which performs worse than our variants with higher probabilities, especially with $P = 0.2$ and $P = 0.4$. As P decreases from 0.6 to 0.2, the performance consistently improves. The reason is that lower P results in more current samples, which can ensure the diversity of the training samples. This phenomenon is different from the conventional use of exemplars in incremental learning, where more balanced exemplar sampling is preferable. We use the notation P = Rand to identify that P is not fixed, but that classes in each cross-task sub-episode are randomly selected from all encountered classes up to now and P is increasing with successive tasks. This achieves worse results because there are more and more previous classes with less diverse samples. Most of our variants outperform EIML by a large margin. We keep $P \geq 0.2$ to ensure that at least one previous class occurs in each episode for 5-way few-shot learning.

Ablation on λ_m and λ_e with $P = 0.2$. To understand the role of each distillation component in Eq. (2.8), we ablate the distillation loss terms. As shown in Figure 2.8b, our method achieves the best results with $\lambda_m = 0.5$ and $\lambda_e = 0.5$, which indicates that both distillation terms play a crucial role in overcoming forgetting and generalizing to unseen tasks. ERD with $\lambda_m = 0.5$, $\lambda_e = 0$ works similarly to ERD with $\lambda_m = 0$, $\lambda_e = 0.5$. They both achieve much better performance than without using distillations ($\lambda_m = 0$ and $\lambda_e = 0$). In Figure 2.8c, we extend the ablation with $\lambda_e = \lambda_m = \{0.0, 0.1, 0.5, 1.0, 2.0, 10.0\}$. We can observe that this hyperparameter gets the best results for {0.5,1.0,2.0}.

Ablation on exemplar selection strategies. To save exemplars after each training session, we need to choose N_{ex} for each class. We performed an ablation to compare nearest-to-center, random selection, herding [142], and a simplified version of

Learner:	Relation Networks											
Datasets:	Mini-ImageNet				CIFAR100				CUB			
Backbone:	4-Conv											
1-shot/5-way ***16-task*** setting												
	Upper bound: 52.0				Upper bound: 59.2				Upper bound: 51.6			
Sessions:	2	4	8	16	2	4	8	16	2	4	8	16
FT	24.4	28.5	25.5	28.7	31.1	30.0	35.2	26.8	37.1	38.1	37.0	34.0
ERD	**27.7**	**29.9**	**34.5**	**30.1**	**35.6**	**39.3**	**45.7**	**35.9**	**37.3**	**42.9**	**47.9**	**42.5**
1-shot/5-way ***4-task*** setting												
	Upper bound: 52.0				Upper bound: 59.2				Upper bound: 51.6			
Sessions:	1	2	3	4	1	2	3	4	1	2	3	4
FT	41.70	41.65	38.51	33.33	42.9	45.3	45.7	42.3	45.7	47.9	48.5	50.3
ERD	41.70	**45.84**	**48.35**	**49.73**	42.9	**48.2**	**51.2**	**51.4**	45.7	**48.5**	**49.4**	**51.4**

Table 2.6 – Meta-test accuracy by training sessions on the 4-task and 16-task settings. We evaluate 1-shot/5-way few-shot recognition on Mini-ImageNet, CIFAR-100 and CUB.

Rainbow Memory [15] (we call RB^*) in Figure 2.8d with $P = 0.2, \lambda = \lambda_m = \lambda_e = 0.5$. For RB^*. Since we have no joint classifier over all classes, which is needed for Rainbow Memory, we imitate the idea by selecting exemplars near the boundary or exemplars near the center of the class with equal probability. This obtains slightly better results, but essentially the exemplar selection strategies differ little in final performance.

Ablation on memory buffer with $P = 0.2, \lambda_m = \lambda_e = 0.5$. In this experiment, we fix other hyper-parameters to show how different numbers of exemplars affect incremental learning performance. We provide results for various N_{ex} and also with a bounded buffer size M which are both commonly used for exemplar replay. From Figure 2.8e we see that increasing N_{ex} leads to a noticeable increase in performance going from 2 to 20 exemplars (note that in EIML increasing the number of exemplars does not influence performance). However, also for ERD the gain is marginal beyond 20 exemplars per class. From Figure 2.8f, we observe that with a smaller bounded buffer with only $M = 500$ exemplars, ERD is still close to the joint training upper bound, showing the importance of proposed sub-episodes.

2.4.6 Extension to Relation Networks

Since in Relation Networks there is no embedding to exploit for computing prototypes as in ProtoNets, IDA and EIML cannot be directly applied. Therefore we only compare with FT in this experiment. As the experimental results shown in Table 2.6, our model not only surpasses the FT baseline significantly, but also gets close to the

Learner	ProtoNets			
Backbone	4-Conv			
Datasets	Mini-ImageNet			
	1-shot 5-way 4-task setting			
	Upper bound: 53.24 ±0.22			
Sessions	1	2	3	4
FT	43.79 ±0.18	44.05 ±0.19	42.44 ±0.21	37.91 ±0.20
IDA	43.79 ±0.18	48.25 ±0.20	47.16 ±0.24	42.33 ±0.23
EIML	43.79 ±0.18	48.79 ±0.19	49.37 ±0.20	47.53 ±0.20
ERD	43.79 ±0.18	51.14 ±0.20	52.27 ±0.21	52.98 ±0.21

Table 2.7 – Meta-test accuracy with confidence intervals under the 1-shot/5-way 4-task setting.

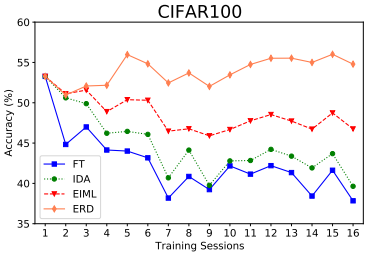
joint training upper bounds after the last task, especially on the CUB dataset.

2.4.7 Confidence intervals

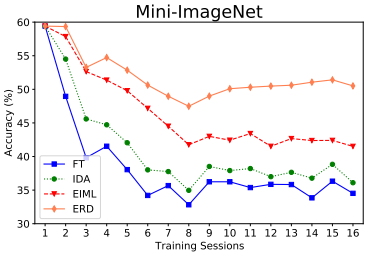
In Table 2.7 we give the meta-test accuracy with confidence intervals on Mini-ImageNet for the 1-shot/5-way/4-task scenario. The confidence intervals are relatively small with respect to average accuracy and almost the same for different methods. Therefore, it is fair to compare different methods mainly based on average accuracy as done.

2.5 Conclusions

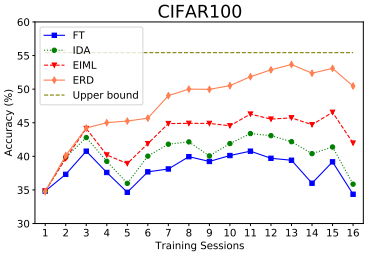
In this chapter we proposed Episodic Replay Distillation, an approach to incremental few-shot recognition. Exemplar replay is one of the most successful methods for incremental learning of classification problems [17, 113, 142]. We are the first to show how this successful tool can be used for incremental meta-learning. The previous attempt only showed marginal improvements with exemplars [137]. Our work shows that also in the incremental meta-learning scenario exemplar replay is a powerful tool for preventing forgetting. We exploit the exemplars to perform cross-task meta-learning which improves the discriminative power of the learned representations. In addition, we also use exemplars to perform our proposed episodic replay distillation. Both contributions are shown to considerably improve performance. Experiments on multiple few-shot learning datasets demonstrate the effectiveness of ERD. Our method is especially effective on long task sequences, where we significantly close the gap between incremental few-shot learning and the joint training upper bound.



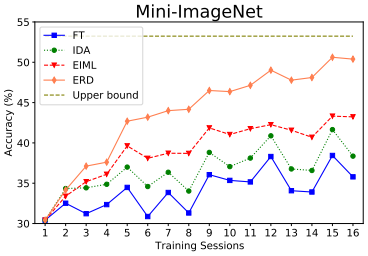
(a) 1-shot mean accuracy on seen



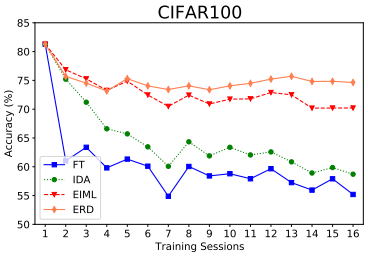
(b) 1-shot mean accuracy on seen



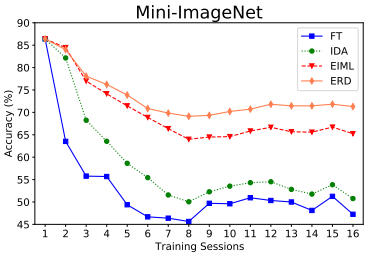
(c) 1-shot meta-test accuracy



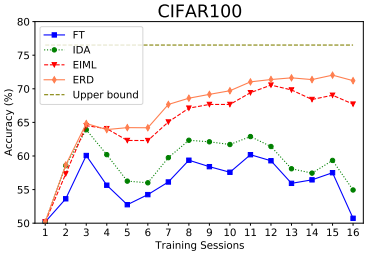
(d) 1-shot meta-test accuracy



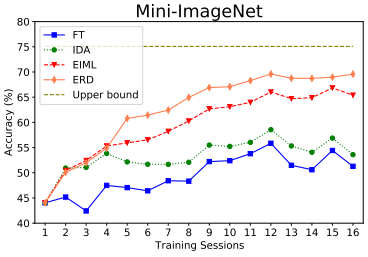
(e) 5-shot mean accuracy on seen



(f) 5-shot mean accuracy on seen

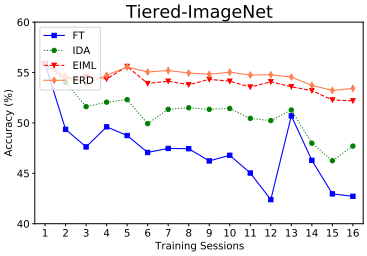


(g) 5-shot meta-test accuracy

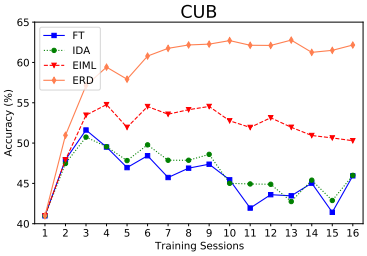


(h) 5-shot meta-test accuracy

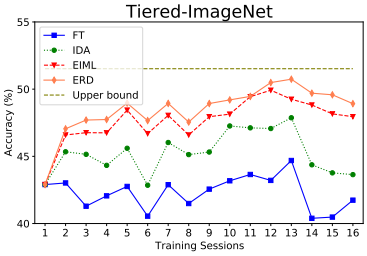
Figure 2.4 – Results on the 1- and 5-shot, 5-way 16-task setup with a 4-Conv backbone and ProtoNets meta-learner. Evaluations are on CIFAR100 and Mini-ImageNet datasets.



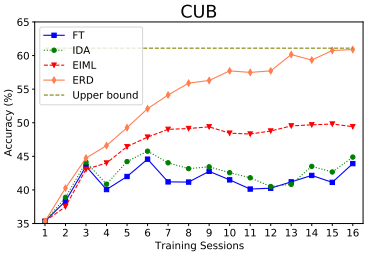
(a) 1-shot mean accuracy on seen



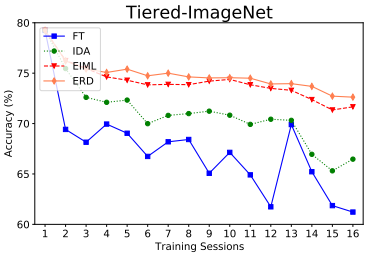
(b) 1-shot mean accuracy on seen



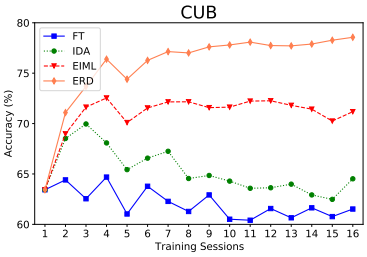
(c) 1-shot meta-test accuracy



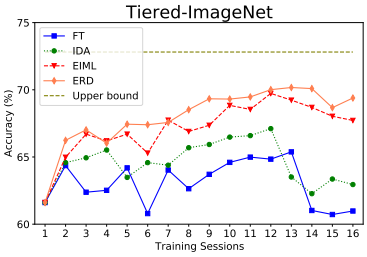
(d) 1-shot meta-test accuracy



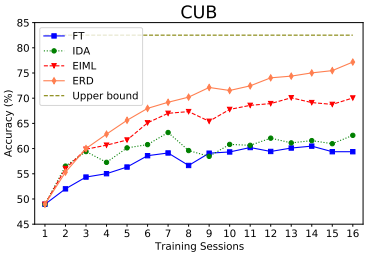
(e) 5-shot mean accuracy on seen



(f) 5-shot mean accuracy on seen

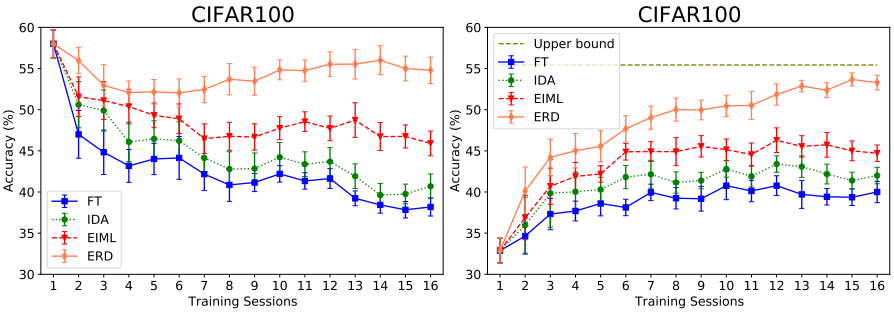


(g) 5-shot meta-test accuracy



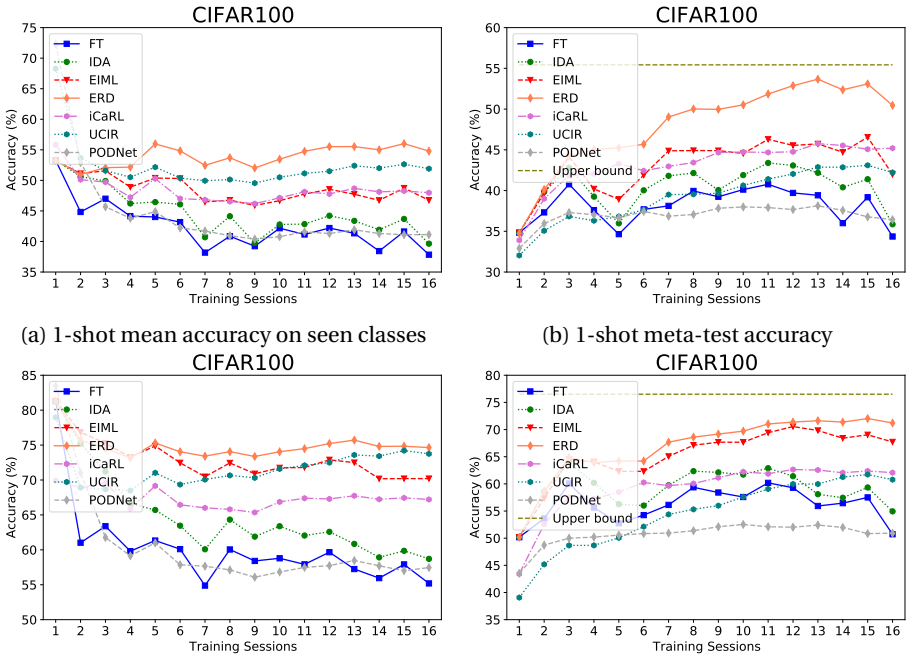
(h) 5-shot meta-test accuracy

Figure 2.5 – Results on the 1- and 5-shot, 5-way 16-task setup with a 4-Conv backbone and ProtoNets meta-learner. Evaluations are on Tiered-ImageNet and CUB datasets.



(a) 1-shot mean Acc. on seen (b) 1-shot meta-test Acc.

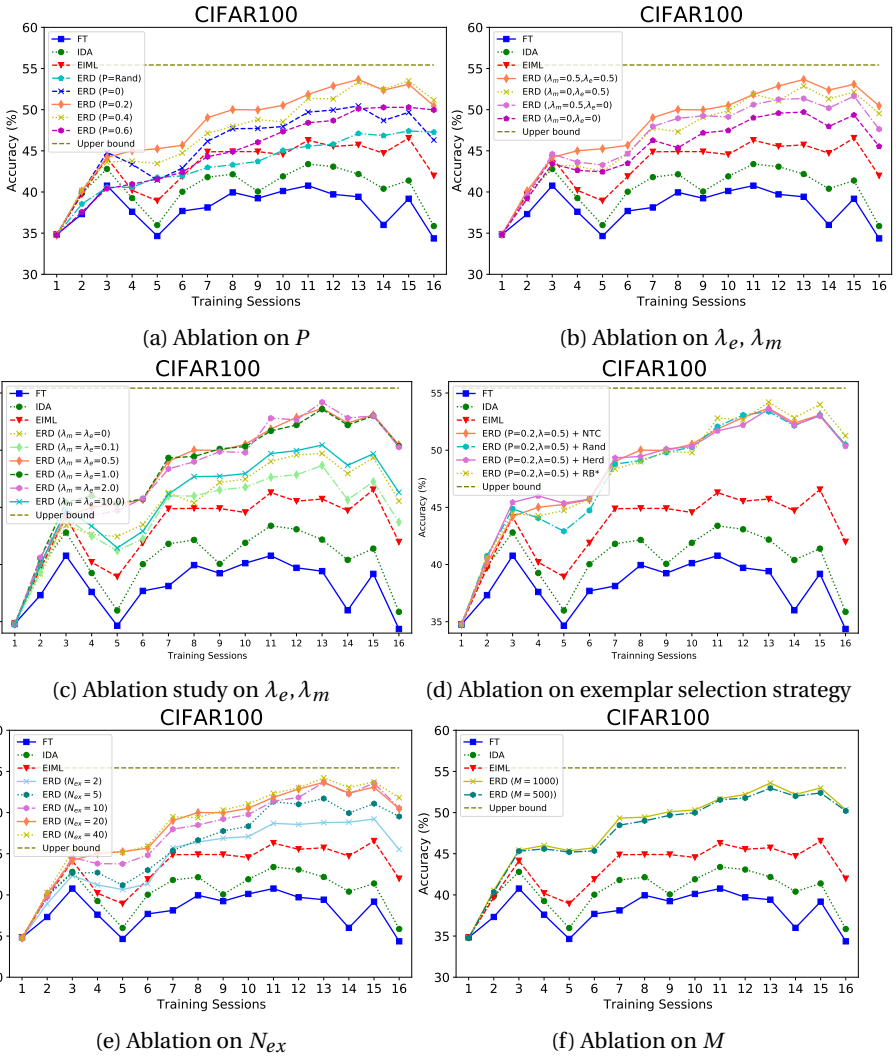
Figure 2.6 – Experimental results with 10 task orderings on CIFAR100.



(a) 1-shot mean accuracy on seen classes (b) 1-shot meta-test accuracy

(c) 5-shot mean accuracy on seen classes (d) 5-shot meta-test accuracy

Figure 2.7 – Comparison with CL methods on 1-shot 5-way 16-task setting with a 4-Conv backbone and ProtoNets meta-learner on CIFAR-100. (Left) Mean accuracy on seen classes. (Right) Meta-test accuracy on the unseen meta-test set.



(a) Ablation on P

(b) Ablation on λ_e, λ_m

tb]

(c) Ablation study on λ_e, λ_m

(d) Ablation on exemplar selection strategy

(e) Ablation on N_{ex}

(f) Ablation on M

Figure 2.8 – Ablation study on 16-task 1-shot/5-way setup on CIFAR100 with 4-Conv. We plot the meta-test accuracy to compare.

3 HCV: Hierarchy-Consistency Verification for Incremental Implicitly-Refined Classification*

3.1 Introduction

In the lifetime of a human being, knowledge is continuously learned and accumulated. However, deep learning models suffer from knowledge forgetting, also known as catastrophic forgetting [76, 117], when presented with a sequence of tasks. Incremental learning [34, 113, 133], also referred to as continual learning, has been a crucial research direction in computer vision that aims to prevent this forgetting of previous knowledge in neural networks.

Another aspect of human learning is the association of new concepts to old concepts, people construct a hierarchy of knowledge to better consolidate this information. Recently, the IIRC (Incremental Implicitly-Refined Classification) setup [1] has been proposed as a novel extended benchmark to evaluate lifelong learning methods in a realistic setting where the construction of hierarchical knowledge is key. On the IIRC benchmark (see Fig. 3.1), each class has multiple granularity levels. But only one label is present at any time, which requires the model to infer whether the related labels have been observed in previous tasks. This setting is much closer to real-life learning, where a learner gradually improves its knowledge of objects (first it labels roses as a plant, later as a flower, and finally a rose).

Based on this benchmark, Abdelsalam et al. [1] adapted and evaluated several state-of-the-art incremental learning methods to address this problem, including iCaRL [142], LUCIR [63], and AGEM [27]. However, their work does not propose an effective solution specifically designed for the IIRC problem. They do not aim to incrementally learn the hierarchical knowledge that is important to correctly label the data in this setting. Furthermore, there are also some other limitations in the current version of the IIRC benchmark: (i) The granularity is limited to two layers, while in reality there are often more layers involved (see WordNet [119] hierarchy of ImageNet [35]). (ii) The first task always contains a large number of superclasses, which means that the learner encounters data from most classes already in these early stages†. This makes training relatively easy, and the proposed

*This chapter is based on a publication in the British Machine and Vision Conference, 2021 [174]

†The actual setup considers 10 superclasses in the first task, meaning that around 50 (of the total 100) subclasses are seen implicitly during the first task.

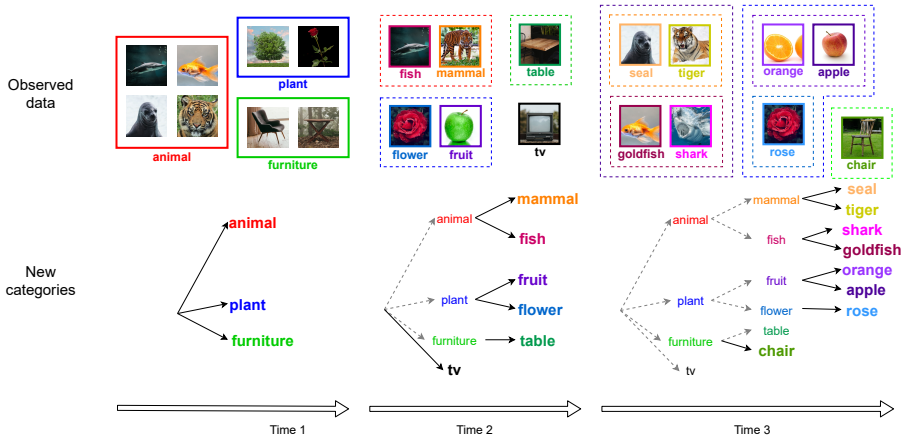


Figure 3.1 – Illustration of 3-layer hierarchy IIRC setting. New categories in each training time are annotated by solid pointers, and the hierarchical relationships among old categories and new categories are denoted with dashed arrows.

setup less applicable.

To overcome catastrophic forgetting under the IIRC setup, we propose a module called Hierarchy-Consistency Verification (HCV). We aim to explicitly learn in an incremental manner the hierarchical knowledge that underlies the data. While learning new tasks with new super and subclasses, we automatically discover relations, e.g. the class 'flower' is a subclass of 'plant'. Next, we show how this knowledge can be exploited to enhance incremental learning. Principally, in the described example, we would not use images from 'flower' as negative examples for the class 'plant' (a problem from which the methods in [1] suffer). Next, we show how the hierarchical knowledge can be used at inference time to improve the predictions. Based on these observations, our main contributions are:

- We propose a Hierarchy-Consistency Verification (HCV) module as a solution to the IIRC setup. It incrementally discovers the hierarchical knowledge underlying the data, and exploits this during both training and inference.
- We extend the IIRC benchmark to a challenging 3-layer hierarchy on the IIRC-CIFAR dataset. In addition, we propose a much harder setup where the superclasses are distributed uniformly over incremental tasks to test the robustness of different methods.
- Experiments show that we successfully acquire hierarchical knowledge, and

that exploiting this knowledge leads to significantly improvements of existing incremental learning methods under the IIRC setup (with absolute accuracy gains of 3-20%).

3.2 Related work

3.2.1 Incremental learning

Incremental learning methods can be categorized into three types [34, 113] as follows.

Regularization-based methods. The first group of techniques add a regularization term to the loss function which impedes changes to the parameters deemed relevant to previous tasks. The difference depends on how to compute the estimation. These methods can be further divided into data-focused [71, 94, 141, 202] and prior-focused [4, 25, 76, 88, 104, 198]. Data-focused methods use knowledge distillation from previously learned models. Prior-focused methods estimate the importance of model parameters as a prior for the new model.

Parameter isolation methods. This family focuses on allocating different model parameters to each task. These models begin with a simplified architecture and updated incrementally with new neurons or network layers in order to allocate additional capacity for new tasks. In Piggyback/PackNet [111, 112], the model learns a separate mask on the weights for each task, whereas in HAT [154] masks are applied to the activations. This method is further developed to the case where no forgetting is allowed in [114]. In general, this branch is restricted to the task-aware (task incremental) setting. Thus, they are more suitable for learning a long sequence of tasks when a task oracle is present.

Replay methods. This type of methods prevent forgetting by including data from previous tasks, stored either in an episodic memory or via a generative model. There are two main strategies: exemplar rehearsal [27, 63, 106, 142, 184] and pseudo-rehearsal [155, 183]. The former stores a small amount of training samples (also called exemplars) from previous tasks. The latter use generative models learned from previous data distributions to synthesize data.

3.2.2 Hierarchical classification and multi-label classification

Classification problem is normally considered that the categories are not overlapped with each other. However, the concepts in real life are connected to each other with hierarchical information. For example, in ImageNet [35], the categories are hierarchized by WordNet [119] knowledge. For hierarchical classification [156],

the system groups things according to an explicit hierarchy, which is important to some applications, such as bioinformatics [44] and COVID-19 identification [135]. Another related area is multi-label classification [204], where each image is related to multiple labels. Multi-label classification is a generalization of the single-label categorizing problem. In the multi-label problem there is no constraint on how many of the classes the instance can be assigned to. While under this setup, there is no hierarchical constraints among categories. By comparison, on the IIRC setup [1], the hierarchical information is implicitly defined. The developed model for this problem should be able to learn this hierarchy by itself and predict the multiple labels for each instance.

