UAB
Universitat Autònoma de Barcelona

Going beyond Classification Problems for the Continual Learning of Deep Neural Networks

ChenshenWu

Universitat Autònoma
de Barcelona

Going beyond Classification Problems for the Continual Learning of Deep Neural Networks

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

Bellaterra, December 16, 2022

Director

Dr. Joost van de Weijer
Centre de Visió per Computador
Universitat Autònoma de Barcelona

Dr. Bogdan Raducanu
Centre de Visió per Computador
Universitat Autònoma de Barcelona

Thesis
committee

Dr. Roberto Javier López Sastre
Departamento de Teoría de la Señal y Comunicaciones
Universidad de Alcalá

Dr. Bartlomiej Twardowski
Centre de Visió per Computador

Dr. Antonio Carta
Department of Computer Science
University of University of Pisa

This document was typeset by the author using LaTeX 2$_{\varepsilon}$.

The research described in this book was carried out at the Centre de Visió per Computador, Universitat Autònoma de Barcelona.

ISBN: 978-84-126409-0-8

Printed by Ediciones Gráficas Rey, S.L.

Acknowledgements

Thanks to my supervisors Joost van de Weijer and Bogdan Raducanu, whose academic guidance helped me to learn to think and understand things differently; their constant confidence in me, which supported me to finish my PhD; and the good equipment and environment they provided, so I could focus on my work.

Thanks to all the partners I have worked with. Thanks to Kai Wang for his generous help at the most difficult time of my research progress. Thanks to Xialei Liu, whose strong belief in his work inspires me and is a great motivation for me to keep working. Thanks to Fei Yang, who often discusses problems with me and keeps me interested in my research. Thanks to Luis Herranz, who provided me with very important inspiration for my first work. Thanks to Yaxing Wang, whose optimism about his work and life has been infectious to me. Thanks to Marc Masana, who is well versed in continue learning and was able to get good answers to every question he asked. Thanks to Héctor Laria for providing me with a proxy so that I can work efficiently from home. Thanks to Juan Antonio for helping me with Catalan and Spanish.

Thanks to all the friends I met in my study and daily life. Thanks to Lei Kang, Lu Yu, Yi Xiao, Shiqi Yang, Yiyong Yang, Danna Xue, Qingshan Chen, Lichao Zhang, Zhijie Fang, Shenyu Zhou, Songlin Wang, Guang Shi, Yifan Wu, it was a lot of fun to hang out with them and make me feel I am not alone anymore. Also, thanks to Dechuan Kong and Xiulian Meng, the in-depth conversation with them made me understand myself better.

Thanks to the China Scholarship Council for providing me with a scholarship to start my PhD and for providing me with important financial support for my studies and life during this period. Thanks to the staff of Computer Vision Center, Joan, Montse, Gigi, Kevin, and others for their help.

Last but not least, I would like to thank my family members. Thanks to my parents for their constant and unconditional support. Thanks to my wife and soul mate, Yi Zhang, for supporting each other through the happy and hard times in Barcelona.

Thanks to my hamster Sanwan, who provided me with warm companionship, as well as generous appearing in the pictures in this thesis.

Abstract

Deep learning has made tremendous progress in the last decade due to the explosion of training data and computational power. Through end-to-end training on a large dataset, image representations are more discriminative than the previously used hand-crafted features. However, for many real-world applications, training and testing on a single dataset is not realistic, as the test distribution may change over time. Continuous learning takes this situation into account, where the learner must adapt to a sequence of tasks, each with a different distribution. If you would naively continue training the model with a new task, the performance of the model would drop dramatically for the previously learned data. This phenomenon is known as catastrophic forgetting.

Many approaches have been proposed to address this problem, which can be divided into three main categories: regularization-based approaches, rehearsal-based approaches, and parameter isolation-based approaches. However, most of the existing works focus on image classification tasks and many other computer vision tasks have not been well-explored in the continual learning setting. Therefore, in this thesis, we study continual learning for image generation, object re-identification, and object counting.

For the image generation problem, since the model can generate images from the previously learned task, it is free to apply rehearsal without any limitation. We developed two methods based on generative replay. The first one uses the generated image for joint training together with the new data. The second one is based on output pixel-wise alignment. We extensively evaluate these methods on several benchmarks.