3.3 Methodology

The original work that presented the IIRC setup [1] ignores the hierarchical nature of the classes during incremental learning. Consequently, some samples are incorrectly used as negative samples for their superclass labels, potentially resulting in a drop of performance. Here we propose our method to incrementally learn the hierarchy and directly exploit this information to remove said interference. Moreover, we also show how the estimated hierarchy can be exploited at inference time. Our method is general and can be applied to existing methods for incremental learning that can be trained with a binary cross-entropy loss (in experiments we will show results for iCaRL [142], and LUCIR [63]).

3.3.1 IIRC setup

Given a series of tasks, each task $t \in [1, T]$ is composed of data D_t from the current class set C_t which can contain both super- and subclasses. During training of task t the model will receive $(x_t^i, y_t^i) \in D_t^{train}$, $y_t^i \in C_t$ where $y_t^i \in \{u_t^i, v_t^i\}$ is either the subclass u_t^i or the superclass v_t^i label of the i-th sample x_t^i, only one of which is present in C_t. In the proposed setup of [1], always first the superclass is learned and later the subclass (like in Fig. 3.1). We will use lowercase y for a one-hot vector, and capital Y to identify a binary vector possibly with multiple non-zero elements. It is important to note that even if during training only a single label y_t^i is provided, during testing after task t we consider test data $(x_t^i, Y_t^i) \in \cup_{j=1}^{t} D_j^{test}$ where multi-class ground-truth vector Y_t^i contains the subclass and superclass label of sample x_t^i (if these are in $\cup_1^t C_t$[‡]), i.e., at test time we are expected to predict all non-zero elements in Y_t^i.

[‡]Some samples might only have a single label since the subclass label is not yet encountered during training.

To make the common recognition model applied in this multi-class case, in [1] they propose to replace the conventional cross-entropy loss by a binary cross-entropy loss:

$$\mathscr{L}_{BCE} = -\sum_{i}[y_t^i \cdot log(\hat{Y}_t^i) + (1 - y_t^i) \cdot log(1 - \hat{Y}_t^i)] \tag{3.1}$$

where $\hat{Y}_t^i = \mathscr{F}_t(x_t^i)$ is the predicted probability vector of sample x_t^i, with $\mathscr{F}_t$ the current prediction model. They apply this equation to several incremental learning algorithms. However, it should be noted that samples can be wrongly used as a negative sample for their own superclass, because this loss only considers the provided label y_t^i.

We extend the two-layer hierarchy proposed in the original IIRC setup to three layers to verify the effectiveness of our module in more complex scenarios. In this case, each sample contains a three-layer label annotations Y_t^i as: (subclass u_t^i, superclass v_t^i, rootclass w_t^i).

3.3.2 HCV: Hierarchy-Consistency Verification

In the previous section, we discussed that the original solution results in interference during training. The challenge here is that the model should correctly learn the relationship between sub classes u_t^i and super classes v_t^i, given only the y_t^i information during training time. Here, we propose our method that address this problem.

To overcome forgetting under the IIRC setup, we incrementally compute the class hierarchy by estimating the relationship between old and new classes. If a new class is highly related to an old class, we identify it as the subclass of the old class. With this estimated hierarchical knowledge, we verify the hierarchy consistency both during training and inference time to boost the performance of the continual learning models. Our algorithm, called *Hierarchy-Consistency Verification* (HCV), contains two phases which we describe in the following (see also Fig. 3.2). Moreover, the learned hierarchy is also exploited at inference.

Phase I: Learning Hierarchical Relations (LHR).

The mission at this stage is to estimate the existing hierarchical relationship between subclasses u_t^i and superclasses v_t^i. This stage occurs before the training of the current task. Supposing we have learned the classifier $\mathscr{F}_{t-1}$ for all previous classes. We could use $\mathscr{F}_{t-1}$ to classify all accessible data D_t^{train} for class y_c and produce a prediction vector p_{y_c}.

$$p_{y_c} = \frac{1}{N} \cdot \sum_{i|y_t^i = y_c} \mathscr{F}_{t-1}(x_t^i) \quad (x_t^i, y_t^i) \in D_t^{train} \tag{3.2}$$

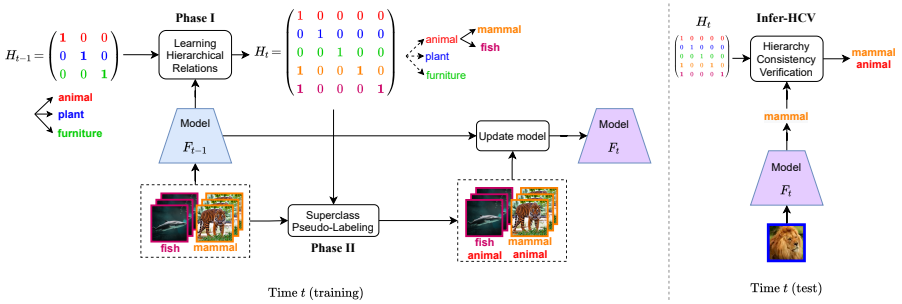


Figure 3.2 – Illustration of our method: Hierarchy-Consistency Verification (HCV). At Phase I, hierarchical relations between subclasses and superclasses H_t are acquired using current data. And then at Phase II, the multi-class labels are generated for each instance. Current model is updated with calibrated labels at training time. The hierarchical relations can be applied during inference time as well to further improve the predictions.

where N is the number of images labeled as $y_c \in C_t$. If the maximum prediction value in p_{y_c} is larger than a threshold τ, we would consider the previous class $\bar{v}_t^i$ with the max probability value is the superclass of class y_t^i. Based on this prior knowledge learned from previous classifiers, we could construct a hierarchical tree H_t, which consists of all hierarchical information up to the current task t.

Phase II: Superclass Pseudo-Labeling (SPL). After learning the superclasses before training task t, we have the hierarchical tree H_t, which contains all estimated hierarchical information up to the current task. Now we can apply this knowledge at both train and test time.

During training time, if a new class is estimated as a subclass of a specific previous superclass, we assign the estimated superclass label $\bar{v}_t^i$ as a *superclass pseudo-label* to the corresponding subclasses label y_t^i (we will use the overline $\bar{\cdot}$ to identify that label is estimated). In this way, the estimated multi-class label $\bar{Y}_t^i$ can be represent as:

$$\bar{Y}_t^i = \begin{cases} y_t^i & \text{if } y_t^i \text{ has no parents in the hierarchical tree } H_t \\ y_t^i \cup \bar{v}_t^i & \text{if } \bar{v}_t^i \text{ is the estimated parent of } y_t^i \end{cases} \tag{3.3}$$

Then, with the new class label vector $\bar{Y}_t^i$, the binary cross-entropy loss is rewritten

as:

$$\mathscr{L}_{BCE} = -\sum_{i}[\bar{Y}_t^i \cdot log(\hat{Y}_t^i) + (1-\bar{Y}_t^i) \cdot log(1-\hat{Y}_t^i)] \tag{3.4}$$

For applying our SPL module to continual learning methods, we simply replacing the orginal BCE loss in Eq. 3.1 with Eq. 3.4.

Inference with HCV (Infer-HCV). At inference time, if a multi-class prediction vector is not consistent with our estimated hierarchical knowledge H, we mark it as a wrong prediction (e.g. it estimates a sub and superclass combination that is not in accordance to our hierarchical knowledge captured by H). Based on this assumption, we process each prediction $\hat{Y}_t^i$ with H_t. If the prediction is in accordance with H_t it remains unchanged. If we need to add labels to $\hat{Y}_t^i$ to make it be in accordance to H_t we do so (add subclass or superclass label). If we need to remove labels from $\hat{Y}_t^i$ to reach accordance with H_t, we randomly select one of the possible solutions containing the least number of removed labels. See Fig. 3.3 for a visual explanation of Infer-HCV. Here we provide some examples for better understanding of Infer-HCV. In Fig. 3.3 there are four examples of how the *Infer-HCV* module works. We address the examples one column at a time:

1. This example is correctly matched by the first row of H_t, so it remains unchanged.
2. This example does not match any row in H_t. We match it with the first row by removing the second label.
3. This example also does not match. We match it by adding the first class label to make it in accordance with the last row of H_t.
4. This example also does not match. It can be modified by removing the 4th or 5th labels, so we randomly choose one from them to make it compatible with H_t.

3.4 Experiments

3.4.1 Experimental setup

Datasets. We use the same two datasets as in IIRC [1]: CIFAR100 [81] and ImageNet [35]. For CIFAR100, we take the two-level hierarchy split IIRC-CIFAR from IIRC [1], we denote this as IIRC-2-CIFAR. It is composed of 15 superclasses and 100

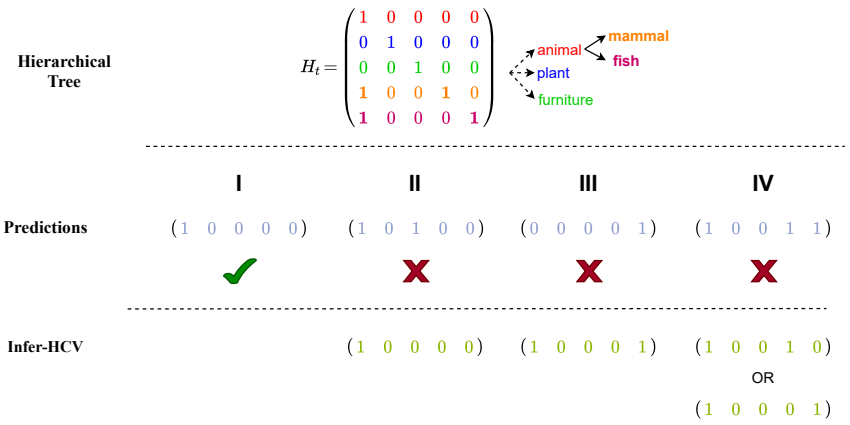


Figure 3.3 – Examples Illustration of Infer-HCV procedure.

subclasses. To further explore the performance of incremental learning methods over multi-level hierarchy, we further extend the IIRC-2-CIFAR into a three-level hierarchy dataset IIRC-3-CIFAR with two highest superclasses (we name them as "root"): "animals" and "plants". That accounts 2 rootclasses, 15 superclasses and 100 subclasses. For ImageNet, due to its huge amount of data, we collect 100 subclasses according to the hierarchy proposed in IIRC [1]. In total there are 10 superclasses and 100 subclasses (including those have no superclass labels). We denote this dataset as IIRC-ImageNet-Subset as a simplified version of the original one.

Incremental task configurations. For IIRC-2-CIFAR, we adopt the training sequence from IIRC [1], where the first task is with 10 superclasses, in the sequential tasks each with 5 classes. And for IIRC-3-CIFAR, we uniformly distribute the rootclasses and superclasses to form 23 tasks in total, the first task is 7 classes and then the coming tasks are 5 for each. For IIRC-ImageNet-Subset, we have 11 tasks each with 10 classes. Here the superclasses are also uniformly distributed. We want to stress that the uniform distribution of superclasses (and rootclasses) leads to a more challenging setting than proposed in the original IIRC.

Baselines and Compared methods. We compare the performance of the following variants: (1) **Incremental Joint** learns the model across tasks and the model has access to all the data from previous tasks with complete information (having access to all the label annotations Y_t). It serves as the upper bound for comparison. (2) **ER-infinite** is similar to *Incremental Joint* but with incomplete information (only

Methods	iCaRL-CNN			iCaRL-norm			LUCIR		
	-	+ SPL	+ SPL + infer HCV	-	+ SPL	+ SPL + infer HCV	-	+ SPL	+ SPL + infer HCV
IIRC-2-CIFAR	28.4	32.7	**35.9**	24.9	29.1	31.9	28.5	33.0	34.7
IIRC-3-CIFAR	20.5	26.0	27.1	19.6	25.6	25.9	16.1	35.5	**37.2**
IIRC-ImageNet-Subset	28.7	29.3	**31.7**	28.2	29.1	31.3	23.3	26.8	28.2

Table 3.1 – We show the average of *pw-JS* from comparison over three datasets with and without our HCV module. + *SPL* means applying HCV in training stage, + *Infer-HCV* means applying HCV module in inference time.

access to the current label annotations y_t). (3) **iCaRL-CNN** is the original version of incremental learning method iCaRL [142]. (4) **iCaRL-norm** is the adapted version of iCaRL [142] with replacement of the distance metric from L2-distance to Cosine similarity. (5) **LUCIR** is the incremental learning method LUCIR [63]. (6) **ER** is the finetuning baseline with 20 image exemplars per class as experience replay. (7) **FT** is the finetuning baseline without image replay.

Implementation details. For most implementation details, we follow the IIRC configurations [1]. For these three setups, we use the ResNet-32 [57] as the classification backbone. For model training, we use SGD (momentum=0.9) as optimizer, which is commonly used in continual learning [121]. For the IIRC-2-CIFAR and IIRC-3-CIFAR setting, the learning rates begin with 1.0 then decay by 0.1 on the plateau of the validation performance. For IIRC-ImageNet-Subset, the learning rate starts with 0.5 and decay by 0.1 on the plateau. The number of training epochs is 140, 140 and 100 for IIRC-2-CIFAR, IIRC-3-CIFAR and IIRC-ImageNet-Subset, respectively. For all these three setups, the batch size is 128 and weight decay is 1e-5.

During training, we apply random resized cropping (of size 32×32) to both CIFAR100 and ImageNet images. Then a random horizontal flip is applied and followed by a normalization. And for images replay, we keep a fixed number of 20 saved exemplars per class by default. For evaluation, we adopt the *precision-weighted Jaccard similarity (pw-JS)* proposed in IIRC [1], which integratedly considers both precision and recall indexes. And the threshold τ is set to 0.6 in all experiments (except in ablation study over it).

3.4.2 Experimental results

HCV applied to existing methods. To verify the performance of our proposed HCV, we apply it to iCaRL-CNN, iCaRL-norm and LUCIR. The average *pw-JS* value is provided in Table 3.1. We conduct experiments using three different settings, that is

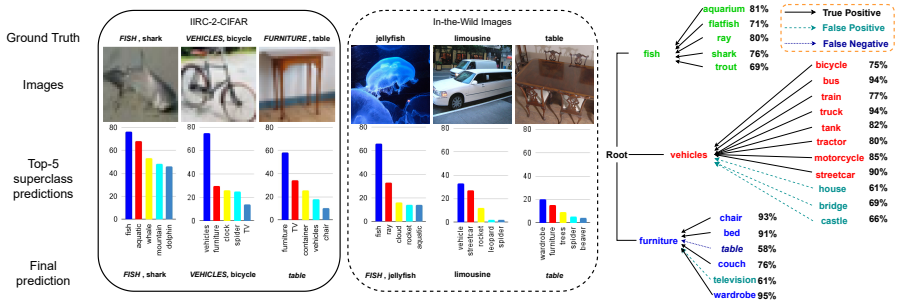
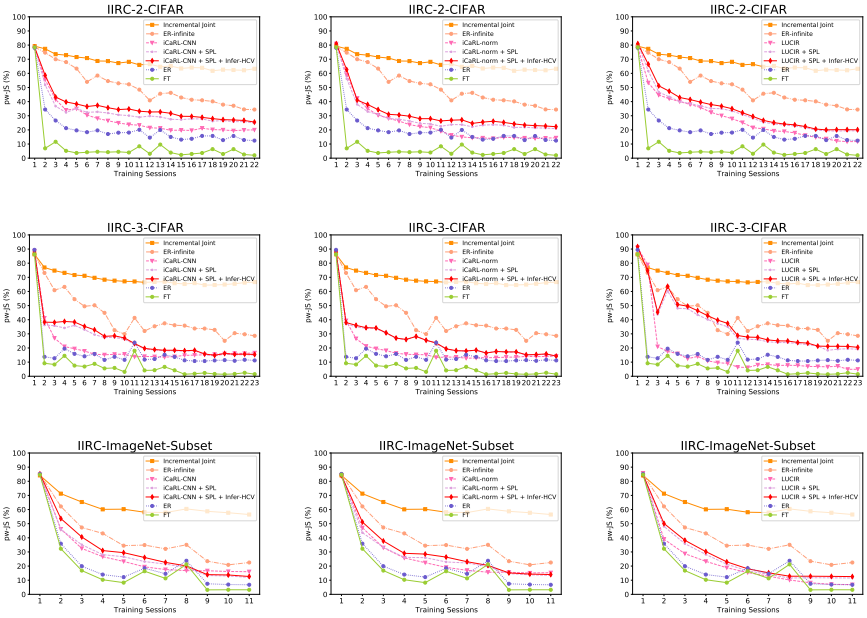


Figure 3.4 – Visual examples of our model applied to IIRC-2-CIFAR setup (annotated with superclasses and subclasses) and in-the-wild images (annotated with class names). We plot the top-5 (ranked by % percentage) predicted superclasses for each query image. We take the default threshold $\tau = 0.6$ to distinguish the success and failure cases. A subgraph of the final predicted graph under IIRC-2-CIFAR setup with iCaRL method is shown on the right. Here the top-1 predicted superclasses with percentages are listed.

IIRC-2-CIFAR, IIRC-3-CIFAR and IIRC-ImageNet-Subset. On IIRC-2-CIFAR setting, with the help of our HCV module during the training stage, the average numbers are increased by nearly 4.3% for all three different continual learning methods. When we apply HCV also at inference time, it further improves the consistency of final predictions achieving the average number by 3.2%, 2.8%, 1.7% for these three methods respectively. On the IIRC-3-CIFAR setting, since it is a much harder setup for incremental learning, all these variants suffer a significant drop of performance. LUCIR is much better compared to iCaRL-CNN and iCaRL-norm. Applying HCV in both training and inference stages helps to boost performance around 6.5% for two iCaRL variants and 21.1% for LUCIR. IIRC-ImageNet-Subset setting has much higher image diversity, thus it also imposes difficulties for these incremental methods. Under this setting, LUCIR performs worse than iCaRL-CNN and iCaRL-norm even with the improvement from HCV. And iCaRL-CNN works similar to iCaRL-norm but with marginally better performance. Overall, using our proposed HCV during training and inference improves performance of existing methods consistently for different settings.

Final estimated hierarchy graph and visual examples. After learning the last task under IIRC-2-CIFAR setup when applying our SPL module to iCaRL-CNN, we estimate the full hierarchy and draw a subgraph with 3 superclasses in Fig. 3.4 (right). We can observe that most subclasses are correctly annotated with its superclasses. However *table* is not correctly annotated because its confidence (58%) does not

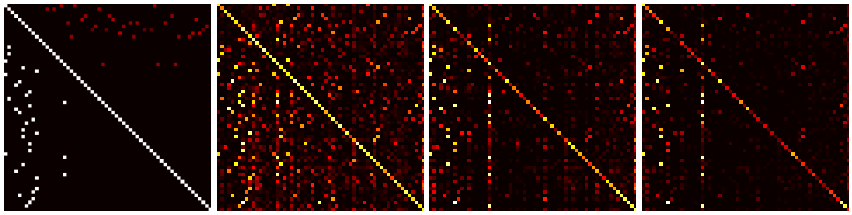


(a) iCaRL-CNN (b) iCaRL-norm (c) LUCIR

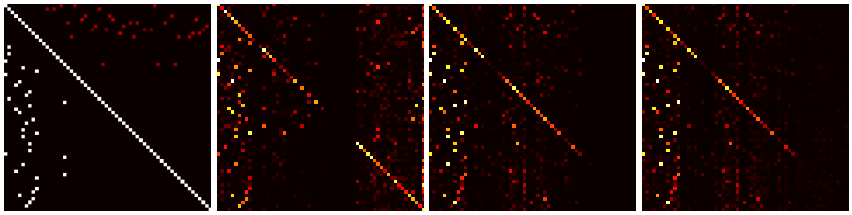
Figure 3.5 – Experimental results over IIRC-2-CIFAR, IIRC-3-CIFAR and IIRC-ImageNet-Subset setups based on three methods: iCaRL-CNN, iCaRL-norm and LUCIR.

reach the threshold. Interestingly, *television* is wrongly classified as a subclass of *furniture*. In real life, we could also regard it as a member of *furniture* and this was learned because *televisions* occur often in *furniture* scenes. This kind of information can help human operators in annotating and verifying the dataset hierarchy. Further, we see that *house, bridge, castle* are false positives, and are classified as subclasses of *vehicles*. This could be because *vehicles* images co-occur with the *house, bridge, castle* classes as their background. Finally, we also show some visual examples from IIRC-2-CIFAR setup and in-the-wild images in Fig. 3.4(left).

Comparison with SOTA methods. In Fig. 3.5 we plot the dynamic performance changes of different methods. The general trend on different settings are similar. Incremental Joint always achieves the best results as an upper bound, benefiting from access to all data and labels, while ER-infinite lacks the knowledge of full labels resulting in a worse performance. Our proposed HCV improves existing methods consistently, but the gap between our best and the two upper bounds



(a) Ground truth (b) iCaRL-CNN (c) + SPL (d) +SPL+HCV



(e) Ground truth (f) LUCIR (g) + SPL (h) +SPL+HCV

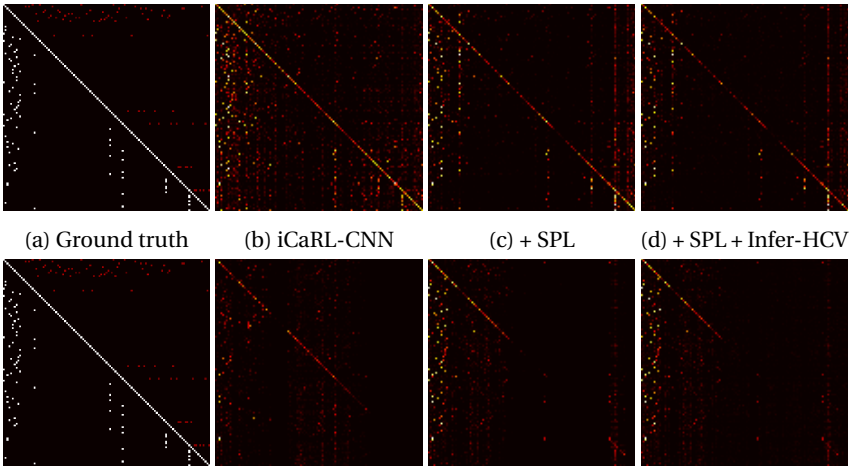
Figure 3.6 – Confusion matrices of groundtruth, original continual learning methods, applying SPL and applying Infer-HCV after task 11 under IIRC-2-CIFAR setup. The first row is obtained with iCaRL-CNN as the base method and the second row is based on LUCIR.

(ER-infinite and Incremental Joint) is still large, which shows that IIRC setting is a very challenging setting requiring more research.

Confusion matrices. Fig. 3.6 and Fig. 3.7 show the confusion matrices after learning task 11 and task 22 (the last task) under IIRC-2-CIFAR setup. They are from the ground truth, original continual learning methods, and HCV applied to both training and inference time. It can be observed that after using HCV, the redundant predictions are cleaned with our learned prior knowledge about the classes hierarchy, therefore HCV plays a role of a de-noising procedure for confusion matrices.

3.4.3 Ablation study

Ablation study over threshold τ. We conduct an ablation study on the threshold τ under IIRC-2-CIFAR setup. In Fig. 3.8a, we compare the values of τ {0.4, 0.5, 0.6, 0.7} when applying HCV on both training and inference stages. We can observe that with different hyper-parameters, it improves over iCaRL-CNN consistently. In Fig. 3.8b, we show how the hierarchy correctness score (HCS) changes with the threshold from 0.1 to 0.8, and is around 75% to 80% when τ is in the range [0.3, 0.7].



(a) Ground truth (b) iCaRL-CNN (c) + SPL (d) + SPL + Infer-HCV

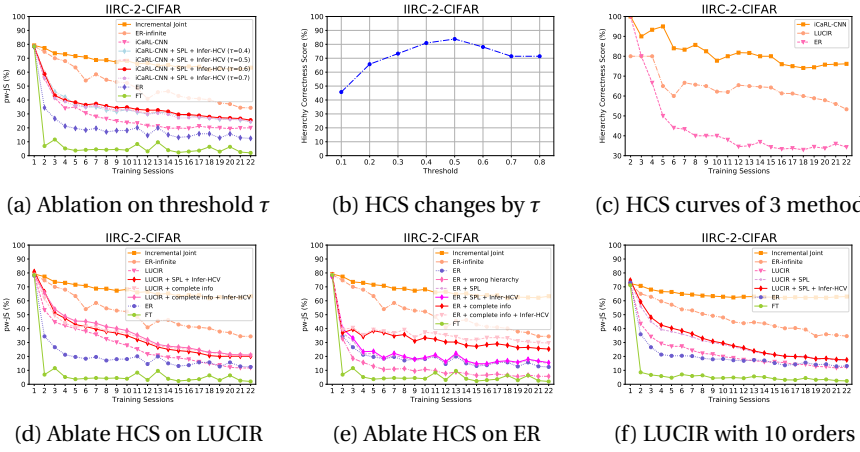
(e) Ground truth (f) LUCIR (g) + SPL (h) + SPL + Infer-HCV

Figure 3.7 – Confusion matrices of groundtruth, original continual learning methods, applying SPL and applying Infer-HCV after the last task under IIRC-2-CIFAR setup. The first row is obtained with iCaRL-CNN as the base continual learning method and the second row is based on LUCIR.

In our experiments, we set $\tau = 0.6$ by default.

Ablation study over hierarchy correctness score (HCS). We also conduct an ablation study over the HCS on LUCIR and ER methods as shown in Fig. 3.8d and Fig. 3.8e. The hierarchy correctness scores for iCaRL, LUCIR, ER are 76.2%, 56.0%, 34.3%, respectively (the HCS curves by training sessions are shown in Fig. 3.8c). The higher hierarchy correctness score for iCaRL-CNN helps it achieve state-of-the-art performance on IIRC-2-CIFAR and IIRC-ImageNet-Subset (Table 3.1 and Fig. 3.5). While LUCIR achieves a much lower score though it is regarded as one of the best methods in continual learning [113].

We also show the performance of the LUCIR and ER methods with the groundtruth hierarchy, which means it has a HCS of 100% (see Fig. 3.8d and Fig. 3.8e). In this case 3.0% and 15.0% improvements are observed for LUCIR and ER respectively. That implies that our HCV module can benefit from a preciser hierarchy estimation to reduce the gap to ER-infinite. To test how a completely wrong class hierarchy influences our model, we randomly generate a hierarchy for IIRC-2-CIFAR and apply it to ER (Fig. 3.8e), we can observe a drop of HCS from 34.3% to 0.0%, and the overall performance drops for ER to nearly 7.0%.



(a) Ablation on threshold τ (b) HCS changes by τ (c) HCS curves of 3 methods

(d) Ablate HCS on LUCIR (e) Ablate HCS on ER (f) LUCIR with 10 orders

Figure 3.8 – Ablation study over threshold τ, HCS and class orders on IIRC-2-CIFAR setup.

HCV (on LUCIR) performance with 10 orders. In Fig. 3.8f the experiments are conducted with all 10 task-orderings proposed in IIRC [1]. We plot the average performance. Here we apply our SPL and Infer-HCV to the LUCIR model. We observe a significant and consistent improvement compared to the ER baseline (≈10.0%) and the basic LUCIR method (≈8.0%). In conclusion, our method improves the performance under various orders and settings.

3.5 Conclusion

In this chapter, we proposed a Hierarchy-Consistency Verification module for Incremental Implicit-Refined Classification (IIRC) problem. With this module, we can boost the existing incremental learning methods by a large margin. From our experiments on three different setups, we evaluate and prove the effectiveness of our proposed module during both training and inference. And from the visualization of confusion matrices, we can also find that our HCV module works as a denoising method to the confusion matrices. For future work, we are interested in associating hierarchical classification, multi-label classification with IIRC problem, thus to have a more robust model to overcome forgetting in more realistic setups.

4 ACAE-REMIND for Online Continual Learning with Compressed Feature Replay*

4.1 Introduction

The vast majority of deep learning papers consider that all training data is available jointly, and the learner can process the data several times (epochs) to learn the optimal parameters to solve the task at hand. However, in many real-world scenarios, this would not be possible, and the learner has only access to data of a single task at the time, before proceeding to learn a new task. This scenario refers to *continual learning* (or *incremental learning, lifelong learning*). The main challenge in this scenario is to learn from the current data while preventing forgetting the knowledge of previous tasks. With a naive finetuning approach the model will suffer a drastic drop in performance on previous tasks because the model aims to be optimal for the current tasks, and ignores performance on previous tasks. This phenomenon is known as *catastrophic forgetting [76, 117]*. The field of continual learning studies methods that prevent forgetting [37, 54, 63, 114, 142, 184].

A challenging setting in continual learning, yet common in practical application, is *online continual learning* of non-iid data streams [6, 27, 108]. In the online setting, each image can only be observed once during model optimization (except exemplars in storage). These applications mainly exist in resource constrained devices, such as mobile phones, robots and other smart devices. The majority of methods in continual learning, known as batch incremental learning methods, allow for several cycles (epochs) over the data [33]. These methods cannot operate in the challenging online continual learning setting. Moreover they take longer to train. In this chapter, we focus on online continual learning.

Among the approaches to address catastrophic forgetting, some of the best performing ones are rehearsal-based [54, 142, 155]. Several methods save a small set of exemplar images of previous classes [27, 63, 142, 184]. Retrieving them during future training is a straightforward way to prevent forgetting. For example, GEM [108], A-GEM [27] and MIR [6], which address online continual learning, belong to this type. However, this strategy leads to increased memory usage and the problem of training from imbalanced data (between previous tasks and the current task). An alternative is to generate images via generative models (e.g. GANs) [155, 183].

*This chapter is based on a publication in the Pattern Recognition Letters, 2021 [175]

However, image generation is still a difficult problem in computer vision and requires complex generative models, which would also need to be continually learned, making this method not practical for complex datasets.

To circumvent the difficulties of image replay, recent work has focused on feature replay [54, 106]. In [106] a generator is trained to replay compact feature representations of the images (after the last average pooling layer of a ResNet-18). In addition, a distillation loss was applied to prevent forgetting of the feature extractor. Instead of generating features, it was also observed by Hayes *et al.* [54] that saving them as exemplars is very efficient, since it required less memory per image. To even further reduce the memory requirements, the REMIND method [54] also applies product quantization [65]. This allows them to save up to 1M compressed feature exemplars, instead of 20K exemplar images saved traditionally, in the same memory buffer. A major drawback of these feature replay methods [54, 106] is that they either allow for very little training [106] or no training at all [54] of the backbone feature extractor (located before the replay layer). As a consequence, if this backbone is not yet optimally trained for future tasks, the performance is sub-optimal.

To address the limitation of feature replay, we propose ACAE-REMIND, an auxiliary classifier auto-encoder (ACAE) that allows for compressed feature replay (as in REMIND) at intermediate layers of the network. This contrasts with current feature replay methods that focus on replaying the features in the last layers. The principal advantage of our method is that we can jointly train all layers after the replay layer. This addresses an important problem of feature replay methods, namely the reduced performance because of a large fixed backbone network. Instead of only the 4M parameters that are trained in REMIND when replaying at block 4 of a ResNet-18, we allow to train 9M parameters jointly when replaying on block 3. This leads to feature representations that are more discriminative between the classes of current and previous tasks. We evaluate our method in the challenging, yet more realistic, *online continual learning* setting. From experiments under multiple settings and datasets, we observe state-of-the-art performance in many-task evaluations and competitive in few-task settings.

4.2 Related work

4.2.1 Continual learning

Continual learning methods can be categorized into three types which we will shortly comment. For a more elaborate overview see the following surveys [33, 113]).

Regularization-based methods. The first group of techniques is based on regu-

larization. These methods add a regularization term to the loss function which impedes changes to the parameters deemed relevant to previous tasks. The difference depends on how to compute the estimation. From these differences, these methods can be further divided into data-focused [94] and prior-focused [76]. Data-focused methods use knowledge distillation from previously learned models. Prior-focused methods estimate the importance of model parameters as a prior for the new model. However, it has been shown that data-focused methods are vulnerable to domain shifts [7] and prior-focused methods might be not sufficient to restrict the optimization process to keep acceptable performances on previous tasks [39].

Parameter isolation methods. This family focuses on allocating different model parameters to each task. These models begin with a simplified architecture and updated incrementally with new neurons or network layers in order to allocate additional capacity for new tasks. In Piggyback/PackNet [111, 112], the model learns a separate mask on the weights for each task, whereas in HAT [154], masks are applied to the activations. This method is further developed to the case were no forgetting is allowed in [114]. In general, this branch is restricted to the task-aware (task incremental) setting. Thus, they are more suitable for learning a long sequence of tasks when a task oracle is present and there is no constraint over model capacities.

Replay methods. This type of methods prevent forgetting by including data (real or synthetic) from previous tasks, stored either in an episodic memory or via a generative model. There are two main strategies: exemplar rehearsal [27,63,142,184] and pseudo-rehearsal [155, 183]. The former store a small amount of training samples (also called exemplars) from previous tasks. The latter use generative models learned from previous data distributions to synthesize data.

One of the main drawbacks of exemplar replay is the high memory usage required to store exemplars of previous tasks. REMIND [54] addresses this drawback, instead of saving original data, it saves compressed latent representation of intermediate layer features via product quantization [65]. This is a more efficient usage of memory and computation. However, due to restriction that the backbone is fixed, the majority of feature extraction modules cannot be adopted to later tasks. Therefore, this model has a strong bias towards the first task. Recently, GDumb [136] proposes training a model only from exemplars. The main idea is to balance the sample reservoir in a selection stage, then the model is learnt from scratch on this balanced set. While not designed for any specific continual learning settings, it achieves excellent performance on many. It reveals that sample balancing is crucial for rehearsal-based continual learning methods. While saving images is always expensive compared to saving features, this point is also mentioned in paper on feature adaption [64].

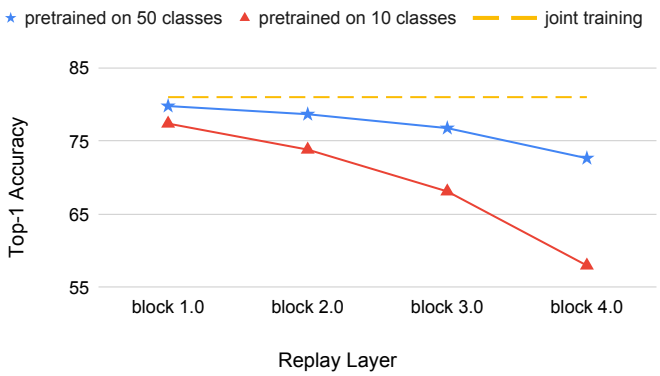


Figure 4.1 – Drop in performance due to frozen backbones (Joint training: 81.0)

4.2.2 Auto-encoders and product quantization

Auto-encoders [79] learn representations in an unsupervised way by encouraging the model to reconstruct the input data. An encoder projects the high-dimensional input to a low-dimensional space, and the decoder tries to project back to the original space minimizing the reconstruction error. Product quantization [65] is an effective quantization method that performs a decomposition of a high-dimensional space into the Cartesian product of a series of subspaces, and quantizes them separately.

In our model, we propose an auto-encoder with an auxiliary classifier (ACAE) to force the reconstructions not only to remain close to original inputs but also keeping the classification characteristics. By combining ACAE with Product Quantization (PQ), the feature spaces are decomposed from high dimension to low dimension, from float numbers to integer indexes, which leads to better compression and therefore allows to save more exemplars.

4.3 Compressed Feature Replay

4.3.1 Feature replay location

Pseudo-rehearsal methods [155, 183] are limited by the performance of generative models to generate high-quality images. As a results these methods perform poorly on more complex real-world datasets. To address this limitation, Liu *et al.* [106]

proposed generative feature replay (GFR) to generate features of an intermediate layer. In their proposal, features before the classifier are generated by a conditional GAN learned in a continual fashion.

REMIND [54] observes that storing features is much more efficient than storing images. They operate on the same block as GFR. With the help of PQ [65] the features are further compressed in REMIND. The features from block 4.0 of ResNet-18 are approximated by a number of codebooks and indexed feature maps. By this means, the floating point values of feature representations are replaced by integer index numbers. This allows them to save 50× more feature exemplars than image exemplars in the same memory space and obtain excellent results for online continual learning.

As noted in the introduction, one of the main drawbacks of REMIND (and feature replay in general) is that these methods freeze the backbone feature extractor (i.e the layers before the feature replay) after training the first task. They only train the layers that come after the feature replay layer for the remaining tasks. Depending on the continual learning scenario this could lead to a significant drop in performance because of a suboptimal backbone network.

To better understand the impact of freezing the backbone network after the first task we perform an experiment on the ImageNet-Subset dataset [35] with ResNet-18 as the backbone. We consider two scenarios, one with the first task containing 50 classes and the remaining 50 classes divided into five more tasks. In the second scenario, we evenly divide the classes over 10 tasks (each with 10 classes). Clearly, the second scenario is more challenging for REMIND, because now the backbone network can only be trained on the 10 classes of the first task. In Fig. 4.1 we can see the drop in performance which is caused by freezing the backbone network as a function of the position of the feature replay. The performance of the different backbone networks is computed in the following way: we first train the first task and fix the backbone network, then we jointly train the remaining layers on all training data of all tasks (this can be seen as the upper bound for this continual learning setting, i.e. joint training with the backbone frozen). As can be seen, the drop in performance is significant (by comparing the difference of the blue and red with the yellow line), dropping 8.36% in the first scenario, and 23.04% in the second scenario when replaying the features of block 4.0. As can be seen the drop diminishes considerably by performing the replay at earlier layers. The reason why REMIND chooses to replay at block 4.0 is because the proposed technique does not scale well to lower positions in the network. This is explicitly discussed and they mention that the quantized features would be too large and would significantly increase storage requirements[†].

[†] See Supplementary material S2 [54].

Table 4.1 – Comparison of replay methods.

method name	replay layer	compression	online
GFR	last block	GAN	✗
REMIND	last block	PQ	✓
Ours	interm. block	ACAE+PQ	✓

To overcome the limitations of feature replay, our proposed ACAE-REMIND model aims to apply replay on an intermediate blocks. To reconstruct features in intermediate layers, we introduce a stronger compression module, which achieves dimension reduction, and feature approximation while maintaining the classification characteristics of the replayed features. The method is an extension of REMIND and is based on an Auto-Encoder with Auxiliary Classifier (ACAE). We can perform joint training on all layers after the replay layer (and not only the last block as in REMIND). This alleviates the drawback of fixing the backbone neural network. A comparison among the discussed feature replay methods is in Table 4.1.

4.3.2 Online continual learning setting

Online continual learning is a subarea of incremental learning, where the algorithm is only allowed to make a single pass through the data of each task. It is more related to real-life and real-time applications since data comes in sequential streams, and they are not allowed to use the same sample more than one time (unless stick to buffer) in the whole learning process.

Suppose we have a data stream of triplets $\{(x_t^i, y_t^i, t)\}_i$, where x_t^i is the i-th input, y_t^i is the corresponding label and t is the task identifier ($t \in \mathcal{T} = \{1, ..., T\}$). Each input-label pair is an identical and independent sample drawn from an unknown distribution $P_t(X, Y)$ of task t. We consider the number of tasks T is unknown and the tasks are coming sequentially as $t = 1, ..., T$. We also assume that data among tasks are disjoint. At inference time, the task-id t is unknown at all time; also referred to as task-agnostic inference. Under this assumption, the resulting model $f(x; \Theta)$, parameterized by Θ, is optimized to minimize a predefined loss $l(x, y; \Theta)$ over new sequential input samples (x_t^i, y_t^i) from current data stream t. And at the same time, the performance on previous tasks should not decrease.

4.3.3 ACAE-REMIND for compressed feature replay

The ACAE-REMIND model is designed for online task-agnostic continual learning in a memory efficient way. As explained before, we aim to execute feature replay on an

intermediate layer. Since all layers after the feature replay can be jointly trained, this strategy can potentially lead to improved performance. Because the distribution in lower layers is more complex, we propose an improved compression mechanism based on an ACAE module. The whole training procedure can be roughly divided into: 1) initialization stage (in Fig. 4.2) of the classification model, ACAE and PQ modules, 2) online continual learning stage (in Fig. 4.3).

Initialization

During initialization, the classification model, ACAE and PQ are trained sequentially with data $\left(x_1^i, y_1^i\right)$ from the first task $t = 1$. In the first step, the whole classification model is optimized in an offline way. This step aims to learn a robust pretrained model for future tasks (similar as in REMIND). The parameters Θ are updated by minimizing the cross-entropy loss:

$$minimize_{\Theta} \mathscr{L}_{CE}(y_1^i, \hat{y}_1^i) = -y_1^i \cdot log\ \hat{y}_1^i \tag{4.1}$$

where the prediction is given by $\hat{y}_1^i = f\left(x_1^i; \Theta\right)$.

Secondly, the ACAE module is inserted into the layer where we will replay the features. We denote the layers before and after the replay layer as $g(x; \Theta_1)$ and $h(z; \Theta_2)$, where Θ is the union of Θ_1 plus Θ_2 and $z = g(x; \Theta_1)$ in step 1. The encoder and decoder of ACAE are denoted as $u = D_{enc}(z; \Gamma)$ and $\hat{z} = D_{dec}(u; \Pi)$. The ACAE is trained with an auxiliary classification loss and auto-encoder reconstruction MSE(mean square error) loss on the same data stream $\left(x_1^i, y_1^i\right)$.

Then parameters Γ, Π are computed by minimizing the ACAE loss:

$$minimize_{\Gamma,\Pi}\ \mathscr{L}_{ACAE}(y_1^i, x_1^i) = \mathscr{L}_{CE}(y_1^i, \hat{y}_1^i) + \left\| z_1^i - \hat{z}_1^i \right\|_2 \tag{4.2}$$

where

$$\begin{aligned} z_1^i &= g\left(x_1^i; \Theta_1\right) \\ \hat{z}_1^i &= D_{dec}\left(D_{enc}\left(z_1^i; \Gamma\right), \Pi\right) \\ \hat{y}_1^i &= h\left(\hat{z}_1^i; \Theta_2\right). \end{aligned} \tag{4.3}$$

After that, the last step is to train a PQ encoder-decoder pair $P_{enc}(u; \Upsilon)$ and $P_{dec}(v; \Psi)$ to approximate latent representations extracted from ACAE's encoder.

Here Υ, Ψ are learnt from the object function of PQ MSE loss:

$$\begin{aligned} minimize_{\Upsilon,\Psi}\ \mathscr{L}_{PQ}\left(x_1^i\right) &= \left\| u_1^i - \hat{u}_1^i \right\|_2 \\ z_1^i &= (g\left(x_1^i;\Theta_1\right) \\ u_1^i &= D_{enc}\left(z_1^i;\Gamma\right) \\ \hat{u}_1^i &= P_{dec}\left(P_{enc}\left(u_1^i;\Upsilon\right),\Psi\right) \end{aligned} \tag{4.4}$$

Online continual learning

In online continual learning, only layers after the ACAE decoder can freely adjust to new tasks (the parameters in Θ_2), other modules (including parameters $\Theta_1, \Gamma, \Pi, \Upsilon, \Psi$) are all fixed during training. The new coming images from the data stream are passed through lower layers, the ACAE encoder and the PQ encoder to get their latent representations v_t^i with corresponding integer indexes. This is computed as:

$$v_t^i = P_{enc}\left(D_{enc}\left(g\left(x_t^i;\Theta_1\right);\Gamma\right),\Upsilon\right) \tag{4.5}$$

Then its representation is mixed with randomly selected $\mathscr{N}$ previous samples $v_{\hat{t}}^j$ ($\hat{t} < t$) from the reservoir to reconstruct features via the PQ decoder and the ACAE decoder. Those features will be taken to optimize the trainable parameters Θ_2 with the cross-entropy loss $\mathscr{L}_{CE}\left(y_{\bar{t}}^i, \hat{y}_{\bar{t}}^i\right)$ and $\hat{y}_{\bar{t}}^i$ is formed as:

$$\hat{y}_{\bar{t}}^i = h\left(D_{dec}\left(P_{dec}\left(v_{\bar{t}}^i;\Psi\right),\Pi\right),\Theta_2\right),\ \bar{t} \leq t \tag{4.6}$$

Reservoir sampling After optimization, the new representation will be stored in the reservoir memory. If the reservoir is full, we randomly select a sample to pop up from one of the classes with most samples in the reservoir.

4.4 Experiments

4.4.1 Experimental setup

Datasets. Our evaluations are performed on three datasets: ImageNet-Subset [35], CIFAR100 [81] both with 100 classes[‡] and CIFAR10 with 10 classes. We use data aug-

[‡]Samples are presented in a random yet fixed presentation order, as proposed in iCaRL [142], and adopted by others [37, 63, 136, 184].

mentation during the initial training of the full model and the ACAE, but removed it to train PQ (we save the representation of the original image - without augmentation -) and during the online continual learning stage. For feature augmentation, we only randomly resize and crop reconstructed features in the online continual learning stage.

Implementation details. We use Resnet-18 as our classification network for ImageNet-Subset. For CIFAR10 and CIFAR100, we use adapted Resnet-18 and Resnet-32 respectively (using only 3 blocks instead of the original 4 blocks). During initialization, the backbone network is learned from scratch with SGD, then the ACAE is trained with Adam [75]. For PQ training, we use the implementation from the Facebook Faiss library [70]. During the online continual learning stage we use SGD.

Evaluation metrics. We consider two widely used metrics: Average of top-1 accuracy over classes (AOC) up to the current task and top-1 accuracy after the last task (LAST).

Experimental settings. We will evaluate our method in five different settings. For the first three settings on ImageNet-Subset and CIFAR100, we use half of the classes as the first task and split the remaining into 5, 25 and 50 tasks with equal split (this setting is widely used [37, 63, 136]). We also refer to these as the 5, 25 and 50 steps setting. The fourth setting is splitting ImageNet-Subset into 10 tasks of the same size (this setting is used in [139, 142, 184]). We compare with several methods: iCaRL [142], BiC [184], UCIR [63], PODNet [37], GDumb [136], RPSnet [139] and REMIND [54]. We note that, except REMIND, the other methods are mainly designed for offline continual learning, which is a simpler setting compared with our online setting.

The fifth setting is on CIFAR10, where we use the commonly used setting from GMED [69], which divides CIFAR10 into 5 tasks equally. And we compare with online continual learning methods: AGEM [27], BGD [199], GEM [108], GSS-Greedy [8], HAL [26], ER [146], MIR [6], and GMED [69].

4.4.2 Results of online continual learning

Few-task evaluation (5 steps setting). We report the AOC metric on ImageNet-Subset with the 5 steps setting in Table 4.2. Every time we have a new input image, we randomly sample $\mathcal{N} = 50$ previous latent representations from the sample reservoir. It can be seen that our method outperforms REMIND and that, especially for larger memory, we can obtain excellent results by replaying lower layers. For comparison, we have also computed results for block 3.0 for standard REMIND (going to lower blocks did further reduce performance). We observe that our method with

Table 4.2 – Comparison on Imagenet-Subset, we show the averages over classes (AOC) with 50 classes as the first task and 5/25/50 steps each with 10/2/1 classes. For REMIND and our method, we show the replay layer (block number) in brackets. The highest numbers in each row are highlighted.

On-line	Exemplar info			Methods	AOC over various steps		
	Num.	Shape (CHW)	Mem. (MB)		5	25	50
✗	2K	3 224 224 (int)	301	iCaRL	65.56	54.56	54.97
				BiC	68.97	59.65	46.49
				UCIR(NME)	69.07	60.81	55.44
				UCIR(CNN)	71.04	62.94	57.25
				PODNet(CNN)	**75.54**	68.31	62.08
				GDumb	-	-	62.86
✓	130K	8 7 7 (int)	51	REMIND(3.0)	70.58	67.93	67.35
				REMIND(4.0)	71.02	70.50	70.14
				Ours(1.0)	60.70	56.37	56.10
				Ours(2.0)	70.24	67.65	66.23
				Ours(3.0)	**72.58**	**71.43**	**70.69**
✓	130K	32 7 7 (int)	204	REMIND(3.0)	72.46	-	-
				REMIND(4.0)	73.98	-	-
				Ours(1.0)	**74.08**	-	-
				$-\mathscr{L}_{CE}$	70.26	-	-
				Ours(2.0)	73.75	-	-
				Ours(3.0)	73.63	-	-

32 codebooks is only 1.46% lower than the state-of-the-art offline PODNet method, and it well outperforms other methods. Even when we have only 8 codebooks, it is still better than all offline algorithms except PODNet. Another interesting phenomenon is that, with 32 codebooks, we get an increase from block 3.0 to block 1.0, but this trend gets reversed with only 8 codebooks. The reason is that in the lower layers, the latent representations contain more information and thus require more codebooks to be represented.

For CIFAR100 with 5 steps, the performance is shown in Table 4.3. Due to smaller image-size, the compression ratio is not as high as in ImageNet-Subset. In this case, offline continual learning (PODNet) outperforms the online settings by a larger margin (6.1%) under the same memory allocation. It should be noted that PODNet runs for 160 epochs over the data whereas the online methods can only do one epoch. Also, among the online methods, our method performs worse than REMIND(3.0) when considering a memory of 6.4MB. This is because a first task with many classes and more data allows REMIND to also learn a high-quality backbone network. For the larger memory setting (12.8MB) our method performs

Table 4.3 – Comparison on CIFAR100 dataset, we show the averages over classes (AOC) with 50 classes as the first task and 5/25/50 steps each with 10/2/1 classes. For REMIND and our method, we show the replay layer (block number) in brackets. The highest numbers in each row are highlighted.

Online	Exemplar info			Methods	AOC over various steps		
	Num.	Shape (CHW)	Mem. (MB)		5	25	50
✗	2000	3 32 32 (int)	6.14	iCaRL	58.08	50.60	44.20
				BiC	56.86	48.96	47.09
				UCIR (NME)	63.63	56.82	48.57
				UCIR (CNN)	64.01	57.57	49.30
				PODNet (CNN)	**64.83**	60.72	57.98
				PODNet (NME)	64.48	**62.72**	**61.40**
				GDumb	-	-	58.40
✓	25000	4 8 8 (int)	6.40	REMIND(2.0)	55.79	58.25	57.93
				REMIND(3.0)	58.71	59.26	58.99
				Ours(1.0)	53.38	53.74	54.25
				Ours(2.0)	57.57	59.28	59.50
✓	50000	4 8 8 (int)	12.8	REMIND(2.0)	58.51	59.94	59.87
				REMIND(3.0)	61.23	61.02	61.00
				Ours(1.0)	56.40	56.98	57.01
				Ours(2.0)	**61.27**	**62.49**	**62.30**
				$-\mathscr{L}_{CE}$	56.11	60.19	60.07

comparable to REMIND(3.0).

Many-task evaluation (25/50 steps setting). Here the number of rehearsed samples is set to $\mathscr{N} = 200$ because there are less samples in each step. The results for 25 steps on ImageNet-Subset are shown in Table 4.2. We obtain state-of-the-art with 3.12%, 0.93% and 4.50% higher than PODNet, REMIND (block 4.0) and REMIND (block 3.0) respectively. The hardest setting is the 50 steps split, where only 1 class is viewed at every time step. We show our performance in Table 4.2. It is 8.49% higher than GDumb in the offline setting and 1.21% better than REMIND in the online setting.

For the many-task evaluation of CIFAR100 shown in Table 4.3, we got marginally better than PODNet in 50 steps and worse in 25 steps under higher memory allocation. With lower memory allocation we still got competitive performances. In conclusion, from the many-task evaluation, we observe that bias correction methods suffer from a drop in performance with more time steps, while our model obtains better results and without much drop in performance when the number of tasks increases.

Equal split with 10 tasks (9 steps) on ImageNet-Subset. In the equal split setting,

Table 4.4 – Comparison on ImageNet-Subset, we show top-5 accuracy with 10 classes as the first task and 9 steps each with 10 classes. For REMIND and our method, we show the replay layer (block number) in brackets. The highest numbers in offline and online settings are highlighted.