Next, we study continual learning for object Re-Identification (ReID). Although most state-of-the-art methods of ReID and continual ReID use softmax-triplet loss, we found that it is better to solve the ReID problem from a meta-learning perspective because continual learning of reID can benefit a lot from the generalization of meta-learning. We also propose a distillation loss and found that the removal of the positive pairs before the distillation loss is critical.

Finally, we study continual learning for the counting problem. We study the mainstream method based on density maps and propose a new approach for density map distillation. We found that fixing the counter head is crucial for the continual learning of object counting. To further improve results, we propose an adaptor to adapt the changing feature extractor for the fixed counter head. Extensive evaluation shows that this results in improved continual learning performance.

Keywords: *continual learning, generative adversarial model, object re-identification, counting*

Resumen

El aprendizaje profundo ha progresado enormemente en la última década debido a la explosión en la medida de los datos de entrenamiento y los avances en poder computacional. A través del entrenamiento de extremo a extremo en un gran conjunto de datos, las representaciones de imágenes son más discriminatorias que las funciones hechas a mano que se usaban anteriormente. Sin embargo, para muchas aplicaciones del mundo real, el entrenamiento y evaluación en un solo conjunto de datos no son realistas, ya que la distribución y características de los datos pueden cambiar con el tiempo. El aprendizaje continuo se enfoca en explorar esta situación, donde el sistema debe adaptarse a una secuencia de tareas, cada una con una distribución diferente. Si ingenuamente continuara entrenando el modelo con una nueva tarea, el rendimiento del modelo se reduciría drásticamente para los datos aprendidos anteriormente. Este fenómeno se conoce como olvido catastrófico.

Se han propuesto muchos enfoques para abordar este problema, que se pueden dividir en tres categorías principales: enfoques basados en la regularización, enfoques basados en ensayos y enfoques basados en el aislamiento de parámetros. Sin embargo, la mayoría de los trabajos existentes se centran en tareas de clasificación de imágenes. Muchas otras tareas de visión artificial no han sido exploradas en el entorno de aprendizaje continuo. Por lo tanto, en esta tesis estudiamos el aprendizaje continuo para la generación de imágenes, la reidentificación de objetos y el recuento de objetos.

Para el problema de generación de imágenes, dado que el modelo puede generar imágenes a partir de la tarea previamente aprendida, es libre de aplicar aprender de esos datos generados sin ninguna limitación. Desarrollamos dos métodos basados en la reproducción generativa. El primero utiliza la imagen generada para el entrenamiento conjunto junto con los nuevos datos. El segundo se basa en la alineación de píxeles de salida. Evaluamos ampliamente estos métodos en varios puntos de referencia.

A continuación, estudiamos el aprendizaje continuo para la reidentificación de objetos (ReID). Aunque la mayoría de los métodos de vanguardia de ReID y ReID continuo usan una función de coste basade en tripletes softmax, descubrimos que es mejor resolver el problema de ReID desde una perspectiva de meta-aprendizaje, porque el aprendizaje continuo de reID puede beneficiarse mucho de su capacidad de generalización. También proponemos una nueva función de coste por destilación y descubrimos que la eliminación de los pares positivos antes de aplicar la función de coste es fundamental.

Finalmente, estudiamos el aprendizaje continuo para el problema de recuento de objetos. Estudiamos el método principal basado en mapas de densidad y proponemos un nuevo enfoque para la destilación de mapas de densidad. Descubrimos que ajustar los parametros de la ultima capa del modlo del contador es crucial para el aprendizaje continuo del recuento de objetos. Para mejorar aún más los resultados, proponemos un adaptador para ajustar el extractor de caracteristicas del modelo manteniendo la ultima capa fija. Mediante una evaluación exhaustiva mostramos resultados de mejora de rendimiento en aprendizaje continuo.

Palabras clave: *aprendizaje continuo, modelo generativo adversarial, reidentificación de objetos, recuento de objetos*

Resum

L'aprenentatge profund ha experimentat un gran progrés en l'última dècada a causa de l'explosió de la mida de les dades d'entrenament i els avenços en potència computacional. Mitjançant el processament d'inici a final de grans conjunts de dades utilitzant xarxes neuronals, les representacions d'imatges són més discriminatòries que les funcions fetes a mà utilitzades anteriorment. Tanmateix, per a moltes aplicacions del món real, la formació i les proves en un únic conjunt de dades no són realistes, ja que la distribució i característiques de les proves pot canviar amb el temps. L'aprenentatge continu té en compte aquesta situació, on l'aprenent s'ha d'adaptar a una seqüència de tasques, cadascuna amb una distribució diferent. Si ingènuament continues entrenant el model amb una tasca nova, el rendiment del model baixaria dràsticament per a les dades apreses anteriorment. Aquest fenomen es coneix com a oblit catastròfic.