Methods		iCaRL	RPSnet	BiC	REMIND (4.0)	Ours (3.0)	Ours (2.0)	Ours (1.0)
Online		✗			✓			
Exem. Info	Num.	2000 (20*100)			130000 (1300*100)			
	Shape (CHW)	3*224*224 (integer)			32*7*7 (integer)			
	Mem. (MB)	301.06			203.84			
Acc.	1	99.3	100	98.4	98.4	98.4	98.4	98.4
	2	97.2	97.4	96.2	91.6	93.3	93.5	94.1
	3	93.5	94.3	94.0	87.1	90.5	91.1	92.7
	4	91.0	92.7	92.9	82.2	87.2	87.7	90.2
	5	87.5	89.4	91.1	79.7	85.3	85.5	89.2
	6	82.1	86.6	89.4	77.7	84.0	85.0	87.8
	7	77.1	83.9	88.1	74.8	81.0	83.7	85.7
	8	72.8	82.4	86.5	72.8	80.9	82.7	85.4
	9	67.1	79.4	85.4	72.2	80.8	83.4	84.7
	10	63.5	74.1	**84.4**	70.9	79.6	81.8	**83.9**
	AOC	83.1	88.0	**90.6**	80.7	86.1	87.1	**89.2**

the 100 classes are divided into 10 tasks with 10 classes each. The top-5 accuracies in each time step are shown in Table 4.4 (For comparison, top-5 accuracy is adopted here since it is commonly used in this case). For this more challenging setting, REMIND obtains 8.5% lower results than ours, which is due to the less flexible backbone model. Especially here, it makes more sense to perform replay on a lower layer. Also note that in this setting, we get competitive results compared with the methods BiC [184] and RPSnet [139]. However, these methods are offline and perform multiple loops over the data.

Equal split with 5 tasks (4 steps) on CIFAR10. Several existing online methods cannot be straightforwardly applied to large datasets. To be able to compare to them, we also include results on CIFAR10 in Table 4.5. We divide the 10 classes into 5 tasks with 2 classes each. On this setting, we got 48.4%, which is 2.6% higher than the best reported results of GDumb. We also outperform the REMIND method with more than a 3% margin.

Table 4.5 – Comparison on CIFAR10 dataset, we show the LAST accuracy with 2 classes as the first task and 4 steps each with 2 classes. For REMIND and our method, we show the replay layer (block number) in brackets. The highest numbers in each row are highlighted.

Online	Exemplar info			Methods	LAST
	Num.	Shape (C*H*W)	Mem. (MB)		4 steps
✓	500	3*32*32 (integer)	1.536	Finetuning	18.5
				AGEM	18.5
				BGD	18.2
				GEM	20.1
				GSS-Greedy	28.0
				HAL	32.1
				ER	33.3
				MIR	34.5
				GMED(ER)	35.0
				GMED(MIR)	35.5
				GDumb	45.8
✓	24000	1*8*8 (integer)	1.536	REMIND(3.0)	45.2
				Ours(2.0)	**48.4**

4.4.3 Ablation study

One of the key ingredients of the ACAE-REMIND method is the auxiliary classification loss that is used during the training of the auto-encoder (see Eq. 4.2). This loss ensures that the compression does not remove the features that are crucial for classification. Here we ablate this factor. To show the impact of the classification loss, we evaluate the classification accuracy after compression with and without the loss (directly after Step 2), and compare this to the results that would be obtained with the uncompressed features (see Table 4.6). The results show that classification loss mitigates the classification drop that occurs due to compression. Finally, we have also ablated the loss in Table 4.2 and Table 4.3 on our best performing setting (indicated by rows with $-\mathscr{L}_{CE}$). The results show that the loss does greatly improve results resulting in a performance gain of 2-5%.

To better evaluate the influence of the block number n (the block where we introduce the ACAE-REMIND as seen in Fig. 4.3) we have included Fig. 4.4. Here we show a comparison on ImageNet-Subset and CIFAR-100 under the 5, 25 and 50 steps settings. We can conclude that under these settings, the performances are increasing with the feature replay from the first block to the penultimate block, and then decreasing when replaying on the last block. As can be seen the optimal

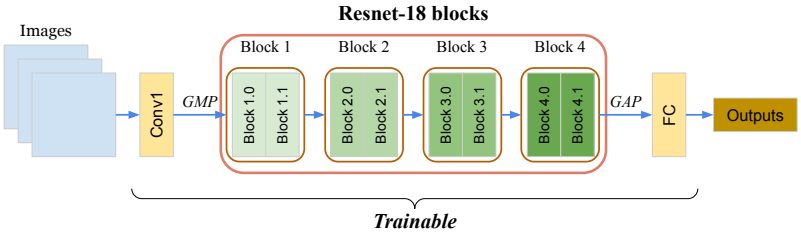
Table 4.6 – Ablation study of classification loss on CIFAR100 and ImageNet-Subset. The features are replayed from block 2.0 for CIFAR100 and block 1.0/2.0/3.0 for ImageNet-Subset.

method name	CIFAR100	ImageNet-Subset		
	block 2.0	block 3.0	block 2.0	block 1.0
Uncompressed features	76.00%	80.96%	80.96%	80.96%
ACAE replay(w/o $\mathscr{L}_{CE}$)	73.31%	78.80%	77.64%	76.52%
ACAE replay(w/ $\mathscr{L}_{CE}$)	74.45%	79.40%	79.76%	80.44%

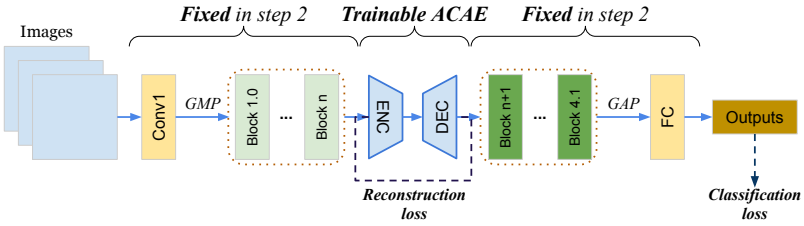
block is relatively stable with respect to the number of steps while keeping the same amount of data for the first task. If we however reduce the number of data for the first task, it becomes more difficult to learn a good backbone network and, as expected, the optimal n decreases: in Fig. 4.4(c) we can see that $n = 1$ yields optimal results.

4.5 Conclusions

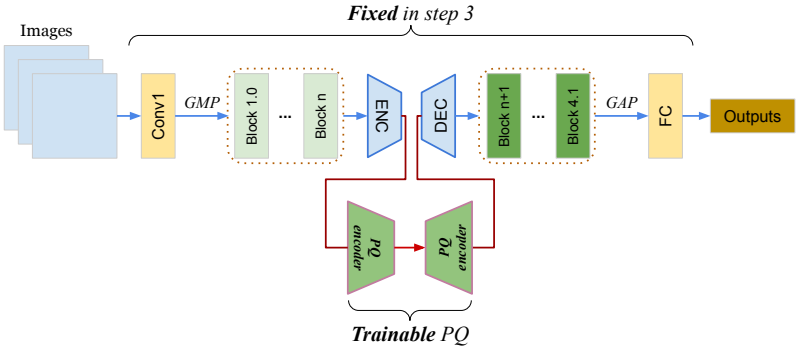
In this chapter, we proposed an extension to the REMIND method, called ACAE-REMIND. We propose a stronger compression module based on an auxiliary classifier auto-encoder that allows to move the feature replay to lower layers. The method is memory efficient and obtains better performance. In evaluation, we perform a comparison over multiple metrics among competitive methods. The strength of our model lies in the fact that with high compression ratio, we could save more feature exemplars than image exemplars. Especially, when the first task is relatively small (the 10-task scenario in ImageNet-Subset and 5-task in CIFAR10) we outperform REMIND with a large margin. As future work, we are interested in extending this framework to other continual learning problems.



(a) Step 1: Classification model (Resnet-18) training



(b) Step 2: ACAE (Auxiliary classifier auto-encoder) training



(c) Step 3: PQ (Product Quantization) training

Figure 4.2 – Overview of the initialization stage (trained on first task).

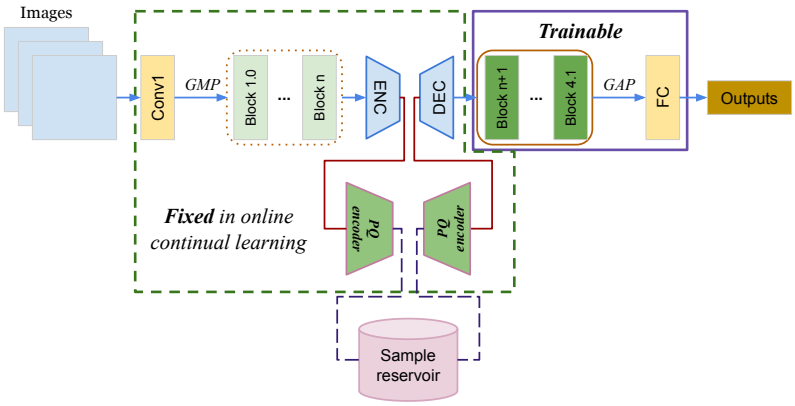
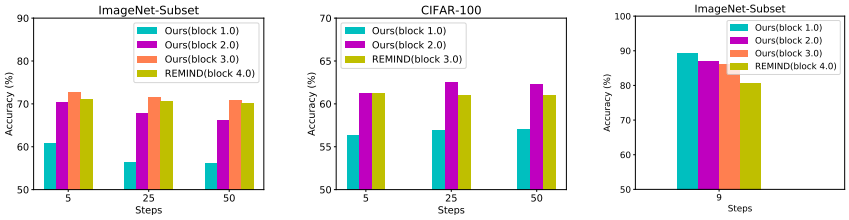


Figure 4.3 – Overview of our online continual learning phase (task $t = 2, ..., T$).



(a) First task with 50 classes then 5/25/50 steps.

(b) First task with 50 classes then 5/25/50 steps.

(c) First task with 10 classes then 9 steps.

Figure 4.4 – Ablation study on the block number n on ImageNet-Subset and CIFAR100 with various settings. The backbone for ImageNet-Subset is a 4-block Resnet-18 and for CIFAR100 is a 3-block Resnet-32. We show top-1 accuracy in (a) and (b), and top-5 accuracy (c).

5 Bookworm continual learning*

5.1 Introduction

Deep learning has brought extraordinary success to visual recognition by learning from large amounts of data (e.g. object classification and detection, scene classification). There are, however, two critical assumptions that stem from a *static* view of the world: all concepts of interest are known before training, and the corresponding training data is also available beforehand. The resulting model is also static and remains unchanged after training. The other limitation of conventional classification models is that there is no explicit notion of semantic similarity between concepts (i.e. a *semantic model*), since classes are represented as one-hot labels (i.e. all classes are equally similar and dissimilar to each other). These assumptions are hardly met in the *dynamic* real world we live in, where new visual data and new semantic concepts are continuously observed and integrated in our own personal knowledge. Similarly, visual recognition in humans greatly leverages all sort of semantic (and contextual) knowledge, enabling sophisticated inference.

Challenging this static world assumption, continual learning (CL) [5, 42, 103, 113, 136, 183, 197] focuses on how to update the visual model when new classes and visual instances are observed over time (see Fig. 5.1(a)). A consequence is that the data is no longer i.i.d. and learning new tasks results in forgetting previous ones (i.e. catastrophic forgetting). This problem has been addressed with different techniques, including weight regularization [5, 76, 103], distillation [94], episodic memories with exemplars [108, 142] and generative replay methods [155, 183].

On the other hand, zero-shot learning (ZSL) [53, 73, 101, 122, 178, 188, 189, 192, 197, 203] enables the recognition of (visually) unseen classes via a semantic model that describes them in connection to the seen classes (see Fig. 5.1(b)). We can also observe that ZSL has an implicit temporal structure, with the class descriptions learned first, then the visual model is learned from the data of seen classes, and then the model is tested over the unseen classes. ZLS is usually tackled as learning the alignment between visual features and class embeddings (via the semantic model) in an shared intermediate space [3, 45, 203, 205]. Recent works also use feature generators to synthesize features of unseen classes [122, 188, 189].

*This chapter is an extension based on a publication in the TASK-CV workshop, 2020 [171]

In this work, we argue that continual leaning and semantic models are both essential for visual recognition. We propose *generalized continual learning* (GCL) as a more realistic setting where visual recognition is addressed with the help of an explicit semantic model, and in a dynamic scenario that requires continual learning. In the rest of this chapter we focus on a particular case that we refer to as *bookworm*[†] *continual learning* (BCL) where the semantic model remains fixed while the visual model is updated continuously (see Fig. 5.1(c)). BCL can be seen as a generalization of CL which is limited by lacking explicit semantic models, and ZSL which is not continual (see Table 5.1). One important challenge in BCL is the effective integration of semantic models and CL.

In addition, we propose a unified BCL framework via feature generation and distillation. In particular, feature generation is a suitable framework to integrate CL and ZSL capabilities since it exploits the same mechanism to prevent forgetting in CL and imagine unseen categories in ZSL.

Essentially, a generative model (a conditional VAE in our case) learns the distribution of features of past classes (in CL) and future classes (in ZSL via the semantic model) and *generates* synthetic features of those classes so a joint classifier on all classes can be trained. For BCL, we generate features of both past and future classes simultaneously.

5.2 Related work

5.2.1 Zero-shot learning

The objective of Zero-shot learning (ZSL) is to perform classification of unseen classes, connected to the seen classes via a semantic model that leverages shared semantic information (typically attributes or word embeddings). The most common approach is to jointly align visual and class representations in a common space. This space can be the semantic space [3, 45], the visual space [203] or an intermediate latent space [205]. Other approaches use convex combination of seen embeddings [129], combination of synthesized classifiers [23] and attention mechanisms [207].

The more challenging Generalized zero-shot learning (GZSL) problem [24] evaluates on the union of seen and unseen classes, where the challenge is to cope with the inherent bias towards seen classes. Liu *et al.* [102] reduces the bias using score calibration. Feature generation approaches [122, 188, 206] generate synthetic

[†]We use an avid reader (i.e. the *bookworm* stereotype) as a metaphor, due to his/her extensive encyclopedic prior knowledge about concepts (e.g. their descriptions) before eventually observing them visually.

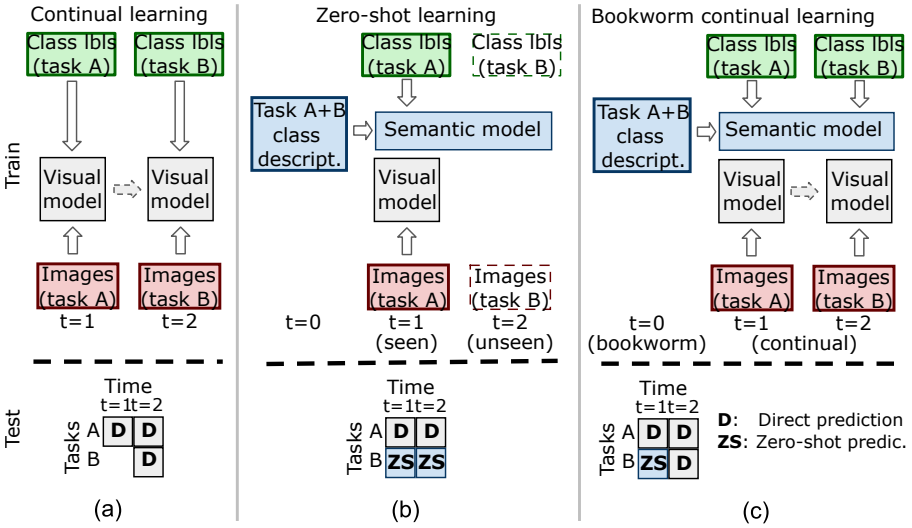


Figure 5.1 – Generalized continual learning: (a) continual learning, (b) zero-shot learning, and (c) bookworm continual learning.

features of unseen classes using a generative model learned with seen classes. A balanced classifier for all classes can be trained combining real and synthetic features.

In this chapter, we differ from previous ZSL works in two aspects. First, our visual model is not static and is updated continuously with new visual data. Second, we study the interplay of the semantic model and the problem of catastrophic forgetting which is specific to continual learning (CL).

5.2.2 Continual learning

Continual learning (a.k.a. lifelong learning) addresses the problem of continuously acquiring new knowledge from data that arrives over time following varying distributions. The main challenge lies in that learning new knowledge under those conditions interferes with previously learned knowledge, resulting in its forgetting [117]. This problem has been addressed with different techniques, including weight regularization [5, 76, 103], distillation [94], episodic memories with exemplars [108, 142]and generative replay methods [155, 183].

The most common evaluation setting involves the knowledge of the task, i.e.

Setting	Sub-models		Continual		Predictions	
	Visual	Sem.	Visual	Sem.	Seen	Unseen
JT	✓	✗	✗	-	✓	✗
CL	✓	✗	✓	-	✓	✗
ZSL	✓	✓	✗	✗	✗	✓
GZSL	✓	✓	✗	✗	✓	✓
BCL	✓	✓	✓	✗	✓	✓
GCL	✓	✓	✓	✓	✓	✓

JT: joint training, **CL**: continual learning, **ZSL**: zero-shot learning
GZSL: generalized ZSL, **BCL**: bookworm CL, **GCL**: generalized CL

Table 5.1 – Comparison of conventional and extended visual recognition settings (with semantic model and continual update).

task-aware, but the task identifier could also be unknown, i.e. task *task-agnostic*. Many methods performing well under task-aware evaluation show poor performance in task-agnostic evaluations, since the joint classifier tends to be biased towards the new learned task [63, 184] (note the similarity with the bias problem in GZSL). This problem can be addressed by saving a small set of data from previous tasks (episodic memory approach) or generating synthetic data from a model of previous tasks learned previously (generative replay approach).

In this work, we also handle the problem of biased classifiers using generative replay, but simultaneously enabling prediction over future classes, which CL lacks. In contrast to CL generative replay methods, our generator is hierarchically integrating a description generator prior to a feature generator, and generates samples of both past and future classes.

5.3 Bookworm continual learning

5.3.1 Bookworm and generalized continual learning

We assume a sequence of image classification tasks $(S_1, \dots, S_K)$. Each task is learned from a dataset $\mathscr{S}_k = \left\{\left(\mathbf{x}_i^k, \mathbf{a}_i^k, y_i^k\right)_{i=1}^{N_k}\right\}$, where $\mathbf{x}_i^k \in \mathscr{X}_k$ is an image, $y_i^k \in \mathscr{Y}_k \subset \mathscr{Y}$ is the corresponding class label and $\mathbf{a}_i^k \in \mathscr{A}_k \subseteq \mathscr{A}$ is the semantic description. We are ultimately interested in learning and continually updating a visual model $p_t\left(y|\mathbf{x}\right) = C_t\left(F_t\left(\mathbf{x}\right)\right)$ that maps images to class probabilities, where $\mathbf{z} = F_t\left(\mathbf{x}\right)$ and $p_t\left(y|\mathbf{z}\right) = C_t\left(\mathbf{z}\right) = \text{softmax}\left(W_t^\intercal \mathbf{z}\right)$ are the visual feature extractor and the classifier at time t, respectively (all implemented jointly as a deep neural network). For simplicity, we assume that k-th task is learned at time $t = k$ and will use t and k interchangeably.

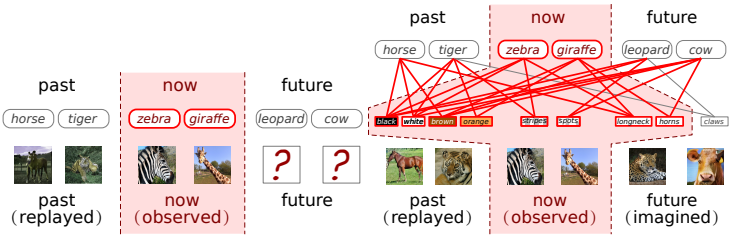


Figure 5.2 – Replay, imagination and semantic information shared across tasks. Note that in practice we generate features, not images.

In addition to the visual model, we also have access to a semantic model that describes classes in terms of semantic descriptions such as attributes[‡]. In particular, we consider that each class is described by an attribute vector as $\mathbf{a} = Ay$, where rows of the attribute matrix A correspond to attribute vectors of classes. The semantic model is learned or annotated from an external source (e.g. class descriptions, taxonomy, Wikipedia), and can be leveraged to help infer classes, including unseen ones, whose instances might have not been observed yet (but their descriptions have). In our experiments we will consider attributes for the semantic model but our theory could be applied to other semantic models. The visual model is always updated over time. In BCL the semantic model is learned prior to the visual model during a *bookworm* stage (at $t = 0$, for simplicity, see Fig. 5.1(c) and Table 5.1). And we assume *task-agnostic* evaluation, i.e. during test the task is unknown and the model has to consider all classes for the prediction.

5.3.2 Zero-shot learning and continual learning

Zero-shot learning (ZSL) can be seen as the particular case of BCL with two tasks and no update after the first one. Using ZSL terms, the first task is seen and the second unseen, i.e. $\mathcal{Y}_1 = \mathcal{Y}_{\text{seen}}$, $\mathcal{Y}_2 = \mathcal{Y}_{\text{unseen}}$. The model is evaluated on $\mathcal{Y}_{\text{seen}}$, which can be inferred using the semantic model. Generalized ZSL (GZSL) corresponds to task-agnostic evaluation, i.e. over $\mathcal{Y}_{\text{seen}} \bigcup \mathcal{Y}_{\text{unseen}}$.

Continual learning (CL) corresponds to the particular case where no semantic model is available, and therefore at time t the model can only discriminate between all the classes seen so far, which we denote as $\mathcal{Y}_{\leq t} = \bigcup_{k=1}^{t} \mathcal{Y}_k$.

[‡]For simplicity, we assume classification tasks and attribute-based semantic models, but our discussion is also valid for any other fixed-size continuous semantic embeddings (e.g. word embeddings, language embeddings).

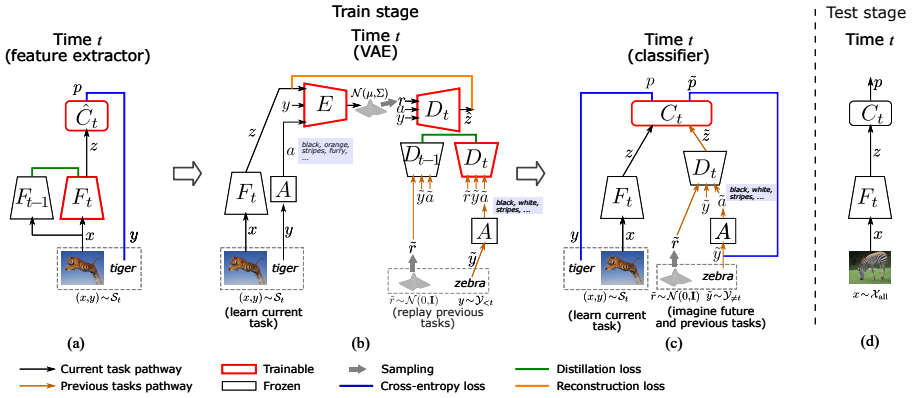


Figure 5.3 – Bidirectional imagination framework (*BImag*): (a-c) training stages (feature extractor, VAE and classifier), and (d) test stage.

Finally, if we further assume no continual update we recover the usual setting where the model is learned with all the data $\mathscr{S} = \bigcup_k \mathscr{S}_k$ (we refer to it as *joint training* (JT)).

5.4 BImag: a feature generation framework for BCL

To address BCL we need to cope with three challenges: (a) *catastrophic forgetting* in the shared feature extractor, (b) *bias* in the classifier (to the most recent observed data), and (c) a way to *predict future classes* (via semantic information). Here (a) is related to CL, (c) to GZSL and (b) to both.

5.4.1 Generative replay and imagination

We address catastrophic forgetting using generative replay [155]. A generative model captures the generative distribution $p(\mathbf{x}|y)$ from samples of the current task, and then samples from it in future tasks (see Fig.5.2a). These synthetic samples are combined with the current real samples, thus alleviating forgetting. However, image generation is a challenging problem, requiring large generative models that are also difficult to train. In contrast, feature generation is an easier problem, requiring smaller generative models and more efficient and effective in preventing forgetting [106]. Thus, our framework uses feature generation to model $p(\mathbf{z}|y)$, combined

Algorithm 2: BImag model update at time t.

Input : current data $\mathscr{S}_t$
Input : feature extractor params ψ_{t-1}, decoder params ϕ_{t-1}, semantic model $p_{\mathbf{a}|y}$
Output : ψ_t, ϕ_t, W_t (classifier params)

1 $\psi_t \leftarrow \psi_{t-1}$, $\phi_t \leftarrow \phi_{t-1}$
2 (optional) Update ψ_t by minimizing $\mathscr{L}_{\widehat{\text{CE}}} + \lambda \mathscr{L}_{\text{FEd}}$.
3 Update ϕ_t and learn θ by minimizing $\mathscr{L}_{\text{VAE}} + \lambda_2 \mathscr{L}_{\text{RA}}$.
4 Update W_t by minimizing $\mathscr{L}_{\text{CE}}$.

with feature distillation to prevent forgetting in the visual feature extractor.

In BCL we also have an additional intermediate level of abstraction, i.e. attributes (see Fig.5.2b). Considering now attributes, we can factorize the previous feature generative distribution as

$$p\left(\mathbf{z}|y\right) = p\left(\mathbf{z}|\mathbf{a}, y\right) p\left(\mathbf{a}|y\right) \tag{5.1}$$

where $p\left(\mathbf{z}|\mathbf{a}, y\right)$ corresponds to the *feature generator* and $p\left(\mathbf{a}|y\right)$ to the *attribute generator*, resulting in a hierchical feature generator. In our case, the attribute generator is deterministic, and sampling from a class y boils down to $\mathbf{z} = Ay$ (see Fig. 5.3).

In contrast to feature generation in continual learning, our generator is bidirectional, i.e. generates synthetic features of both previous (i.e. replay or recall) and future classes (i.e. imagination). Hence, we loosely refer to our framework as *bidirectional imagination* (*BImag*). This allows to train always a classifier with all classes, and thus predict any category at any time, while also allowing for continual updates.

5.4.2 BImag framework

Overview.

The BImag framework consists of a feature extractor and a classifier, together with a variational autoencoder (VAE) which implements the feature generator. To prevent forgetting via distillation, we also keep frozen copies of the previous feature extractor and the previous decoder of the VAE. The model is trained in three steps (see Fig. 5.3 and pseudocode in Algorithm 2). Semantic information (i.e. attributes) is only used during training, while the inference model is a direct mapping from image to class, without mapping to any intermediate semantic space.