S'han proposat molts mètodes per abordar aquest problema, que es poden dividir en tres categories principals: mètodes basats en regularització, mètodes basats en assaig i mètodes basats en aïllament de paràmetres.

Tot i això, la majoria dels treballs existents s'enfoquen a tasques de classificació d'imatges i moltes altres tasques de visió per ordinador no han estat ben explorades en l'entorn d'aprenentatge continu. Per tant, en aquesta tesi, estudiem l'aprenentatge continu per a la generació d'imatges, la reidentificació d'objectes i el recompte d'objectes.

Pel que fa a la qüestió de la generació d'imatges, com que el model pot generar imatges a partir de la tasca prèviament apresa, és lliure d'aplicar l'assaig sense cap limitació. Hem desenvolupat dos mètodes basats en la reproducció generativa. La primera utilitza la imatge generada per a l'entrenament conjunt juntament amb les noves dades. El segon es basa en l'alineació de píxels de sortida. Avaluem àmpliament aquests mètodes en diversos punts de referència.

A continuació, estudiem l'aprenentatge continu per a la reidentificació d'objectes (ReID). Tot i que la majoria dels mètodes d'última generació de ReID i ReID contínua usen tècniques basades en triplets softmax, vam trobar que és millor resoldre el problema de ReID des d'una perspectiva de meta-aprenentatge, ja que l'aprenentatge continu de reID es pot beneficiar molt de la seva capacitat de generalització. També vam proposar una funció de cost per destil·lació i vam trobar que l'eliminació dels parells positius abans de calcular el cost per destil·lació és crítica.

Finalment, estudiem l'aprenentatge continu per al problema del recompte. Estudiem el mètode principal basat en mapes de densitat i proposem un nou enfocament per a la destil·lació de mapes de densitat. Hem trobat que ajustar els paràmetres de les últimes capes del model del comptador és crucial per a l'aprenentatge continu del recompte d'objectes. Per millorar encara més els resultats, proposem un adaptador per adaptar l'extractor de funcions canviants per a les últimes capes del comptador fix. Una avaluació àmplia mostra que això es tradueix en un millor rendiment de l'aprenentatge continu.

Paraules clau: *aprenentatge continu, model generatiu adversari, reidentificació d'objectes, recompte d'objectes*

Contents

List of Figures

List of Tables

1 Introduction

Deep learning has seen tremendous progress in the last decade due to the explosive growth of training data and computational power. Its ability to jointly learn features and a classifier, known as end-to-end training, resulted in image representations which were much more discriminative than the hand-crafted features used before. As a result, it has been applied to most computer vision tasks and achieves remarkable result, e.g., in image classification [46, 133], face recognition [98, 142], and image generation [11, 38].

The most common paradigm nowadays for deep learning is to train a model on a very large dataset, where the distribution is assumed to be identical to the one on which the model is going to be tested. The distribution of the training set does not change throughout the training process. However, for many real-world applications this is not the case, and the training distribution could change over time. Continual learning considers this scenario, where a learner has to adapt to a sequence of tasks, each with a different distribution. If you would naively continue training the model with a new task, the performance of the model would drop drastically for the previously learned data. This phenomenon is called catastrophic forgetting [97].

However, this is not the way we learn. We accumulate and preserve the knowledge learned from previous tasks and use it in learning new tasks and solving new problems. The knowledge learned from the new task also helps us to solve previous problems, even if we have not seen this problem for some time. The knowledge we learn is general and independent of certain tasks. Even if we forget the "data" (how we learned it), we still remember the knowledge. When a new task arrives, we can quickly recall it and adjust to this new task.