In a first stage to learn a new task at time t (see Fig. 5.3-(a)), the feature extractor $F_t(\mathbf{x}) = F(\mathbf{x};\psi_t)$ is updated using an auxiliary classifier $\hat{C}_t(\mathbf{x}) = \hat{C}(\mathbf{x};\hat{W})$ minimizing the cross-entropy loss over the current task:

$$\mathscr{L}_{\widehat{\text{CE}}}(\psi_t, \hat{W}|\mathscr{S}_t) = \mathbb{E}_{(\mathbf{x},y)\sim\mathscr{S}_t}\left[\mathscr{CE}\left(\hat{C}_t(F_t(\mathbf{x})), y\right)\right] \tag{5.2}$$

Given the task data $\mathscr{S}_t$, forgetting is alleviated by distilling on $\mathscr{X}_t$. in particular we apply l_2 loss between the current feature extractor F_t and a copy of the previous one F_{t-1}, computed over the images of the current task:

$$\mathscr{L}_{\text{FEd}}(\psi_t|\mathscr{X}_t, \psi_{t-1}) = \mathbb{E}_{x\sim\mathscr{X}_t}\left[\|F_t(x) - F_{t-1}(x)\|^2\right] \tag{5.3}$$

Training a joint classifier requires input data for all classes. Since only current features are available, synthetic features are hallucinated for past and future classes using a hierarchical feature generator, where an attribute vector is first sampled given the class label (i.e. $\mathbf{z} = Ay$), and features are then sampled given the attribute vector and class (i.e. $\mathbf{z} \sim p(\mathbf{z}|\mathbf{a}, y)$). We train a conditional variational autoencoder with an encoder $[\mu, \Sigma] = E(\mathbf{z}, \mathbf{a}, y; \theta)$ (that estimates the parameters of the multivariate Gaussian latent distribution), and a decoder $D_t(\mathbf{r}, \mathbf{a}, y) = D(\mathbf{r}, \mathbf{a}, y; \phi_t)$, where D is parameterized by ϕ_t and taking $\mathbf{r}, \mathbf{a}, y$ as inputs. ($\mathbf{r}$ is a random latent vector sampled from the latent multivariate Gaussian distribution, i.e. $\mathbf{r} \sim \mathscr{N}(\mu, \Sigma)$). In VAE, the latent multivariate Gaussian distribution is assumed to be uncorrelated along dimensions, thus the covariance matrix Σ is a diagonal matrix and μ is the mean value. The decoder will act as feature generator. To learn the parameters θ and ϕ_t we maximize the evidence lower bound (ELBO) over current data $\mathscr{S}_t$ by minimizing

$$\begin{aligned}\mathscr{L}_{\text{VAE}}(\phi_t, \theta|\mathscr{S}_t, A) = \mathbb{E}_{(\mathbf{x},y)\sim\mathscr{S}_t}\Big[&-\mathscr{KL}\left(\mathscr{N}(\mu, \Sigma), \mathscr{N}(0, \mathbf{I})\right) \\ &+\mathbb{E}_{\mathbf{r}\sim\mathscr{N}(\mu,\Sigma)}\left\|\mathbf{z} - D_t(\mathbf{r}, Ay, y)\right\|^2\Big]\end{aligned} \tag{5.4}$$

The feature extractor remains fixed during this stage, and the encoder is learned from scratch every time. In addition, we include the replay alignment loss [183] between the past decoder D_{t-1} and the current decoder D_t, which is a form of distillation to prevent forgetting in the feature generator.

$$\begin{aligned}&\mathscr{L}_{\text{RA}}(\phi_t|\mathscr{X}_t, \phi_{t-1}, A) = \\ &\qquad\mathbb{E}_{\substack{\mathbf{r}\sim\mathscr{N}(0,\mathbf{I}) \\ y\sim\mathscr{Y}_{<t}}}\left[\left\|D_t(\mathbf{r}, Ay, y) - D_{t-1}(\mathbf{r}, Ay, y)\right\|^2\right]\end{aligned} \tag{5.5}$$

Once the VAE is trained, the decoder is used to generate a set of synthetic features $\tilde{\mathscr{S}}_{\neq t}$ for both past and future classes simply conditioning on attributes and

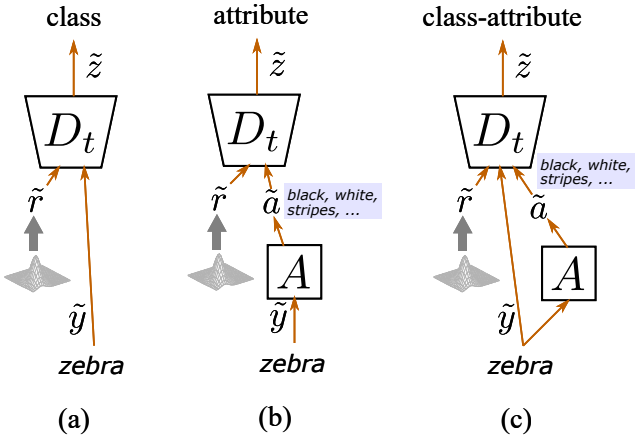


Figure 5.4 – Different conditional feature generators: (a) class (i.e. continual learning), (b) attributes, (c) class and attributes.

classes (). The classifier C_t is trained with both real and synthetic features (see Fig. 5.3-(c))), i.e. $\mathscr{S}_t \bigcup \tilde{\mathscr{S}}_{\neq t}$ using the cross-entropy loss.

$$\mathscr{L}^{\mathrm{CE}}\left(\phi_t, W_t | \mathscr{S}_t, \tilde{\mathscr{S}}_{\neq t}\right) = \mathbb{E}_{(\mathbf{z},y)\sim \mathscr{S}_t \bigcup \tilde{\mathscr{S}}_{\neq t}}\left[\mathscr{CE}\left(C_t\left(\mathbf{z}\right), y\right)\right] \tag{5.6}$$

Recovering CL and GZSL

If we consider the trivial attribute matrix $A = \mathbb{I}$ the only effective condition is the class label. Since in that case there are no shared attributes, future classes cannot be sampled. This case boils down to conventional continual learning (see Fig. 5.4a).

Similarly, BImag at $t = 1$ corresponds to GZSL to the initial step of BCL, where the first task corresponds to training with seen classes, and the model is then tested on all classes. Usual approaches to GZSL with feature generation approaches for GZSL [122, 188, 189] condition the generator only on the attribute vector (see Fig. 5.4b). In contrast, we condition it also on the class, which we found to also benefit GZSL performance.

5.4.3 Conditioning and forgetting

It turns out that conditioning on both class and attributes is even more important in BCL. Fig. 5.4 shows different variants of BImag depending on the condition.

Method	Generator	Condition	FE	CUB				AWA				SUN			
				seen	unseen	H	AUTAC	seen	unseen	H	AUTAC	seen	unseen	H	AUTAC
BImag	VAE	attr	fix	60.84	39.70	48.05	0.347	72.28	**62.02**	**66.76**	0.540	39.30	21.11	27.47	0.221
	VAE	attr	ft	**77.74**	41.30	53.94	0.484	73.83	59.97	66.18	0.555	41.94	22.43	29.23	**0.245**
	VAE	attr+class	fix	59.28	40.97	48.45	0.349	74.20	54.18	62.63	0.453	40.08	20.28	26.93	0.226
	VAE	attr+class	ft	73.57	45.09	55.91	**0.515**	**76.93**	51.40	61.63	**0.578**	40.73	21.67	28.23	0.231
Mishra *et al.*	VAE	attr	fix	-	-	34.5	-	-	-	51.2	-	-	-	26.7	-
f-CLSWGAN	GAN	attr	fix	57.7	43.7	49.7	-	61.4	57.9	59.6	-	42.6	36.6	39.4	-
f-VAEGAN-D2	VAE, GAN	attr	fix	60.1	48.4	53.6	-	70.6	57.6	63.5	-	45.1	38.0	41.3	-
	VAE, GAN	attr	ft	75.6	**63.2**	**68.9**	-	76.1	57.1	65.2	-	**50.1**	**37.8**	**43.1**	-

Table 5.2 – Experiments on GZSL (accuracies in %) and related works using feature generation. H refers to the harmonic mean and AUTAC is the area under the task accuracies curve.

Conditioning only on attributes, we observed that impairs the capacity of the generator to prevent forgetting, that is, attributes, rather than helping, are harming the ability to learn new tasks and prevent forgetting previous ones. In that case, we are implicitly assumming that $p\left(\mathbf{z}|\mathbf{a}, y\right) = p\left(\mathbf{z}|y\right)$ in Eq 5.1, i.e. generated features do not depend directly on the class label, only indirectly through the attributes.

This effect is probably due to two factors. First, when training the VAE, the same encoder and decoder observe the same attribute vector for all the instances of the same class (i.e. *class-level* description). However, the visual feature varies across instances, so the inconsistency between class-level description and visual instance can be confusing (e.g. the class cow can be described as being black, white and brown, and having spots, while there are instances where cows are just black without spots, or just white with brown spots). This prevents the VAE from learning the intra-class diversity by ignoring the different modes of the class distribution, and thus generating non realistic features. Secondly, attribute descriptions are also imperfect, not fully capturing all relevant visual features. Thus, conditioning only on class forces the VAE to learn the underlying feature distribution only from visual information. Together with attributes provides another path of dependency between class and visual feature, allowing the VAE to ignore attributes if necessary to avoid inconsistencies and also capture discriminative visual information not represented in the attribute space.

5.5 Experiments

We evaluate our approach with the Caltech UCSD Birds 200 (CUB), Animals with Attributes 2 (AwA) and SUN datasets on the GZSL, CL and BCL settings. Code and dataset splits are provided on *https://github.com/wangkai930418/bookworm_continual_learning/*.

5.5.1 Settings

Notation

For convenience, we use $t = 1, 2, \ldots$ to index time and $k = A, B, \ldots$ to index tasks. We assume that the k-th task is learned at time $t = k$.

Datasets and splits.

CUB is a fine-grained recognition dataset with 200 classes [170], and SUN [134] has 717 fine-grained classes, while AwA has 50 coarser classes [187], which are commonly used in ZSL. For experiments with class-level descriptions, we follow the settings and preprocessing used in conventional ZSL methods. We use the CUB, AwA and SUN data, attribute matrices, class splits and train/test splits proposed by [187], adapting them to our BCL setting. This results in two tasks A/B with class splits 150/50 for CUB, 40/10 for AwA and 645/72 for SUN (in ZSL task A/B are referred to as seen/unseen classes, respectively). Since there is no training for task B in ZSL, we created our own train/test splits for task B. We further split task A and created three task splits 100/50/50 for CUB and 30/10/10 for AwA.

Implementation details.

Our implementation is based on PyTorch and trained using NVIDIA GTX 1080Ti GPUs. The feature extractor in our model is a Resnet-101 [57], as commonly used in previous works in ZSL, and then fine tuned every new task as typically done in CL. Our conditional VAE consists of an encoder with three fully connected layers and a decoder with two fully connected layers. The conditions can be attribute vectors and/or class labels as one-hot vectors according to the specific BImag variant (Fig. 5.4). To train the joint classifier (Fig. 5.3-(c)), we generate 300 synthetic features per class for both past and future classes.And the classifiers are trained with the commonly used cross-entropy loss as in most deep learning classification models (such as ResNet [57], VGG [159]). We set $\lambda_1 = 1$, $\lambda_2 = 0.1$. We use Adam optimizer [75] with learning rates 0.0001 for the feature extractor and 0.001 both for classifier and VAE.

In these three different training stages as shown in Fig. 5.3, we train the feature extractor, VAE and joint classifier respectively. Here the feature extractor is the ResNet-101 model, VAE consists of a pair of encoder and decoder both with two fully-connected layers, and the joint classifier is composed with a linear layer (with a total of 170MB (ResNet-101) + 16.5MB (CVAE depending on conditions) trainable parameters). The same number of layers for the VAE is used in [64, 106] also for feature generation, where they obtained optimal results with this setting. And the ResNet-101 is commonly chosen as the backbone in generalized zero-shot

learning [186, 188–190].

Baselines and variants.

We use *BImag (only class)* with fine tuned feature extractor, distillation and replay alignment as main CL baseline. We extend this baseline with different semantic models to the BCL variants *BImag (only attr)* and normal *BImag*. Note that BCL methods at step $t = 1$ correspond to GZSL. We also compare with four popular CL methods (LwF, EWC, MAS and BiC) in the ablation study.

Metrics.

Following the common practice in GZSL, we use the *harmonic mean* of per-task mean class-accuracies [187] at a particular time t of BCL. We also compute the overall mean class-accuracy (including past, present and future classes). To evaluate how a particular approach is able to make predictions for any task or class at any time, which is the main objective in BCL, we compute their averages across time, i.e. *mean of harmonic means* (MH) and the *mean of overall means* (MO) (averaged over 5 runs).

While *harmonic mean* (MH) is the standard evaluation metric used in generalized zero-shot learning [187], Changpinyo *et al.* [23] have shown that it can be misleading as it depends on the relative scores of seen and unseen classes, and requires them to be calibrated properly. As a result, they concluded MH to be a confusing evaluation metric. Hence as an alternative, they propose characterizing a GZSL method by its seen-unseen curve obtained by varying a calibration parameter γ as

$$\hat{y} = \arg\max f_c(x) - \gamma \mathbb{I}[c \in \mathscr{U}] \tag{5.7}$$

The area under such curve, i.e. area under the task-accuracies curve (AUTAC), is proposed as evaluation metric. A calibrated HM could also be a suitable metric, but that requires observing unseen data (it could be a held out set of unseen classes different than the used for test).

For BCL we adapt the seen-unseen curve as task-accuracies curve, and use the area under the task-accuracies curve (AUTAC) as evaluation metric.

$$\hat{y} = \arg\max f_c(x) - \gamma \mathbb{I}[c \in \mathscr{T}_2] \tag{5.8}$$

To evaluate three tasks, the curve becomes a surface, therefore we propose using the volume under the task-accuracies surface (VUTAS) as evaluation metric. In

	FE	FEd	RA	Harmonic mean (%)			Overall accuracy (%)			AUTAC		
				$t=1$	$t=2$	MH	$t=1$	$t=2$	MO	$t=1$	$t=2$	Mean
SFT	ft			0.00	55.52	27.76	61.73	51.90	56.81	0.0205	0.5057	0.2631
LwF	ft	✓		0.00	66.55	33.28	61.73	62.20	61.96	0.0205	0.4768	0.2487
EWC	ft	✓		0.00	74.01	37.01	61.73	73.45	67.59	0.0205	0.6455	0.3330
MAS	ft	✓		0.00	74.81	37.41	61.73	73.13	67.43	0.0205	0.7297	0.3751
iCaRL	ft	✓		0.00	75.62	37.81	61.73	75.23	68.48	0.0205	0.7325	0.3765
BiC	ft	✓		0.00	**77.58**	38.79	61.73	**76.54**	69.13	0.0205	0.7478	0.3842
	fix			0.00	58.25	29.12	58.37	55.88	57.12	0.3485	0.3098	0.3792
	fix		✓	0.00	61.50	30.75	58.37	62.11	60.24	0.3485	0.4269	0.3877
BImag (only class)	ft			0.00	57.27	28.63	61.73	53.63	57.68	0.4815	0.5148	0.4982
	ft	✓		0.00	75.43	37.72	61.73	74.10	67.92	0.4815	0.6746	0.5781
	ft	✓	✓	0.00	76.71	38.36	61.73	76.27	69.00	0.4815	0.6852	**0.5834**
	fix			48.05	44.38	46.22	55.55	42.67	48.41	0.3467	0.3924	0.3696
	fix		✓	48.05	55.97	52.01	55.55	55.38	54.77	0.3467	0.4860	0.4164
BImag (only attr)	ft			53.94	53.22	53.58	68.63	49.94	59.29	0.4843	0.4390	0.4617
	ft	✓		53.94	66.58	60.26	68.63	66.70	67.66	0.4843	0.6001	0.5422
	ft	✓	✓	53.94	75.03	**64.49**	68.63	74.21	**71.42**	0.4843	0.6695	0.5769

Table 5.3 – Ablation study on CUB 150/50 with various metrics.

general, for M tasks we obtain the (hyper)surface

$$\hat{y} = \arg\max f_c(x) - \Sigma_{t>1} \gamma_t \mathbb{I}[c \in \mathcal{T}_t] \tag{5.9}$$

where the surface is obtained by varying $\{\gamma_2, \dots \gamma_M\}$

5.5.2 Generalized zero-shot learning

We first evaluate our framework in the GZSL setting (equivalent to BCL at $t = 1$) and compare with recent GZSL methods with similar architectures as VAE or GAN. Table 5.2 shows the results for CUB 150/50, AwA 40/10 and SUN 645/72, including recent works using feature generators [122], f-CLSWGAN [188], f-VAEGAN-D2 [189], with either fixed (*fix*) or fine tuned (*ft*) feature extractor. Our BImag variants for comparison are conditioned on attributes or concatenation of attributes and class labels. We can see that finetuning (ft) of the backbone network leads to a large performance gain. Interestingly, conditioning on class labels in addition to attributes was beneficial in CUB but not in AwA and SUN. Although it is not our main objective, BImag achieves comparable results under Harmonic means and AUTAC metrics when compared to other zero-shot methods, especially on the AWA dataset.

	CUB 150/50			AWA 40/10			SUN 645/72		
	CL	GZSL/BCL		CL	GZSL/BCL		CL	GZSL/BCL	
conditions	cls	attr	cls-att	class	attr	cls-att	class	attr	cls-att
$t = 1$ (GZSL)	0.018	0.484	**0.515**	0.039	0.555	**0.578**	0.003	**0.245**	0.231
$t = 2$	**0.691**	0.670	0.685	0.917	0.914	**0.923**	**0.295**	0.205	0.292
Mean	0.355	0.577	**0.600**	0.478	0.735	**0.750**	0.149	0.225	**0.262**

Table 5.4 – Two tasks experiments (AUTAC metric) on CUB 150/50, AwA 40/10 and SUN 645/72.

5.5.3 Bookworm continual learning

Ablation study

Our ablation study in Table 5.3 shows the effect of different components of BImag feature extractor (*fix* or *ft*), distillation in feature extractor (*FEd*) and replay alignment in VAE decoder (*RA*). We also include the sequential fine tuning (*SFT*) and five continual learning methods: learning without forgetting (*LwF*) [94], Elastic Weight Consolidation (*EWC*) [76], iCaRL [142], Memory Aware Synapses (*MAS*) [5], and Bias Correction (*BiC*) [184]. *BiC* is considered a state-of-the-art replay method (see [113]). The quantitative results of these settings are shown in Table 5.3. Overall, our method BImag obtains the best performance in MH/MO and AUTAC mean metrics with a large margin over compared methods, which proves that BImag benefits the Bookworm continual learning. We could also observe that fine tuning the feature extractor is very helpful, especially when combined with distillation. Replay alignment also contributes with some additional gain. Also, our method outperforms the *BiC* method, which is not surprising since that method does not have zero-shot capability and therefore cannot predict the presence of unseen classes.

Overall performance

Table 5.4 shows the results for two tasks for the different variants of BImag in AUTAC metric on three datasets. The CL variant *BImag (only class)* cannot predict future classes, in contrast to the variants with semantic models (i.e. BCL variants). The lower performance at $t = 2$ of *BImag (only attr)* compared to *BImag (only class)* highlights the limitations of class-descriptions as only condition to replay features of previous classes, probably due to a poorer VAE model when visual instances were already observed. Augmenting the condition with the class label (i.e. *BImag*) significantly alleviates this problem and achieves the best performance in CUB 150/50 in AUTAC metric. In AwA *BImag* performs best in t=1,2 and also mean of AUTAC. On SUN, *BImag* is the best. In summary, BCL methods outperform CL (i.e.

	CUB 100/50/50			AWA 30/10/10		
	CL	GZSL/BCL		CL	GZSL/BCL	
conditions	class	attr	class-att	class	attr	class-att
$t = 1$	0.000	0.120	**0.123**	0.005	**0.236**	0.158
$t = 2$	0.008	0.220	**0.224**	0.056	0.323	0.310
$t = 3$	**0.395**	0.310	0.376	**0.730**	0.729	**0.730**
Mean	0.134	0.217	**0.241**	0.263	**0.429**	0.399

Table 5.5 – Three tasks experiments (VUTAS metric) on CUB 100/50/50 and AwA 30/10/10.

CL column) at initial times (thanks to the semantic model), while outperforming GZSL ($t = 1$ row) by updating the visual model over time. Overall, properly using semantic attribute information and class labels in our VAE component helps us to generalize continual learning and generalized zero-shot learning, which lead to better prediction in throughout time.

Three tasks experiments

We also evaluated our model on three tasks setting on AWA and CUB (Table 5.5) and observed similar trends under our proposed VUTAS metric, i.e. our methods consistently improve over time after observing new images, while the semantic models allow to predict unseen classes at any time. By contrast, continual learning only outperforms our bookworm proposal in the last time stage where $t = 3$, and during other time stages, our BCL method (independent of conditioning on attributes or class labels concatenated with attributes) consistently performs better than the continual learning baseline. Overall, *BImag* performs well on the CUB dataset and *BImag (only attr)* gets better performances on AWA dataset.

Confusion matrices

In order to illustrate the effect of the semantic model, we show the confusion matrices of *class-BImag* and *attr-BImag* for CUB 150/50. In *class-BImag* at $t = 1$ we can see that all new classes are misclassified as known classes (e.g. if the *zebra* class is unknown, a zebra image is likely to be classified as the most similar known class *horse*), while *attr-BImag* is able to predict them we reasonable accuracy. Compared with this GZSL, updating the visual model in $t = 2$ improves the accuracy on the unseen classes, and the time averaged matrix has higher accuracy and less confused predictions.

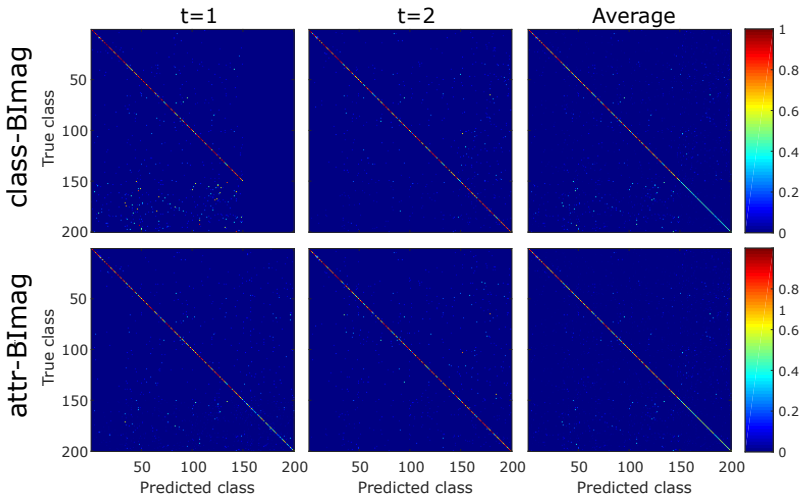


Figure 5.5 – Confusion matrices at $t = 1$, $t = 2$ and their average. Best viewed in electronic version with color using zoom. The results show that attr-BImage obtains superior results for classes 150-200 at t=1.

5.6 Conclusion

Intelligent systems should be designed and evaluated in controllable but realistic scenarios. Following that, in this chapter we proposed BCL as a novel setting to evaluate visual recognition where continual learning is augmented with an explicit semantic model. In BCL, we unify and generalize ZSL and CL. We go beyond ZSL, which cannot adapt to new knowledge, and beyond CL, which is limited by not using semantics; in contrast, BCL provides a more general setting, closer to human-like continual knowledge acquisition.

To address BCL, we proposed our BImag framework. More specifically, we focus on attributes annotations as the semantic information to enhance the hierarchical feature generation in both forward and backward temporal directions. Our method generates features of past classes to prevent forgetting, and features of future classes to perform zero-shot detection. We find that it is important to both condition the feature generator on the attributes as well as class information. In the experiments, we compare BImage with existing zero-shot and continual learning methods, and show that we outperform these with a wide margin; the results confirm that our model successfully prevents forgetting of past classes, as well as predicts future

classes based on the semantic model. We have designed a new set of experiments (as well as metrics) that, we hope, can serve the community to further explore the new research direction of generalized continual learning. For future work, we are interested in exploring dynamic semantic models which are continually improved during learning of new tasks and how to generalize the *Bookworm* concept to other deep learning tasks such as image retrieval [60], image segmentation [196], cross-model retrieval [172] and so on.

6 Continual learning in cross-modal retrieval*

6.1 Introduction

Human intelligence requires integrating, processing and comparing information from multiple modalities. Ideally, mental representations should lie in an abstract common space that is decoupled from the specific modality of the perceived information. Language and vision already interact in simple tasks such as object classification, where images are mapped to concepts in a closed vocabulary of categories. However, multimodal representations [14] allow for richer interactions enabling cross-modal tasks such as cross-modal retrieval [28, 31, 38, 176, 180], image captioning [32, 50, 132], visual question answering [29, 67, 128, 178], and more recently text-to-image synthesis [93, 201]. Language models are also useful to extend visual classification beyond the limited categories seen during training by projecting to language spaces, also known as zero-shot recognition [45, 192].

Another characteristic of humans is their ability for continual learning, which allows us to perform well tasks learned long back in time. In contrast, neural networks suffer from catastrophic interference [113, 117], which leads to almost complete forgetting of previous tasks when adapting to new ones, being a critical limitation to advance towards highly autonomous agents that can learn and adapt to changing environments. Continual learning (often referred to also as lifelong, sequential or incremental learning) in neural networks is an active research area, with recent methods addressing catastrophic forgetting with novel regularization [94, 104, 197], architectural [18, 114, 154] and (pseudo)-rehearsal [106, 155, 171, 183] mechanisms. Most continual learning methods focus on classification tasks.

Motivated by these two challenges, here we study continual learning in multimodal embedding spaces applied to cross-modal retrieval, and the specific problems that arise in this scenario. In continual learning, training (of new tasks) can happen at different points in time. In a retrieval scenario, we must consider also the indexing operation, where an embedded representation is extracted from the input sample and stored in the database for future comparison. Since indexing in a continual setting could also happen at different points in time, we pay special

*This chapter is based on a publication in the 2nd CLVISION workshop in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021 [172]

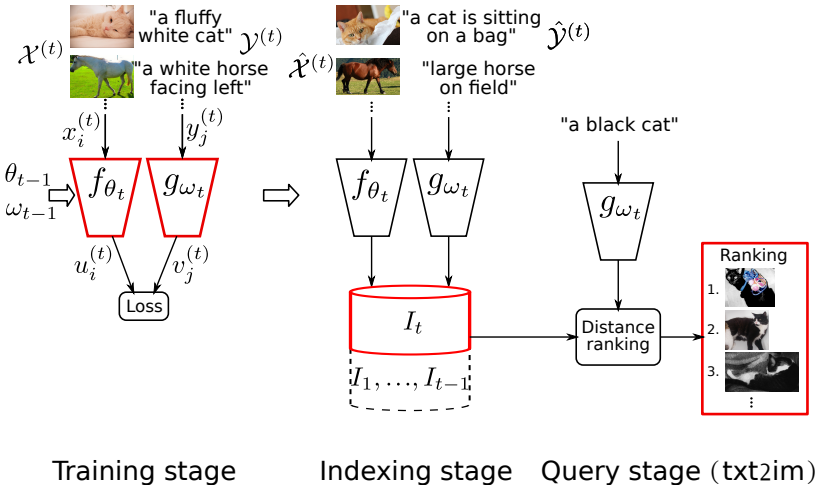


Figure 6.1 – Stages in continual cross-modal retrieval (i.e. training feature extractors, indexing and query). The output of each stage is highlighted in red (i.e. feature extractors, index and ranking, respectively).

attention to the role of this additional stage (see Fig. 6.1). An advantage of learning embedding networks instead of classification networks is that we operate in a single space shared by all tasks, so we can naturally retrieve data regardless whether we know or not the task related to that particular query sample (often referred to as task-aware and task-agnostic settings). Retrieval performance in continual learning is affected by how the embedded space may be distorted and cause representations to drift, as a result of the catastrophic interference. Additionally, these distortions and drifts may be unequal for each multimodality. Similarly, catastrophic forgetting affects differently to indexed data and query data.

In this chapter we propose a continual cross-modal retrieval framework that can effectively perform retrieval in known and unknown domains. We identify and study the different factors that lead to forgetting in cross-modal embeddings and retrieval. Addressing those factors, we study modifications in the retrieval framework, network architecture and regularization that can help to alleviate them.

6.2 Related Work

6.2.1 Deep metric learning

Deep metric learning learns both feature extraction and a distance metric in an end-to-end fashion. It maps images to an embedding space in which a simple distance metric such as the Euclidean distance can be applied. For training, it requires positive pairs (PP), which should be close in the embedding space, and negative pairs (NP), which are mapped at least a margin apart. Initial work was based on Siamese networks [19] which consist of two identical neural networks with shared weights, each taking one of the two inputs and map them to an embedding space. They are widely used in patch matching [158], face verification [151], image retrieval [48], etc.

Regarding the training loss, two of the most widely used are contrastive loss [52] and triplet loss [59]. The former continually pushes similar instances closer, whereas negative pairs are only required to be at least a margin away. In the latter, similar samples are only required to be closer to each other than to any dissimilar ones. The training of Siamese and triplet networks is known to be difficult. Especially, since many of the negative pairs are already far apart in embedding space, they do not result in any training signal. Therefore, it was shown to be important to perform hard negative mining [158]. Later works observed that it was computationally advantageous to first pass the images through a single network, and only form the pairs in the loss layer [105, 130]. Other losses include center loss [181] and proxy-NCA [123].

6.2.2 Cross-modal retrieval

Cross-modal retrieval requires a coordinated representation [14] that allows computing a similarity measure between the query representation and that of the retrieved data, even when they belong to different modalities and extracted with different feature extractors. There are two main aproaches to this problem: canonical correlation analysis (CCA) [62] and metric learning [83].

CCA [62] learns linear projections to a space where the projections of two random vairables are maximally correlated, which makes it attractive to cross-modal retrieval. CCA has also been extended to deep networks [10, 177], and in particular to cross-modal retrieval [40, 47, 78]. A limitation of CCA approaches is the expensive computation of the covariance matrix that requires having all data in memory.

Metric learning has also been applied sucessfully to cross-modal retrieval. Early examples of joint text-image embeddings are WSABIE [182] and DeViSE [45] which map image and text embeddings into a single space using ranking losses. Kiros *et*

al. [77] applied a similar approach to sentences using an LSTM model. Socher *et al.* [162] use an extended language model which includes dependency trees. Xu *et al.* [191] propose a joint representation for video and sentences. Two-branch networks [176] address image-text matching tasks with a bi-directional ranking loss. Multimodal representations have been also used for cross-modal retrieval of more structured visual-text documents, such as recipes [120, 150] and learning facts from images [38].

Efficient retrieval from large databases is also a concern, so cross-modal hashing [20] learns compact representations in binary spaces where indexing and retrieval can be performed efficiently. Cross-modal hashing has also been extended to deep models [22, 68].

6.2.3 Continual learning

A well known phenomenon in neural networks is catastrophic forgetting, where learning new tasks interferes with remembering previous ones [113, 117]. To enable networks to succeed in scenarios requiring continual learning, different techniques have been proposed.

A popular approach is to add regularization terms to the loss. Weight regularization methods [4, 76, 104, 198] add quadratic terms to penalize large differences to the solution for previous tasks, weighted by some importance measure so differences in more important parameters are penalized more. Elastic weight consolidation (EWC) [76] uses the diagonal approximation of the Fisher information matrix to estimate the importance. Rotated EWC [104] proposes a reparametrization that makes EWC more effective. Synaptic intelligence (SI) [198] estimates the importance measure during training by accumulating gradients. Memory aware synapses (MAS) [4] uses perturbation theory to estimate the importance in an unsupervised way. Forgetting can also be prevented by regularizing the activations, as in learning without forgetting (LwF) [94], where a snapshot of the network right before starting to learn the new task (and therefore not suffering interference from it) is used as a teacher and a distillation loss [58] is used during the training of the new task. Encoder-based lifelong learning [141] uses distillation in task-specific projections, estimated by autoencoders.

Another way of avoiding forgetting is rehearsal [142, 145], where a fraction of data (i.e. exemplars) from previous tasks is kept and revisited during training, and pseudo-rehearsal [11, 145], where pseudo-exemplars are sampled from an auxiliary model trained to model previous tasks.

Recent pseudo-rehearsal methods include deep generative models models [155, 183]. Other approaches to continual learning include networks that expand their capacity to allocate new tasks [149, 152, 194] and task-attention mechanisms [154].

While most works focus on classification, continual learning has also been studied in other settings such as image generation [126, 153, 183], word embeddings [72, 115], Atari games [76] and continual adaptation of agents [124]. MAS [4] is evaluated in facts learning that involves image and structured text. However, to our knowledge, there is not any wlighteork specifically studying the implications of continual learning in a retrieval setting, and catastrophic forgetting from the perspective of cross-modal embeddings.

6.3 Continual cross-modal retrieval

6.3.1 Cross-modal deep metric learning

Our framework is based on a two-branch network [176], with image-specific and text-specific embedding branches that project images and text into a common space. The image embedding operation is $u = f_{\boldsymbol{\theta}}(x)$, where $u \in \mathbb{R}^E$ is the image embedding of an input image x, extracted by the image embedding network $f_{\boldsymbol{\theta}}$, parametrized by $\boldsymbol{\theta}$. Similarly, the text embedding $v \in \mathbb{R}^E$ of an input text y is obtained as $v = g_{\boldsymbol{\omega}}(y)$ by the text embedding network $g_{\boldsymbol{\omega}}$ parametrized by $\boldsymbol{\omega}$. Both u and v are normalized using l_2 norm. Images and text are compared in the embedding space using the Euclidean distance as $d(x,y) = \|u - v\| = \left\| f_{\boldsymbol{\theta}}(x) - g_{\boldsymbol{\omega}}(y) \right\|$.

The image set $\mathscr{X} = \{x_i\}_{i=1}^{N_\mathrm{I}}$ is aligned with a text set $\mathscr{Y} = \{y_j\}_{j=1}^{N_\mathrm{T}}$ via a pairwise similarity matrix S. This cross-modal pairwise similarity is indicated by a variable $s_{ij} \in S$ which takes value 1 when x_i and y_j are similar (i.e. *positive pair*) and 0 otherwise (i.e. *negative pair*). We want the distance between positive pairs to be significantly lower than the distance between negative pairs. In order to do that we use the bi-directional ranking loss of [176], which selects triplets and imposes constraints

$$\begin{aligned} &d(x_i, y_j) + m \leq d(x_i, y_k) \\ &\quad \text{s.t. } s_{ij} = 1 \text{ and } s_{ik} = 0 \end{aligned} \tag{6.1}$$

and (in the other direction)

$$\begin{aligned} &d(y_{i'}, x_{j'}) + m \leq d(y_{i'}, x_{k'}) \\ &\quad \text{s.t. } s_{i'j'} = 1 \text{ and } s_{i'k'} = 0 \end{aligned} \tag{6.2}$$

where m is the predefined margin. The triplets are constructed based on a positive pair, and a negative pair creating by replacing either the image or the text by a dissimilar one. These triplet constraints are included using a margin-based loss

function (where $[z]_+ = \max(0, z)$):

$$\begin{aligned} L_{\mathrm{T}}(\mathcal{X}, \mathcal{Y}) = \lambda_1 \sum_{i,j,k} \left[d\left(x_i, y_j\right) + m - d\left(x_i, y_k\right) \right]_+ \\ + \lambda_2 \sum_{i',j',k'} \left[d\left(y_{i'}, x_{j'}\right) + m - d\left(y_{i'}, x_{k'}\right) \right]_+ \end{aligned} \tag{6.3}$$

6.3.2 Training, indexing and query stages

In general, machine learning assumes two different stages, namely *training* and *evaluation* (or *test*), which take place in that exact order (although in continual learning it is not the case). We focus on retrieval with a learned feature extractor (i.e. embedding networks in our case). In this scenario we identify three stages (see Fig. 6.1):

Training (feature extractors(s)). Described in the previous section, the training stage learns the embedding networks from the image and text datasets $\mathcal{X}$ and $\mathcal{Y}$, and its result is the parameters $\boldsymbol{\theta}$ and $\boldsymbol{\omega}$.

Indexing (database data). The database datasets $\hat{\mathcal{X}}$ and $\hat{\mathcal{Y}}$ to be indexed are processed using the embedding networks to obtain the text and image embeddings, which are subsequentially indexed in the database. Note that training data and database data are not required to be the same.

Querying (query data). This stage computes the similarity between a query sample and the indexed data. The result is a ranking with the most similar sample on top. In our cross-modal case there are two directions: querying with images, retrieving from indexed texts (*im2txt*) and querying with text, retrieving from indexed images (*txt2im*).

Note that these three stages are assumed to take place in that particular order, and a deployed system only performs the querying stage. For simplicity we consider that the database data is also used as training data, i.e. $\hat{\mathcal{X}} = \mathcal{X}$ and $\hat{\mathcal{Y}} = \mathcal{Y}$.

6.3.3 A framework for continual retrieval

Now we consider a continual learning setting, in which data is presented as a sequence of tasks $\left\{\mathcal{T}^{(t)}\right\}_{t=1}^{T}$. Each task $\mathcal{T}^{(t)} = \left(\mathcal{X}^{(t)}, \mathcal{Y}^{(t)}, \mathcal{S}^{(t)}\right)$ involves data from a different domain (e.g. animals, vehicles). We assume that the embedding networks are updated (i.e. fine tuned) with data of a particular task (i.e. training stage) before indexing data of that task. The resulting parameters after training task t are $\boldsymbol{\theta}_t$ and $\boldsymbol{\omega}_t$.

The retrieval system is evaluated in the querying stage with separate data from every task. We consider two settings for evaluation: *known task* and *unknown task*,

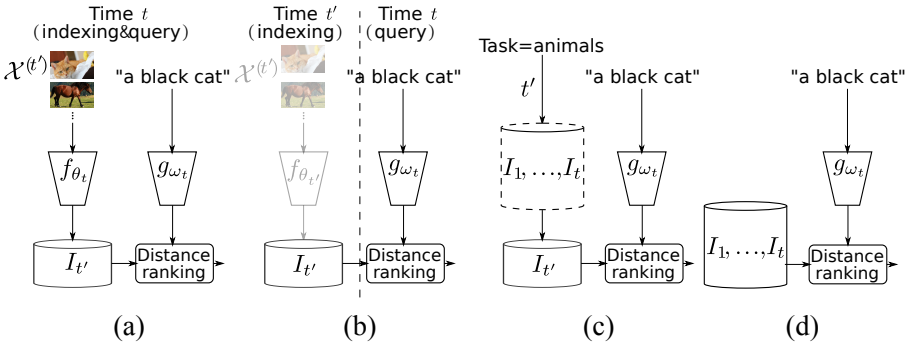


Figure 6.2 – Variants of indexing data from a previous task t' when queried at time $t > t'$ (a-b) and retrieval (c-d): (a) reindexing, (b) not reindexing, (c) task known, (d) task unknown.

depending on whether that information is available at query time (see Fig. 6.2a-b).

As described previously, the network is trained using cross-modal positive and negative pairs. When all data is presented jointly, all negative pairs are available for sampling. However, in the continual setting, pairs are formed within the same task, i.e. combining samples from $\mathcal{X}^{(t)}$ and $\mathcal{Y}^{(t)}$. Thus, we further classify a negative pair (x_i, y_j) as intra-task negative pair (ITNP), when $x_i \in \mathcal{X}^{(t)}$ and $y_j \in \mathcal{Y}^{(t)}$ ($s_{ij} = 0$, $s_{ij} \in \mathcal{S}^{(t)}$), or as cross-task negative pair (CTNP), when $x_i \in \mathcal{X}^{(t')}$ and $y_j \in \mathcal{Y}^{(t)}$, $t' \neq t$. Note that, for simplicity, we assume that all positive pairs are intra-task. In continual retrieval, CNTPs are not available during training (see Fig. 6.3).

6.3.4 Do or do not reindex?

The conventional retrieval scenario assumes that training and indexing are performed once. In this case, there is only a static set of embeddings, extracted with the same network at the same time. The same network is used to extract embeddings from queries. In continual retrieval this may not be the case, since training and indexing are performed every time a new task is presented.

Reindexing. We first consider the straightforward extension of cross-modal retrieval that assumes that current and previous tasks all are reindexed with the version of the embedding networks with updated parameters $f_{\boldsymbol{\theta}_t}$ and $g_{\boldsymbol{\omega}_t}$ after a new task t is learned (see Fig. 6.2a). We refer to this case as *reindexing*. However, it has the drawbacks of being time and resource consuming, since it requires indexing the same data multiple times, and always requiring access to the image and text samples of previous tasks. It has the advantage that database and query samples

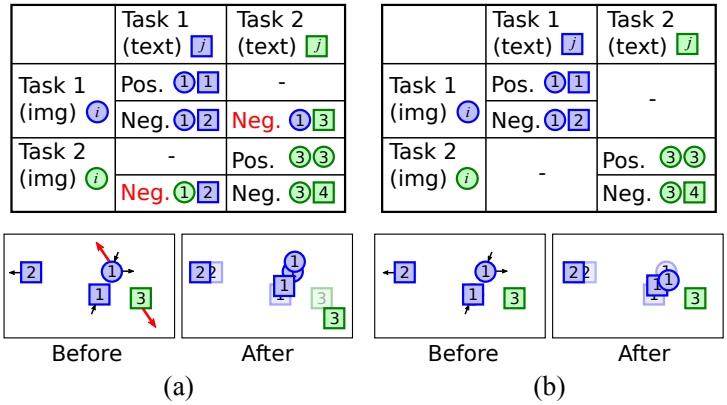


Figure 6.3 – Types of pairs in continual cross-modal retrieval: (a) available in joint training, and (b) available in continual learning, i.e. without cross-task negative pairs (CTNP). CTNPs are crucial to avoid overlap between samples of different tasks (bottom). Best viewed in color.

are processed with the same networks.

No reindexing. We also propose the variant *no reindexing* that only indexes the data of current task t after training task t (see Fig. 6.2). This variant is more efficient, since database samples are processed only once, and flexible since it does not require access to previous images and text (only to their indexed embeddings for retrieval). On the other hand, no reindexing introduces asymmetry, since query embeddings are extracted with $f_{\boldsymbol{\theta}_t}$ (or $g_{\boldsymbol{\omega}_t}$), while database embeddings with $g_{\boldsymbol{\omega}_{t'}}$ (or $f_{\boldsymbol{\theta}_{t'}}$, with $t' \leq t$).

6.4 Catastrophic forgetting in cross-modal embeddings

Learning a new task implies that the values of the network parameters will shift away from the previous ones. This is particularly important when the new task is very different from previous, causing interference between new and previous tasks that leads to lower performance in the latter. For simplicity, we will refer to this drop in performance as *catastrophic forgetting*. In the following, we identify several phenomena that may lead to forgetting in continual cross-modal retrieval.

Embedding networks. We first consider forgetting in each embedding network separately, without considering pairwise interactions. As their parameters move away from the optimal values for $t-1$ (see Fig. 6.4a), the embeddings u and v will

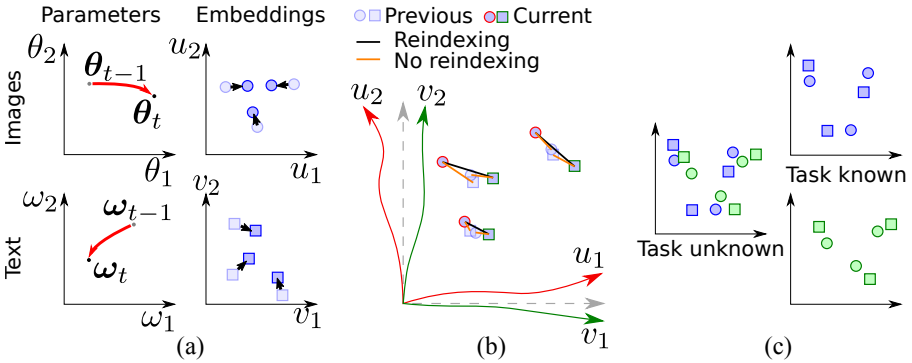


Figure 6.4 – Causes of forgetting in cross-modal embeddings: (a) embedding networks become less discriminative due to drift in parameter space, and (b) unequal drift increases cross-modal misalignment, and (c) task overlap in embedded space (when task is unknown). Best viewed in color.

also drift from their previous values. In general, the new values $f_{\boldsymbol{\theta}_t}$ and $g_{\boldsymbol{\omega}_t}$ are less discriminative than previous previous $f_{\boldsymbol{\theta}_{t-1}}$ and $g_{\boldsymbol{\omega}_{t-1}}$, causing lower performance, because the embedding spaces of $u^{(t)}$ and $v^{(t)}$ are also less discriminative.

Embedding misalignment. In the particular case of cross-modal networks, embeddings of different modalities may drift differently (see Fig. 6.4a). This unequal drift in u and v spaces causes additional misalignment that also leads to higher distances than in the optimal case. Note that unimodal retrieval with Siamese networks or Triplet networks does not suffer from this problem, since parameters are shared across de various branches.

Task overlap. Negative pairs pull dissimilar samples away in the embedded space. However, in continual retrieval CTNPs cannot be sampled (unless we include some samples from previous tasks). CNTPs are the only repulsive force between samples of different tasks. Without them, it is likely that samples from different tasks will overlap in the embedded space (see Fig. 6.4a). Knowing the task at query time makes this problem less important, since data from other tasks are not considered at query time.

6.5 Preventing forgetting

In the following we propose several tools to alleviate forgetting by addressing the previous causes.

6.5.1 Preventing embedding drift

A common approach to prevent forgetting is regularizing the weights with a quadratic term in the loss that penalizes the weighted Euclidean distance (in the parameter spaces) to the solution for previous tasks [4, 76, 104, 198]. This can help to avoid significant drift in the embeddings and to keep them discriminative for previous tasks. We can write the particular regularization term for our case as

$$\begin{aligned} L_{\mathrm{R}} = &\sum_{k} \Theta_k^{(t-1)} \left(\theta_k^{(t-1)} - \theta_k\right)^2 + \\ &+ \sum_{k'} \Omega_{k'}^{(t-1)} \left(\omega_{k'}^{(t-1)} - \omega_{k'}\right)^2 \end{aligned} \tag{6.4}$$

where Θ_k and $\Omega_{k'}$ control the regularization strength depending on the importance of $\boldsymbol{\theta_k}$ and $\boldsymbol{\omega_k}$, respectively, for previous tasks. During the training of the first task there is no regularization, i.e. $\Theta_k^{(0)} = 0$ and $\Omega_{k'}^{(0)} = 0$. The way to compute the importance differs in different methods. We consider two variants:

Global. Here we estimate the importance with respect to the loss, adapting elastic weight consolidation (EWC) to our particular triplet loss as (L_{TR} represents the triplet loss):

$$\Theta_k^{(t)} = \mathbb{E}_{x,y}\left[\left(\frac{\partial}{\partial \theta_k} L_{\mathrm{TR}}\left(\mathscr{X}^{(t)}, \mathscr{Y}^{(t)} | \boldsymbol{\theta}_t, \boldsymbol{\omega}_t\right)\right)^2\right] \tag{6.5}$$

which is computed by sampling triplets as in 6.1 and 6.2, and analogously for $\Omega_{k'}$. This loss already takes into account triplets and their interactions.

Branch. Instead of estimating importance values that depend on a joint loss, we consider regularizing each branch independently. In this case we estimate the importance using the approach memory aware synapses (MAS), which can be computed unsupervisedly for each branch with images or text. The importance for the image branch is estimated as:

$$\Theta_k^{(t)} = \Theta_k^{(t-1)} + \mathbb{E}_{x_i \sim \mathscr{X}^{(t)}}\left[\frac{\partial}{\partial \boldsymbol{\theta}_k} l_2^2(f_{\theta_t}(x_i))\right] \tag{6.6}$$

which is accumulated over previously computed one. For the text branch the estimation of $\Omega_{k'}$ is analogous. In this equation, l_2^2 is the squared l^2 norm of the function outputs, which is used to estimate the importance of parameters in MAS method.

The final loss combines (6.3) and (6.4) as $L = L_{\mathrm{T}} + \lambda_3 L_{\mathrm{R}}$.

6.5.2 Preventing unequal drift

In order to prevent unequal drift we propose tying the networks by sharing layers at the top (bottom layers must remain modality-specific). In this way, the unequal drift can be alleviated since the gradients are tied and only differ in the lower layers.

In some cases when the drifts in text and image embedding are in opposite directions, refraining from reindexing the database can be an effective tool to alleviate drift, since only one of the embeddings is affected while the other remains fixed. Fig. 6.4b illustrates how in that case no reindexing keeps matching pairs at lower distances.

6.5.3 Decoupling retrieval directions

So far we assumed only a single model is trained to perform both text to image and image to text retrieval. This is reasonable when embeddings are reindexed since the architecture and the loss are symmetric. However, when database data is not reindexed and query is, the forgetting is asymmetric. In that case we can decouple both directions and train one model for each direction, only regularizing the weights in the query branch. This can also be beneficial in some cases when the image and text embeddings drift in different directions, keeping one fixed in the previous position can keep the distance lower (see example in Fig. 6.4b).

6.5.4 Preventing cross-task overlap

The lack of CTNPs can lead to cross-task overlap, since there is no force separating them. However, reducing the drift, keeping the embeddings discriminative via weight regularization and sharing layers may indirectly help to keep tasks separated (we observed that in our experiments).

Nevertheless, we made some preliminary experiments creating pseudo-CTNPs (u_i, x_j) in models with decoupled retrieval directions using the already indexed embeddings (analogously for text to image retrieval for the other direction), but we found they did not help in our experiments, probably because the asymmetric force that only pushes the embeddings of one branch. In this case the gradients are only backpropagated through one branch. We leave their study more in depth for future work.

6.6 Experiments

Baselines and variants We evaluate the different variants of our continual cross-modal framework in two tasks involving images and text, one focusing on regions

and the other on scenes. We follow the implementation of the two-branch networks in [176] where 4096-dim image features are extracted from a VGG-19 model trained on ImageNet, and text features are 6000-dim from HGLMM features (reduced with PCA from initial 18000-dim) [78]. The image branch includes two additional fully connected layers with sizes 2048 and 64 (for SeViGe, 2048 and 512 for SeCOCO) on top and l_2 normalization, and the same for the text branch. We focus our study on the two fully connected layers on top, while the initial feature extractors remain fixed. As in [176] we set $\lambda_1 = 1.0$, $\lambda_2 = 1.5$ and the margin $m = 0.05$. The resulting model is trained with Adam [75] and a learning rate of 0.0001, and using dropout after ReLu with probability 0.5. We evaluate different variations of this architecture:

- **Joint vs continual.** We compare the variants of the proposed framework (*continual*) with two baselines that learn all tasks jointly (*joint*), differing on whether CTNPs are sampled or not during training.
- **Retrieval direction.** We evaluate both text to image retrieval (*txt2im*) and image to text retrieval (*im2txt*).
- **Task knowledge.** We evaluate both the cases where the task is *known* and *unknown*.
- **Reindexing.** We consider the embeddings for database samples are extracted when the corresponding task was learned (*no reindex*), or are at the same time as the query embeddings (*reindex*).
- **Weight regularization.** We consider fine tuning with no regularization (*ft*), with joint regularization on the loss (*EWC*) and with regularization on each u and v embedding independently (*MAS*). We set $\lambda_3 = 10^6$.
- **Decoupled directions.** For *no reindex* we also consider variants where EWC or MAS are only computed in the branch extracting query embeddings (e.g. *MAS-txt* when MAS is computed only on the text branch). In this case we run two different experiments, each specialized for one particular retrieval direction.
- **Layer sharing.** We consider keeping both embedding networks independent (*no sharing*) or sharing the top fully connected layer (*sharing*).

We consider experiments where each task consist in updating the embedding networks by learning a new domain. After training the model, the same training data is indexed (i.e. we extract image and text embeddings) and then the retrieval performance can be evaluated. We report the final results after all tasks are learned. We use Recall@K as evaluation metric (with $K = 10$), with respect to the indexed data of the same domain (*known*) or to the whole indexed data with all domains (*unknown*). We repeat each experiment five times and report the average.

6.6.1 Sequential Visual Genome

Sequential Visual Genome (SeViGe) dataset. We created a dataset based on the regions with object-description pairs in the Visual Genome dataset [80]. Based on the object categories of those regions, we selected pairs related with the domains *animals* (9 categories), *vehicles* (6 categories) and *clothes* (6 categories), which are learned in sequence as tasks in our experiments. Each task has a total of 10481, 7531 and 10200 training images, respectively, and additional 900/900, 600/600 and 600/600 for validation/test, respectively (100/100 per category).

Cross-modal retrieval. We evaluate the different methods in cross-modal region image-text retrieval. The results for both directions are shown in Table 6.1. We focus on *average* for evaluation when the task is known, and *A+V+C* to evaluate when the task is unknown (i.e. the aggregate of all domains). We first observe that training all tasks jointly delivers higher performance than the continual setting, as expected. Significant part of that superiority is due to CTNP, since the performance of joint training drops significantly when not sampled. This provides a more realistic and tighter upper bound, since in the continual scenario CTNPs are not available. This drop is more moderate when the task is known (~1-2% known, ~3-4% unknown), since task overlapping is not a problem. This drop still suggests that CTNPs still contribute to shape the embedding spaces to be discriminative beyond simply avoiding task overlap. When the top layer is shared, the drop is also smaller, although the overall performance is also lower than not sharing the layer.

Focusing on continual learning we observe that the single modification that most reduces forgetting is not reindexing the database, which provides 3% and 5.1% boosts in *im2txt* and *txt2im* directions, respectively (1.6% and 3.4% if task is unknown). This surprising result, showing that reindexing can be harmful, suggests that the misalignment caused by the unequal drift of image and text embeddings is more critical than the misalignment caused by not extracting embeddings at the same time, and that keeping good and discriminative representations in the database is also important (recall that in *no reindexing* only the query embedding has endured catastrophic interference). Note that this is specific to cross-modal retrieval because branches do not share parameters. This may not be the case in image-to-image or text-to-text retrieval with Siamese or Triplet networks because the embeddings of the two branches drift equally.

Sharing layers by itself gives an improvement of 1.4% in *im2txt*, while not having impact in *txt2im*. Not reindexing also gives a similar boost as in the previous case. Interestingly, sharing layers harms performance in joint training, while for the continual setting it improves the performance, probably because it reduces the unequal drift by tying the drift of both modalities at least in the shared layers.

Weight regularization has moderate impact and could harm the performance

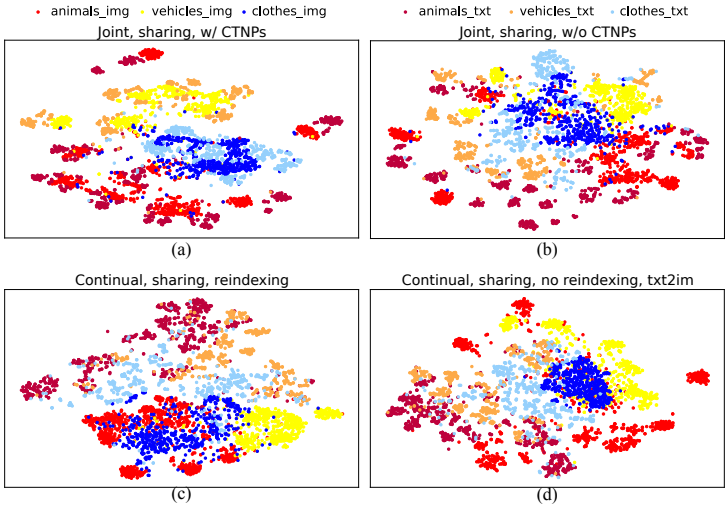


Figure 6.5 – t-SNE visualization of the cross-modal embedding space of SeViGe, with the *sharing* architecture: (a) joint training (with CTNPs), (b) joint training (without CTNPs), (c) continual (reindexing), and (d) continual (no reindexing). Best viewed in color.

sometimes. Decoupling both modalities and applying regularization only in the query network extractor seems to help in some cases (e.g. +1.3%/+1.6% gain with *MAS-txt* vs *MAS* in *txt2im*, and +0.7%/+0.9% in *im2txt* in the *sharing* architecture). In this dataset regularizing embeddings independently with MAS instead of the whole network with EWC seems to work better, although the differences are very marginal. Here we can see that the forgetting in embedding network is not a significant problem in cross-modal retrieval setting.

Overall, the best combination provides improvements of 6%/6.3% in known/unknown *txt2im* retrieval, and more moderate improvements of 2.9%/2% in *im2txt* retrieval.

Insights about the embedding space. We use t-SNE [110] to visualize the embedding space of variants with shared layers. Although distances in t-SNE do not reflect real distances, it is useful to identify structure. We combine text and image embeddings and runt-SNE, color coding data with modality and task labels. Joint training (see Fig. 6.5a) generates embeddings where data is structured clearly in separated tasks, and within tasks, in separated clusters (probably the categories within each task in SeViGe). This happens in both modalities, which also overlap,

Domain	*im2txt*										*txt2im*									
	Joint		*Continual*								*Joint*		*Continual*							
	CTNP		*reindexing*			*no reindexing*					*CTNP*		*reindexing*			*no reindexing*				
	Yes	No	*ft*	*EWC*	*MAS*	*ft*	*EWC*	*EWC-im*	*MAS*	*MAS-im*	Yes	No	*ft*	*EWC*	*MAS*	*ft*	*EWC*	*EWC-txt*	*MAS*	*MAS-txt*
Architecture: *no sharing*																				
animals	29.1	26.0	16.1	16.8	16.9	24.5	24.6	24.2	24.7	24.3	27.8	25.9	15.4	15.2	15.4	20.8	20.8	20.9	19.8	20.7
vehicles	30.9	27.7	20.8	23.3	22.7	24.0	25.1	24.8	26.0	24.8	30.9	27.0	17.5	18.6	19.5	27.2	29.4	28.0	28.8	28.7
clothes	27.9	27.5	27.4	27.0	27.5	27.4	27.0	27.3	27.5	26.3	29.3	27.7	28.1	27.5	28.0	28.1	27.5	27.4	28.0	28.5
average	**29.3**	27.0	21.5	22.3	**22.4**	24.5	24.6	24.2	**24.7**	24.3	**29.3**	26.8	20.3	20.5	**21.0**	25.4	25.9	25.4	25.6	**26.0**
A+V+C	**28.5**	24.4	17.0	**18.4**	17.8	18.6	17.9	17.5	**19.0**	18.3	**28.0**	23.8	16.3	16.3	**16.9**	20.7	21.3	20.9	20.9	**21.4**
Architecture: *sharing*																				
animals	28.3	25.3	18.4	17.1	16.4	23.1	21.2	21.4	21.1	21.4	26.8	24.4	16.6	14.8	14.3	22.1	20.7	21.1	20.6	22.2
vehicles	30.2	28.6	22.6	24.7	23.5	23.0	24.9	25.0	23.8	26.0	31.2	27.9	16.9	17.8	16.3	27.3	29.4	29.5	28.4	28.7
clothes	26.7	27.4	27.7	26.9	27.1	27.7	26.9	27.3	27.1	26.7	27.5	26.8	27.2	27.0	26.0	27.2	27.0	27.5	26.0	28.0
average	28.4	27.1	22.9	**22.9**	22.3	24.6	24.3	24.6	24.0	**24.7**	28.5	26.4	**20.3**	19.9	18.9	25.6	25.7	26.0	25.0	**26.3**
A+V+C	27.8	24.5	18.2	**18.2**	17.6	**19.0**	17.9	18.2	17.9	18.8	27.2	23.7	**15.9**	15.5	14.9	21.8	21.5	22.2	21.0	**22.6**

Table 6.1 – Results in SeViGe after learning all tasks (Recall@10 in %). *average* measures performance with *known* task, while *A+V+C* with *unknown* task. Best joint learning result in **green**, best continual learning result in **red**.

aligned according to the related clusters. Not sampling CTNPs still results in intra-task structure (see Fig. 6.5b, e.g. category clusters are clear), but the modalities are significantly more misaligned and with larger overlap, showing the important role of CTNPs in aligning modalities and separating tasks. When learned in a continual fashion (see Fig. 6.5c), the misalignment is more extreme, even resulting in text and image samples distributed in different halves of the space. No reindexing (see Fig. 6.5d for *txt2im* direction) seems to keep the image embeddings (database) more discriminative, which may explain the improved results compared to reindexing. For example, the image embeddings of animals and vehicles seem much better separated in Fig. 6.5d than in Fig. 6.5c.

6.6.2 Sequential MS-COCO

Sequential MS-COCO (SeCOCO) dataset. We created a second dataset with image-description pairs of MS-COCO [97]. Each image in MS-COCO is annotated with five image-level descriptions of the scene and a variable number of object annotations localized to specific regions and labeled with one of 80 disjoint object categories. Object categories are further organized in 12 disjoint super-categories. Organizing the data into tasks is challenging in this case since we want to avoid overlap between tasks, but there are many object annotations in each image. We organized the data into groups of super-categories and removed the images with object annotations in more than one group. After removing overlapping images we use those groups as tasks. We finally selected *animal*, *accessory*, *kitchen*, *food* and *furniture* for task 1, *vehicle*, *outdoor*, *electronic*, *appliance* and *indoor* for task 2 and *person* and *sports*

Domain	*im2txt*										*txt2im*									
	Joint		*Continual*								*Joint*		*Continual*							
	CTNP		*reindexing*			*no reindexing*					*CTNP*		*reindexing*			*no reindexing*				
	Yes	No	*ft*	*EWC*	*MAS*	*ft*	*EWC*	*EWC-im*	*MAS*	*MAS-im*	Yes	No	*ft*	*EWC*	*MAS*	*ft*	*EWC*	*EWC-txt*	*MAS*	*MAS-txt*
Architecture: *no sharing*																				
task1	65.7	63.8	33.6	32.0	33.0	49.8	48.1	47.2	50.5	47.1	69.7	68.2	40.1	38.0	38.2	59.8	59.2	58.3	60.0	59.7
task2	56.5	54.9	39.8	38.5	40.0	47.0	46.6	46.4	47.0	46.9	65.2	62.6	46.8	44.7	46.9	54.6	55.5	55.1	55.5	55.9
task3	38.2	39.9	39.7	40.1	40.2	39.7	40.1	39.9	40.5	39.7	44.6	45.7	46.7	46.7	46.0	46.7	46.7	46.7	46.0	46.2
average	**53.5**	52.9	**37.7**	36.9	**37.7**	45.5	44.9	44.5	**46.0**	44.6	59.8	58.9	**44.5**	43.1	43.7	53.7	53.8	53.4	53.8	**54.0**
total	**52.4**	49.8	**33.0**	32.1	**33.0**	37.1	36.2	35.6	**37.4**	36.0	58.5	56.3	**40.4**	38.7	39.7	48.3	48.0	47.3	48.2	**48.4**
Architecture: *sharing*																				
task1	65.3	63.9	32.9	31.9	34.1	48.4	47.7	47.7	47.8	45.1	70.2	67.7	38.2	37.4	39.8	58.6	56.3	58.4	57.1	57.5
task2	55.7	55.3	40.6	39.9	40.4	46.3	46.0	45.2	44.0	44.4	64.7	63.1	46.0	45.7	46.3	54.6	54.2	55.6	54.6	54.9
task3	37.6	40.1	39.6	39.7	39.3	39.6	39.7	39.9	40.0	39.7	44.8	46.5	46.2	45.8	45.7	46.2	45.8	45.7	46.7	46.1
average	52.9	53.1	37.7	37.2	**37.9**	**44.8**	44.5	44.3	43.9	43.1	**59.9**	59.1	43.5	43.0	**43.9**	53.1	52.1	**53.2**	52.8	52.8
total	51.8	50.1	33.2	32.5	**33.5**	**36.1**	35.9	35.4	35.5	35.3	**58.7**	56.4	39.3	38.9	**39.9**	47.7	46.8	**48.1**	47.1	47.5

Table 6.2 – Results in SeCOCO after learning all tasks (Recall@10 in %). *average* measures performance with *known* task, while *total* with *unknown* task. Best joint learning result in **green**, best continual learning result in **red**.

for task 3, with 22475, 13903 and 13919 training images respectively, in addition to 1000/1000 images for validation/test for each task. Note that many other objects and concepts remain unannotated, so there is still semantic overlap across tasks that we cannot control.

Cross-modal retrieval. Table 6.2 shows the recall@10 for different methods on SeCOCO. In this case joint training also performs better than continual learning methods. The drop due to not training with CTNPs is relatively lower than in SeViSe. This can be explained by a higher semantic overlap between tasks that makes CTNPs less critical. The relative importance of sharing layers is also less important in this case, with very little difference in the results.

Regarding continual learning methods, no reindexing is again the most helpful tool to prevent forgetting. Comparing with the *ft* baselines it gives important boosts of roughly 7-9%/3-8% in known/unknown tasks, for both sharing and not sharing layers. As in joint training, sharing layers does not have significant impact in this dataset. Similarly, weight regularization only brings marginal gains. In total, the best result for *txt2im* retrieval improves 9.5%/8% over the baseline for known/unknown tasks. For *im2txt* retrieval the improvement is 8.3%/4.4%. The results are still far from joint training, so there is space for improvement in future works.

6.7 Conclusion

In this chapter we propose, to our knowledge, the first study on how forgetting affects multimodal embedding spaces, focusing on cross-modal retrieval. We propose a continual cross-modal retrieval model that emphasizes the important role of the

indexing stage. Cross-modal drifts are also key factors in forgetting in cross-modal tasks. We evaluated several specific tools to alleviate forgetting.

7 Conclusions and Future Work

7.1 Conclusions

In this thesis, we have aimed to improve the performance of deep learning models in various applications, including meta-learning, hierarchical classification, zero-shot learning, cross-model retrieval and so on. For the continual learning of these various applications, we proposed corresponding solutions in this thesis:

- **Chapter 2: Episodic Replay Distillation for Incremental Meta Learning.** Previous attempts with exemplar replay failed to improve performance of incremental meta-learning. This is counter intuitive since exemplar replay is one of the most successful methods for incremental learning of classification problems. In this chapter, we showed how exemplar replay can be adapted for incremental meta-learning. We exploited the exemplars to perform cross-task meta-learning which improves the discriminative power of the learned representations. In addition, we also used exemplars to perform our proposed episodic replay distillation. Both contributions have been shown to considerably improve performance. Experiments on multiple few-shot learning datasets demonstrated the effectiveness of episodic replay distillation. Our method is especially effective on long task sequences, where we significantly close the gap between incremental few-shot learning and the joint training upper bound.
- **Chapter 3: Incremental implicit-refined classification.** In this chapter, we proposed a hierarchy-consistency verification module for the Incremental Implicit-Refined Classification (IIRC) problem. We showed how the hierarchical information can be learned in an incremental manner, and how this information can be beneficial at both training and testing time. With our module, we outperformed the existing incremental learning methods by a large margin. In our experiments on three different setups, we evaluated and confirmed the effectiveness of our proposed module during both training and inference. And from the visualization of confusion matrices, we found that our HCV module works as a denoising method to the confusion matrices.
- **Chapter 4: ACAE-REMIND for online continual learning.** In this chapter,

we proposed an extension to the REMIND method, called ACAE-REMIND. We proposed a stronger compression module based on an auxiliary classifier auto-encoder that allows to move the feature replay to lower layers. The method is memory efficient and obtains better performance. In evaluation, we perform a comparison over multiple metrics among competitive methods. The strength of our model lies in the fact that with a high compression ratio, we could save more feature exemplars than image exemplars. Especially, when the first task is relatively small (the 10-task scenario in ImageNet-Subset and 5-task in CIFAR10) we outperformed REMIND with a large margin. As future work, we are interested in extending this framework to other continual learning problems.

- **Chapter 5: Bookworm continual learning.** Intelligent systems should be designed and evaluated in controllable but realistic scenarios. Following that, in this chapter, we proposed bookworm continual learning as a novel setting to evaluate visual recognition where continual learning is augmented with an explicit semantic model. In this setting, we unify and generalize zero-shot learning and continual learning. We go beyond zero-shot learning, which cannot adapt to new knowledge, and beyond continual learning, which is limited by not using semantics; in contrast, bookworm continual learning provides a more general setting, closer to human-like continual knowledge acquisition.

- **Chapter 6: Continual learning in cross-model retrieval.** In this chapter we proposed the first study on how forgetting affects multimodal embedding spaces, focusing on cross-modal retrieval. We proposed a continual cross-modal retrieval model that emphasizes the important role of the indexing stage. Cross-modal drifts are also key factors in forgetting in cross-modal tasks. We evaluated several specific tools to alleviate forgetting.

7.2 Future work

For future work we are interested in exploring continual learning problems in more computer vision tasks. In recent years, most research on continual learning is focusing on image classifications. However, image recognition is only one of the many computer vision problems. There are still quite lots of applications not explored in continual learning scenarios. For example, continual learning in semi-supervised [160] or unsupervised [56, 74] manner, continual learning in multi-label classification [96], continual domain adaptation [164] and so on. Besides that, we also want to achieve a system which could be more generalizable to various tasks.

The current methods are mostly developed for a specific problem and hard to be applied to other tasks. Thus, a unified framework would be a desirable outcome.

Publications

1. **Wang, K.**, Liu, X., Bagdanov, A., Herranz, L., Rui, S., van de Weijer, J. (2021). Incremental Meta-Learning via Episodic Replay Distillation for Few-Shot Image Recognition. (3rd CLVISION workshop at CVPR 2022)

2. **Wang, K.**, Liu, X., Herranz, L., van de Weijer, J. (2021). HCV: Hierarchy-Consistency Verification for Incremental Implicitly-Refined Classification. (BMVC 2021)

3. **Wang, K.**, Herranz, L., van de Weijer, J. (2021). ACAE-REMIND for Online Continual Learning with Compressed Feature Replay. (Pattern Recognition Letters)

4. **Wang, K.**, Herranz, L., Dutta, A., van de Weijer, J. (2020). Bookworm continual learning: beyond zero-shot learning and continual learning. (TASK-CV workshop)

5. **Wang, K.**, Herranz, L., van de Weijer, J. (2020). Continual learning in cross-modal retrieval. (2nd CLVISION workshop at CVPR 2021)

6. Yang, S.,**Wang, K.**, Herranz, L., van de Weijer, J. (2020). Simple and effective localized attribute representations for zero-shot learning. (IEEE Signal Processing Letters).

7. Yu, L., Twardowski, B., Liu, X., Herranz, L., **Wang, K.**, Cheng, Y., Jui, S., Weijer, J. V. D. (2020). Semantic drift compensation for class-incremental learning. (CVPR 2020)

8. Caglayan, O., Bardet, A., Bougares, F., Barrault, L., **Wang, K.**, Masana, M., Luis Herranz, van de Weijer, J. (2018). LIUM-CVC submissions for WMT18 multimodal translation task. (WMT 2018 in EMNLP)

Bibliography

[1] Mohamed Abdelsalam, Mojtaba Faramarzi, Shagun Sodhani, and Sarath Chandar. Iirc: Incremental implicitly-refined classification. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[2] Idan Achituve, Aviv Navon, Yochai Yemini, Gal Chechik, and Ethan Fetaya. Gp-tree: A gaussian process classifier for few-shot incremental learning. *International Conference on Machine Learning (ICML)*, 2021.

[3] Zeynep Akata, Florent Perronnin, Zaid Harchaoui, and Cordelia Schmid. Label-embedding for image classification. *Transactions of Pattern Recognition and Machine Analyses (PAMI)*, 38(7):1425–1438, 2015.

[4] 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)*, pages 139–154, 2018.

[5] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *European Conference on Computer Vision (ECCV)*, 2018.

[6] 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*, pages 11849–11860, 2019.

[7] 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*, pages 3366–3375, 2017.

[8] Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning. In *Advances in Neural Information Processing Systems*, pages 11816–11825, 2019.

[9] Han Altae-Tran, Bharath Ramsundar, Aneesh S Pappu, and Vijay Pande. Low data drug discovery with one-shot learning. *ACS central science*, 3(4):283–293, 2017.

[10] Galen Andrew, Raman Arora, Jeff Bilmes, and Karen Livescu. Deep canonical correlation analysis. In *International conference on machine learning*, pages 1247–1255, 2013.

[11] Bernard Ans and Stéphane 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.

[12] Antreas Antoniou, Harrison Edwards, and Amos Storkey. How to train your maml. *International Conference on Learning representations (ICLR)*, 2019.

[13] Sungyong Baik, Junghoon Oh, Seokil Hong, and Kyoung Mu Lee. Learning to forget for meta-learning via task-and-layer-wise attenuation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[14] Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency. Multimodal machine learning: A survey and taxonomy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(2):423–443, 2019.

[15] Jihwan Bang, Heesu Kim, YoungJoon Yoo, Jung-Woo Ha, and Jonghyun Choi. Rainbow memory: Continual learning with a memory of diverse samples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8218–8227, 2021.

[16] Peyman Bateni, Raghav Goyal, Vaden Masrani, Frank Wood, and Leonid Sigal. Improved few-shot visual classification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14493–14502, 2020.

[17] Eden Belouadah, Adrian Popescu, and Ioannis Kanellos. A comprehensive study of class incremental learning algorithms for visual tasks. *Neural Networks*, 135:38–54, 2021.

[18] David Berga, Marc Masana, and Joost Van de Weijer. Disentanglement of color and shape representations for continual learning. *arXiv preprint arXiv:2007.06356*, 2020.

[19] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a" siamese" time delay neural network. In *Advances in neural information processing systems*, pages 737–744, 1994.

[20] Michael M Bronstein, Alexander M Bronstein, Fabrice Michel, and Nikos Paragios. Data fusion through cross-modality metric learning using similarity-sensitive hashing. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 3594–3601. IEEE, 2010.

[21] Massimo Caccia, Pau Rodriguez, Oleksiy Ostapenko, Fabrice Normandin, Min Lin, Lucas Caccia, Issam Laradji, Irina Rish, Alexande Lacoste, David Vazquez, et al. Online fast adaptation and knowledge accumulation: a new approach to continual learning. In *Annual Conference on Neural Information Processing Systems (NIPS)*, 2020.

[22] Yue Cao, Mingsheng Long, Jianmin Wang, Qiang Yang, and Philip S Yu. Deep visual-semantic hashing for cross-modal retrieval. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1445–1454. ACM, 2016.

[23] S. Changpinyo, W. Chao, B. Gong, and F. Sha. Synthesized classifiers for zero-shot learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5327–5336, 2016.

[24] Wei-Lun Chao, Soravit Changpinyo, Boqing Gong, and Fei Sha. An empirical study and analysis of generalized zero-shot learning for object recognition in the wild. In *European Conference on Computer Vision (ECCV)*, pages 52–68. Springer, 2016.

[25] Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *European Conference on Computer Vision (ECCV)*, pages 532–547, 2018.

[26] Arslan Chaudhry, Albert Gordo, Puneet K Dokania, Philip Torr, and David Lopez-Paz. Using hindsight to anchor past knowledge in continual learning. *arXiv preprint arXiv:2002.08165*, 2020.

[27] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-gem. *International Conference on Learning representations (ICLR)*, 2019.

[28] Hui Chen, Guiguang Ding, Xudong Liu, Zijia Lin, Ji Liu, and Jungong Han. Imram: Iterative matching with recurrent attention memory for cross-modal image-text retrieval. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[29] Long Chen, Xin Yan, Jun Xiao, Hanwang Zhang, Shiliang Pu, and Yueting Zhuang. Counterfactual samples synthesizing for robust visual question answering. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[30] Tianshui Chen, Liang Lin, Xiaolu Hui, Riquan Chen, and Hefeng Wu. Knowledge-guided multi-label few-shot learning for general image recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[31] Sanghyuk Chun, Seong Joon Oh, Rafael Sampaio de Rezende, Yannis Kalantidis, and Diane Larlus. Probabilistic embeddings for cross-modal retrieval. *arXiv preprint arXiv:2101.05068*, 2021.

[32] Marcella Cornia, Matteo Stefanini, Lorenzo Baraldi, and Rita Cucchiara. Meshed-memory transformer for image captioning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[33] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *Transactions of Pattern Recognition and Machine Analyses (PAMI)*, 2021.

[34] 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. *Transactions of Pattern Recognition and Machine Analyses (PAMI)*, 2021.

[35] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[36] Jiankang Deng, Jia Guo, Jing Yang, Alexandros Lattas, and Stefanos Zafeiriou. Variational prototype learning for deep face recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11906–11915, June 2021.

[37] Arthur Douillard, Matthieu Cord, Charles Ollion, Thomas Robert, and Eduardo Valle. Podnet: Pooled outputs distillation for small-tasks incremental learning. In *European Conference on Computer Vision (ECCV)*, pages 86–102. Springer, 2020.

[38] Mohamed Elhoseiny, Scott Cohen, Walter Chang, Brian Price, and Ahmed Elgammal. Sherlock: Scalable fact learning in images. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[39] Sebastian Farquhar and Yarin Gal. Towards robust evaluations of continual learning. *arXiv preprint arXiv:1805.09733*, 2018.

[40] Fangxiang Feng, Xiaojie Wang, and Ruifan Li. Cross-modal retrieval with correspondence autoencoder. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 7–16. ACM, 2014.

[41] Zhengyang Feng, Shaohua Guo, Xin Tan, Ke Xu, Min Wang, and Lizhuang Ma. Rethinking efficient lane detection via curve modeling. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[42] Enrico Fini, Stéphane Lathuilière, Enver Sangineto, Moin Nabi, and Elisa Ricci. Online continual learning under extreme memory constraints. In *European Conference on Computer Vision (ECCV)*, pages 720–735. Springer, 2020.

[43] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning (ICML)*, pages 1126–1135. PMLR, 2017.

[44] Alex Freitas and André Carvalho. A tutorial on hierarchical classification with applications in bioinformatics. *Research and trends in data mining technologies and applications*, pages 175–208, 2007.

[45] Andrea Frome, Greg S Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Tomas Mikolov, et al. Devise: A deep visual-semantic embedding model. In *Advances in neural information processing systems*, pages 2121–2129, 2013.

[46] Spyros Gidaris and Nikos Komodakis. Dynamic few-shot visual learning without forgetting. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4367–4375, 2018.

[47] Yunchao Gong, Liwei Wang, Micah Hodosh, Julia Hockenmaier, and Svetlana Lazebnik. Improving image-sentence embeddings using large weakly annotated photo collections. In *European conference on computer vision*, pages 529–545. Springer, 2014.

[48] Albert Gordo, Jon Almazán, Jerome Revaud, and Diane Larlus. Deep image retrieval: Learning global representations for image search. In *European conference on computer vision*, pages 241–257. Springer, 2016.

[49] Jiechao Guan, Zhiwu Lu, Tao Xiang, Aoxue Li, An Zhao, and Ji-Rong Wen. Zero and few shot learning with semantic feature synthesis and competitive learning. *IEEE transactions on pattern analysis and machine intelligence*, 43(7):2510–2523, 2020.

[50] Longteng Guo, Jing Liu, Xinxin Zhu, Peng Yao, Shichen Lu, and Hanqing Lu. Normalized and geometry-aware self-attention network for image captioning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[51] Gunshi Gupta, Karmesh Yadav, and Liam Paull. La-maml: Look-ahead meta learning for continual learning. In *Annual Conference on Neural Information Processing Systems (NIPS)*, 2020.

[52] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE, 2006.

[53] Zongyan Han, Zhenyong Fu, and Jian Yang. Learning the redundancy-free features for generalized zero-shot object recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[54] Tyler L Hayes, Kushal Kafle, Robik Shrestha, Manoj Acharya, and Christopher Kanan. Remind your neural network to prevent catastrophic forgetting. In *European Conference on Computer Vision (ECCV)*, 2020.

[55] Tyler L Hayes, Kushal Kafle, Robik Shrestha, Manoj Acharya, and Christopher Kanan. Remind your neural network to prevent catastrophic forgetting. In *European Conference on Computer Vision (ECCV)*, pages 466–483. Springer, 2020.

[56] Jiangpeng He and Fengqing Zhu. Unsupervised continual learning via pseudo labels. *AAAI International Joint Conference on Artificial Intelligence (IJCAI)*, 2021.

[57] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

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

[59] Elad Hoffer and Nir Ailon. Deep metric learning using triplet network. In *International Workshop on Similarity-Based Pattern Recognition*, pages 84–92. Springer, 2015.

[60] Nazgol Hor and Shervan Fekri-Ershad. Image retrieval approach based on local texture information derived from predefined patterns and spatial domain information. *arXiv preprint arXiv:1912.12978*, 2019.

[61] Timothy Hospedales, Antreas Antoniou, Paul Micaelli, and Amos Storkey. Meta-learning in neural networks: A survey. *Transactions of Pattern Recognition and Machine Analyses (PAMI)*, 2021.

[62] HAROLD HOTELLING. Relations between two sets of variates. *Biometrika*, 28(3-4):321–377, 12 1936.

[63] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 831–839, 2019.

[64] Ahmet Iscen, Jeffrey Zhang, Svetlana Lazebnik, and Cordelia Schmid. Memory-efficient incremental learning through feature adaptation. In *European Conference on Computer Vision (ECCV)*, pages 699–715. Springer, 2020.

[65] Herve Jegou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE transactions on pattern analysis and machine intelligence*, 33(1):117–128, 2010.

[66] Wei Ji, Shuang Yu, Junde Wu, Kai Ma, Cheng Bian, Qi Bi, Jingjing Li, Hanruo Liu, Li Cheng, and Yefeng Zheng. Learning calibrated medical image segmentation via multi-rater agreement modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12341–12351, June 2021.

[67] Huaizu Jiang, Ishan Misra, Marcus Rohrbach, Erik Learned-Miller, and Xinlei Chen. In defense of grid features for visual question answering. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[68] Qing-Yuan Jiang and Wu-Jun Li. Deep cross-modal hashing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3232–3240, 2017.

[69] Xisen Jin, Junyi Du, and Xiang Ren. Gradient based memory editing for task-free continual learning. *arXiv preprint arXiv:2006.15294*, 2020.

[70] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus. *arXiv preprint arXiv:1702.08734*, 2017.

[71] Heechul Jung, Jeongwoo Ju, Minju Jung, and Junmo Kim. Less-forgetting learning in deep neural networks. *arXiv preprint arXiv:1607.00122*, 2016.

[72] Nobuhiro Kaji and Hayato Kobayashi. Incremental skip-gram model with negative sampling. *arXiv preprint arXiv:1704.03956*, 2017.

[73] Rohit Keshari, Richa Singh, and Mayank Vatsa. Generalized zero-shot learning via over-complete distribution. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[74] Shivam Khare, Kun Cao, and James Rehg. Unsupervised class-incremental learning through confusion. *arXiv preprint arXiv:2104.04450*, 2021.

[75] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning representations (ICLR)*, 2015.

[76] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[77] Ryan Kiros, Ruslan Salakhutdinov, and Richard S Zemel. Unifying visual-semantic embeddings with multimodal neural language models. *arXiv preprint arXiv:1411.2539*, 2014.

[78] Benjamin Klein, Guy Lev, Gil Sadeh, and Lior Wolf. Fisher vectors derived from hybrid gaussian-laplacian mixture models for image annotation. *arXiv preprint arXiv:1411.7399*, 2014.

[79] Mark A Kramer. Nonlinear principal component analysis using autoassociative neural networks. *AIChE journal*, 37(2):233–243, 1991.

[80] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision*, 123(1):32–73, 2017.

[81] Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009.

[82] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[83] Brian Kulis et al. Metric learning: A survey. *Foundations and Trends® in Machine Learning*, 5(4):287–364, 2013.

[84] Yann Labbe, Justin Carpentier, Mathieu Aubry, and Josef Sivic. Single-view robot pose and joint angle estimation via render & compare. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1654–1663, June 2021.

[85] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[86] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[87] Kwonjoon Lee, Subhransu Maji, Avinash Ravichandran, and Stefano Soatto. Meta-learning with differentiable convex optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10657–10665, 2019.

[88] Sang-Woo Lee, Jin-Hwa Kim, Jaehyun Jun, Jung-Woo Ha, and Byoung-Tak Zhang. Overcoming catastrophic forgetting by incremental moment matching. In *Advances in neural information processing systems*, pages 4652–4662, 2017.

[89] Timothée Lesort, Vincenzo Lomonaco, Andrei Stoian, Davide Maltoni, David Filliat, and Natalia Díaz-Rodríguez. Continual learning for robotics: Definition, framework, learning strategies, opportunities and challenges. *Information fusion*, 58:52–68, 2020.

[90] Kai Li, Yulun Zhang, Kunpeng Li, and Yun Fu. Adversarial feature hallucination networks for few-shot learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13470–13479, 2020.

[91] Wei-Hong Li, Xialei Liu, and Hakan Bilen. Universal representation learning from multiple domains for few-shot classification. *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[92] Wenyu Li, Tianchu Guo, Pengyu Li, Binghui Chen, Biao Wang, Wangmeng Zuo, and Lei Zhang. Virface: Enhancing face recognition via unlabeled shallow data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14729–14738, June 2021.

[93] Yitong Li, Zhe Gan, Yelong Shen, Jingjing Liu, Yu Cheng, Yuexin Wu, Lawrence Carin, David Carlson, and Jianfeng Gao. Storygan: A sequential conditional gan for story visualization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6329–6338, 2019.

[94] Zhizhong Li and Derek Hoiem. Learning without forgetting. *Transactions of Pattern Recognition and Machine Analyses (PAMI)*, 40(12):2935–2947, 2017.

[95] Weixin Liang, Yuhui Zhang, Yongchan Kwon, Serena Yeung, and James Zou. Mind the gap: Understanding the modality gap in multi-modal contrastive representation learning. *arXiv preprint arXiv:2203.02053*, 2022.

[96] Yan-Shuo Liang and Wu-Jun Li. Optimizing class distribution in memory for multi-label continual learning. *arXiv preprint arXiv:2104.04450*, 2021.

[97] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[98] Jinlu Liu, Liang Song, and Yongqiang Qin. Prototype rectification for few-shot learning. *European Conference on Computer Vision (ECCV)*, 2020.

[99] Qing Liu, Orchid Majumder, Alessandro Achille, Avinash Ravichandran, Rahul Bhotika, and Stefano Soatto. Incremental few-shot meta-learning via indirect discriminant alignment. In *European Conference on Computer Vision (ECCV)*, pages 685–701. Springer, 2020.

[100] Quande Liu, Cheng Chen, Jing Qin, Qi Dou, and Pheng-Ann Heng. Feddg: Federated domain generalization on medical image segmentation via episodic learning in continuous frequency space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1013–1023, June 2021.

[101] Shaoteng Liu, Jingjing Chen, Liangming Pan, Chong-Wah Ngo, Tat-Seng Chua, and Yu-Gang Jiang. Hyperbolic visual embedding learning for zero-shot recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[102] Shichen Liu, Mingsheng Long, Jianmin Wang, and Michael I Jordan. Generalized zero-shot learning with deep calibration network. In *Advances in Neural Information Processing Systems*, pages 2005–2015, 2018.

[103] X. Liu, M. Masana, L. Herranz, J. Van de Weijer, A. M. López, and A. D. Bagdanov. Rotate your networks: Better weight consolidation and less catastrophic forgetting. In *International Conference on Pattern Recognition (ICPR)*, pages 2262–2268, 2018.

[104] Xialei Liu, Marc Masana, Luis Herranz, Joost Van de Weijer, Antonio M Lopez, and Andrew D Bagdanov. Rotate your networks: Better weight consolidation and less catastrophic forgetting. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 2262–2268. IEEE, 2018.

[105] Xialei Liu, Joost van de Weijer, and Andrew D Bagdanov. Rankiqa: Learning from rankings for no-reference image quality assessment. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1040–1049, 2017.

[106] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2020.

[107] Vincenzo Lomonaco, Lorenzo Pellegrini, Pau Rodriguez, Massimo Caccia, Qi She, Yu Chen, Quentin Jodelet, Ruiping Wang, Zheda Mai, David Vazquez, et al. Cvpr 2020 continual learning in computer vision competition: Approaches, results, current challenges and future directions. *Artificial Intelligence*, 303:103635, 2022.

[108] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. *Annual Conference on Neural Information Processing Systems (NIPS)*, 2017.

[109] Zhi Lu, Yang Hu, Yan Chen, and Bing Zeng. Personalized outfit recommendation with learnable anchors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12722–12731, June 2021.

[110] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[111] Arun Mallya, Dillon Davis, and Svetlana Lazebnik. Piggyback: Adapting a single network to multiple tasks by learning to mask weights. In *European Conference on Computer Vision (ECCV)*, pages 67–82, 2018.

[112] Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7765–7773, 2018.

[113] Marc Masana, Xialei Liu, Bartlomiej Twardowski, Mikel Menta, Andrew D Bagdanov, and Joost van de Weijer. Class-incremental learning: survey and performance evaluation. *arXiv preprint arXiv:2010.15277*, 2020.

[114] Marc Masana, Tinne Tuytelaars, and Joost van de Weijer. Ternary feature masks: continual learning without any forgetting. *arXiv preprint:2001.08714*, 2020.

[115] Chandler May, Kevin Duh, Benjamin Van Durme, and Ashwin Lall. Streaming word embeddings with the space-saving algorithm. *arXiv preprint arXiv:1704.07463*, 2017.

[116] Pratik Mazumder, Pravendra Singh, and Piyush Rai. Few-shot lifelong learning. *AAAI Conference on Artificial Intelligence (AAAI)*, 2021.

[117] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24. Elsevier, 1989.

[118] Qiang Meng, Shichao Zhao, Zhida Huang, and Feng Zhou. Magface: A universal representation for face recognition and quality assessment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14225–14234, June 2021.

[119] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[120] Weiqing Min, Shuqiang Jiang, Jitao Sang, Huayang Wang, Xinda Liu, and Luis Herranz. Being a supercook: Joint food attributes and multimodal content modeling for recipe retrieval and exploration. *IEEE Transactions on Multimedia*, 19(5):1100–1113, 2017.

[121] Seyed Iman Mirzadeh, Mehrdad Farajtabar, Razvan Pascanu, and Hassan Ghasemzadeh. Understanding the role of training regimes in continual learning. *arXiv preprint arXiv:2006.06958*, 2020.

[122] Ashish Mishra, Shiva Krishna Reddy, Anurag Mittal, and Hema A. Murthy. A generative model for zero shot learning using conditional variational autoencoders. In *The IEEE Conference on Computer Vision and Pattern Recognition Workshops*, June 2018.

[123] Yair Movshovitz-Attias, Alexander Toshev, Thomas K Leung, Sergey Ioffe, and Saurabh Singh. No fuss distance metric learning using proxies. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 360–368, 2017.

[124] Anusha Nagabandi, Chelsea Finn, and Sergey Levine. Deep online learning via meta-learning: Continual adaptation for model-based rl. *arXiv preprint arXiv:1812.07671*, 2018.

[125] Jiquan Ngiam, Aditya Khosla, Mingyu Kim, Juhan Nam, Honglak Lee, and Andrew Y Ng. Multimodal deep learning. In *ICML*, 2011.

[126] Cuong V Nguyen, Yingzhen Li, Thang D Bui, and Richard E Turner. Variational continual learning. *arXiv preprint arXiv:1710.10628*, 2017.

[127] Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

[128] Yulei Niu, Kaihua Tang, Hanwang Zhang, Zhiwu Lu, Xian-Sheng Hua, and Ji-Rong Wen. Counterfactual vqa: A cause-effect look at language bias. *arXiv preprint arXiv:2006.04315*, 2020.

[129] Mohammad Norouzi, Tomas Mikolov, Samy Bengio, Yoram Singer, Jonathon Shlens, Andrea Frome, Greg Corrado, and Jeffrey Dean. Zero-shot learning by convex combination of semantic embeddings. In *International Conference on Learning representations (ICLR)*, 2014.

[130] Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4004–4012, 2016.

[131] Boris N Oreshkin, Pau Rodriguez, and Alexandre Lacoste. Tadam: Task dependent adaptive metric for improved few-shot learning. *Annual Conference on Neural Information Processing Systems (NIPS)*, 2018.

[132] Yingwei Pan, Ting Yao, Yehao Li, and Tao Mei. X-linear attention networks for image captioning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[133] German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, 2019.

[134] Genevieve Patterson and James Hays. Sun attribute database: Discovering, annotating, and recognizing scene attributes. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[135] Rodolfo M Pereira, Diego Bertolini, Lucas O Teixeira, Carlos N Silla Jr, and Yandre MG Costa. Covid-19 identification in chest x-ray images on flat and hierarchical classification scenarios. *Computer Methods and Programs in Biomedicine*, 194:105532, 2020.

[136] Ameya Prabhu, Philip HS Torr, and Puneet K Dokania. Gdumb: A simple approach that questions our progress in continual learning. In *European Conference on Computer Vision (ECCV)*, pages 524–540. Springer, 2020.

[137] Nan Pu, Wei Chen, Yu Liu, Erwin M Bakker, and Michael S Lew. Lifelong person re-identification via adaptive knowledge accumulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7901–7910, 2021.

[138] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

[139] Jathushan Rajasegaran, Munawar Hayat, Salman H Khan, Fahad Shahbaz Khan, and Ling Shao. Random path selection for continual learning. In *Advances in Neural Information Processing Systems*, pages 12669–12679, 2019.

[140] Jathushan Rajasegaran, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Mubarak Shah. itaml: An incremental task-agnostic meta-learning approach. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13588–13597, 2020.

[141] Amal Rannen, Rahaf Aljundi, Matthew B Blaschko, and Tinne Tuytelaars. Encoder based lifelong learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1320–1328, 2017.

[142] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2001–2010, 2017.

[143] Mengye Ren, Renjie Liao, Ethan Fetaya, and Richard S Zemel. Incremental few-shot learning with attention attractor networks. *Annual Conference on Neural Information Processing Systems (NIPS)*, 2019.

[144] Mengye Ren, Eleni Triantafillou, Sachin Ravi, Jake Snell, Kevin Swersky, Joshua B Tenenbaum, Hugo Larochelle, and Richard S Zemel. Meta-learning for semi-supervised few-shot classification. *International Conference on Learning representations (ICLR)*, 2018.

[145] Anthony Robins. Catastrophic forgetting, rehearsal and pseudorehearsal. *Connection Science*, 7(2):123–146, 1995.

[146] David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. In *Advances in Neural Information Processing Systems*, pages 350–360, 2019.

[147] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[148] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[149] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

[150] Amaia Salvador, Nicholas Hynes, Yusuf Aytar, Javier Marin, Ferda Ofli, Ingmar Weber, and Antonio Torralba. Learning cross-modal embeddings for cooking recipes and food images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3020–3028, 2017.

[151] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, pages 815–823, 2015.

[152] Jonathan Schwarz, Jelena Luketina, Wojciech M Czarnecki, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Progress & compress: A scalable framework for continual learning. In *International Conference on Machine Learning (ICML)*, 2018.

[153] Ari Seff, Alex Beatson, Daniel Suo, and Han Liu. Continual learning in generative adversarial nets. *arXiv preprint arXiv:1705.08395*, 2017.

[154] Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. *arXiv preprint arXiv:1801.01423*, 2018.

[155] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. In *Annual Conference on Neural Information Processing Systems (NIPS)*, 2017.

[156] Carlos N Silla and Alex A Freitas. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery*, 22(1):31–72, 2011.

[157] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.

[158] Edgar Simo-Serra, Eduard Trulls, Luis Ferraz, Iasonas Kokkinos, Pascal Fua, and Francesc Moreno-Noguer. Discriminative learning of deep convolutional feature point descriptors. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 118–126, 2015.

[159] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[160] James Smith, Jonathan Balloch, Yen-Chang Hsu, and Zsolt Kira. Memory-efficient semi-supervised continual learning: The world is its own replay buffer. *arXiv preprint arXiv:2101.09536*, 2021.

[161] Jake Snell, Kevin Swersky, and Richard S Zemel. Prototypical networks for few-shot learning. *Annual Conference on Neural Information Processing Systems (NIPS)*, 2017.

[162] Richard Socher, Andrej Karpathy, Quoc V Le, Christopher D Manning, and Andrew Y Ng. Grounded compositional semantics for finding and describing images with sentences. *Transactions of the Association for Computational Linguistics*, 2:207–218, 2014.

[163] Jong-Chyi Su, Subhransu Maji, and Bharath Hariharan. When does self-supervision improve few-shot learning? In *European Conference on Computer Vision (ECCV)*, pages 645–666. Springer, 2020.

[164] Peng Su, Shixiang Tang, Peng Gao, Di Qiu, Ni Zhao, and Xiaogang Wang. Gradient regularized contrastive learning for continual domain adaptation. *arXiv preprint arXiv:2007.12942*, 2020.

[165] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. Learning to compare: Relation network for few-shot learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1199–1208, 2018.

[166] Xiaoyu Tao, Xiaopeng Hong, Xinyuan Chang, Songlin Dong, Xing Wei, and Yihong Gong. Few-shot class-incremental learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12183–12192, 2020.

[167] Sebastian Thrun. A lifelong learning perspective for mobile robot control. In *Intelligent robots and systems*, pages 201–214. Elsevier, 1995.

[168] Matthew Turk. Multimodal interaction: A review. *Pattern recognition letters*, 36:189–195, 2014.

[169] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. *Annual Conference on Neural Information Processing Systems (NIPS)*, 2016.

[170] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.

[171] Kai Wang, Luis Herranz, Anjan Dutta, and Joost van de Weijer. Bookworm continual learning: beyond zero-shot learning and continual learning. In *TASK-CV workshop in ECCV 2020*, 2020.

[172] Kai Wang, Luis Herranz, and Joost van de Weijer. Continual learning in cross-modal retrieval. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3628–3638, 2021.

[173] Kai Wang, Xialei Liu, Andy Bagdanov, Luis Herranz, Shangling Rui, and Joost van de Weijer. Incremental meta-learning via episodic replay distillation for few-shot image recognition. *3rd CLVISION workshop in CVPR 2022*, 2021.

[174] Kai Wang, Xialei Liu, Luis Herranz, and Joost van de Weijer. Hcv: Hierarchy-consistency verification for incremental implicitly-refined classification. In *BMVA British Machine Vision Conference (BMVC)*, 2021.

[175] Kai Wang, Joost van de Weijer, and Luis Herranz. Acae-remind for online continual learning with compressed feature replay. *Pattern Recognition Letters*, 150:122–129, 2021.

[176] Liwei Wang, Yin Li, Jing Huang, and Svetlana Lazebnik. Learning two-branch neural networks for image-text matching tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(2):394–407, 2019.

[177] Weiran Wang, Raman Arora, Karen Livescu, and Jeff Bilmes. On deep multi-view representation learning. In *International Conference on Machine Learning*, pages 1083–1092, 2015.

[178] Xinyu Wang, Yuliang Liu, Chunhua Shen, Chun Chet Ng, Canjie Luo, Lianwen Jin, Chee Seng Chan, Anton van den Hengel, and Liangwei Wang. On the general value of evidence, and bilingual scene-text visual question answering. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[179] Yaqing Wang, Quanming Yao, James T Kwok, and Lionel M Ni. Generalizing from a few examples: A survey on few-shot learning. *ACM Computing Surveys (CSUR)*, 53(3):1–34, 2020.

[180] Jiwei Wei, Xing Xu, Yang Yang, Yanli Ji, Zheng Wang, and Heng Tao Shen. Universal weighting metric learning for cross-modal matching. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[181] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In *European conference on computer vision*, pages 499–515. Springer, 2016.

[182] Jason Weston, Samy Bengio, and Nicolas Usunier. Wsabie: Scaling up to large vocabulary image annotation. In *Twenty-Second International Joint Conference on Artificial Intelligence*, 2011.

[183] Chenshen Wu, Luis Herranz, Xialei Liu, Joost van de Weijer, Bogdan Raducanu, et al. Memory replay gans: Learning to generate new categories without forgetting. *Annual Conference on Neural Information Processing Systems (NIPS)*, 31:5962–5972, 2018.

[184] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 374–382, 2019.

[185] Peter R Wurman, Samuel Barrett, Kenta Kawamoto, James MacGlashan, Kaushik Subramanian, Thomas J Walsh, Roberto Capobianco, Alisa Devlic, Franziska Eckert, Florian Fuchs, et al. Outracing champion gran turismo drivers with deep reinforcement learning. *Nature*, 602(7896):223–228, 2022.

[186] Yongqin Xian, Zeynep Akata, Gaurav Sharma, Quynh Nguyen, Matthias Hein, and Bernt Schiele. Latent embeddings for zero-shot classification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 69–77, 2016.

[187] Yongqin Xian, Christoph H Lampert, Bernt Schiele, and Zeynep Akata. Zero-shot learning—a comprehensive evaluation of the good, the bad and the ugly. *Transactions of Pattern Recognition and Machine Analyses (PAMI)*, 41(9):2251–2265, 2018.

[188] Yongqin Xian, Tobias Lorenz, Bernt Schiele, and Zeynep Akata. Feature generating networks for zero-shot learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[189] Yongqin Xian, Saurabh Sharma, Bernt Schiele, and Zeynep Akata. f-vaegan-d2: A feature generating framework for any-shot learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[190] Xian et al. Zero-shot learning—a comprehensive evaluation of the good, the bad and the ugly. *PAMI*, 2018.

[191] Ran Xu, Caiming Xiong, Wei Chen, and Jason J Corso. Jointly modeling deep video and compositional text to bridge vision and language in a unified framework. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[192] Shiqi Yang, Kai Wang, Luis Herranz, and Joost van de Weijer. On implicit attribute localization for generalized zero-shot learning. *IEEE Signal Processing Letters*, 2021.

[193] Shuo Yang, Lu Liu, and Min Xu. Free lunch for few-shot learning: Distribution calibration. *International Conference on Learning representations (ICLR)*, 2021.

[194] Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong learning with dynamically expandable networks. *arXiv preprint arXiv:1708.01547*, 2017.

[195] Sung Whan Yoon, Do-Yeon Kim, Jun Seo, and Jaekyun Moon. Xtarnet: Learning to extract task-adaptive representation for incremental few-shot learning. In *International Conference on Machine Learning (ICML)*, pages 10852–10860. PMLR, 2020.

[196] Lu Yu, Xialei Liu, and Joost van de Weijer. Self-training for class-incremental semantic segmentation. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[197] Lu Yu, Bartlomiej Twardowski, Xialei Liu, Luis Herranz, Kai Wang, Yongmei Cheng, Shangling Jui, and Joost van de Weijer. Semantic drift compensation for class-incremental learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6982–6991, 2020.

[198] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. *Proceedings of machine learning research*, 70:3987, 2017.

[199] Chen Zeno, Itay Golan, Elad Hoffer, and Daniel Soudry. Task agnostic continual learning using online variational bayes. *arXiv preprint arXiv:1803.10123*, 2018.

[200] Chi Zhang, Nan Song, Guosheng Lin, Yun Zheng, Pan Pan, and Yinghui Xu. Few-shot incremental learning with continually evolved classifiers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12455–12464, 2021.

[201] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5907–5915, 2017.

[202] Junting Zhang, Jie Zhang, Shalini Ghosh, Dawei Li, Serafettin Tasci, Larry Heck, Heming Zhang, and C-C Jay Kuo. Class-incremental learning via deep model consolidation. In *Proceedings of the IEEE Workshop on Applications of Computer Vision*, pages 1131–1140, 2020.

[203] Li Zhang, Tao Xiang, and Shaogang Gong. Learning a deep embedding model for zero-shot learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2021–2030, 2017.

[204] Min-Ling Zhang and Zhi-Hua Zhou. A review on multi-label learning algorithms. *IEEE transactions on knowledge and data engineering*, 26(8):1819–1837, 2013.

[205] Ziming Zhang and Venkatesh Saligrama. Zero-shot learning via semantic similarity embedding. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4166–4174, 2015.

[206] Yizhe Zhu, Mohamed Elhoseiny, Bingchen Liu, Xi Peng, and Ahmed Elgammal. A generative adversarial approach for zero-shot learning from noisy texts. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1004–1013, 2018.

[207] Yizhe Zhu, Jianwen Xie, Zhiqiang Tang, Xi Peng, and Ahmed Elgammal. Semantic-guided multi-attention localization for zero-shot learning. In *Advances in Neural Information Processing Systems*, pages 14917–14927, 2019.