To make the deep learning system more capable of learning knowledge from the real world scenario like humans, people studied the *continual learning* problem [66, 75]. In this setting, the model is learning from a changing distribution or several training data. And the performance of the model is measured on a distribution that combines all the trained data. The ability to learn from data with varying distributions is a desired characteristic for many AI systems. In this thesis, we aim to contribute to the research on continual learning and extend its applications to

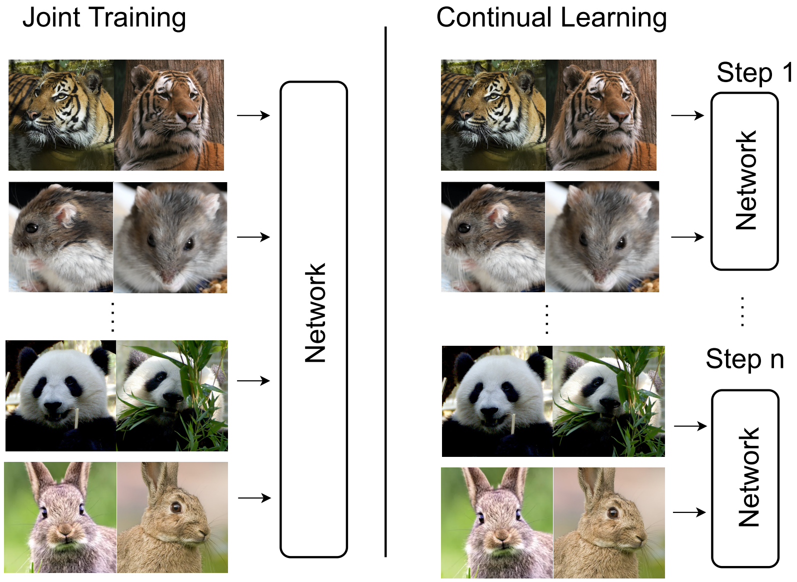


Figure 1.1: The most common paradigm for deep learning nowadays is to train a model with all the data together (Joint Training). A more realistic setting is Continual Learning (CL), where the data comes in sequence and the model learns incrementally.

various domains of computer vision.

1.1 Continual Learning

Currently, most research on continual learning is focussed on the image classification problem [55, 66, 84, 128]. In this case, the setup that is most commonly considered splits the dataset in a number of tasks, where each task defines an image classification tasks over a limited number of classes. The learner has to learn then from the sequence of tasks. When training on a new task, the learner does not have access (or only has limited access) to data from previous tasks. At each moment, however, the learner has to be able to classify all classes previously seen. There are two main scenarios called task-incremental learning (TIL) and class-incremental learning (CIL). The former requires that at inference time the method knows the task-ID of the test sample. The latter does not assume that the task-ID is given. This makes CIL a more challenging setup, since the method needs to be able to discriminate between all classes from all tasks. People have addressed the problem from several aspects and proposed a large amount of techniques. These can be mainly divided in five types of techniques:

Regularization. Regularization based techniques save an old model and apply a regularization loss between the new and old model. Parameter regularization based techniques, like EWC [66], MAS [3] and SI [172], apply regularization on the changing of parameters. They evaluate an importance factor for each parameter, where the parameter regularization loss of each parameter is scaled by it. Different methods evaluate the importance matrix differently. The data regularization based techniques, e.g. LwF [75], iCaRL [119], LUCIR [54], PoDNET [27], apply the regularization loss on the changing of the output of the network. Some [2, 75, 119] apply it only the final prediction probability and some others apply it on the final features [54] or also the intermediate features [27]. Regularization methods have shown good performance and are generally easy to implement, as such they are also often used as one of the techniques in more complex continual learning systems.

Parameter Isolation. Parameter isolation based techniques, e.g, PackNet [92], PiggyBack [91] and HAT [128], use only part of the parameters in the network for each task and these parameters are fixed after training. Most of the methods using parameter isolation requires a task-ID to select the needed parameters, so they are only available for TIL (Task Incremental Learning) setting. But there are also methods in this category that can be applied to CIL (Class Incremental Learning). For example, RPS [116] uses all the available paths during inference. Expandable network [164, 167] is also a special type of parameter isolation where the size of the network is growing during the learning process. Parameter isolation methods are very good at preventing forgetting, however, they typically require the capacity of the network to grow over time, which might not be desirable for all applications.

Gradient Update Modification. Gradient update modification based techniques modify the gradient directly to prevent the performance drop of the previous task. For example, GEM [84] and AGEM [15] apply an inequality constrain on the gradient to prevent the loss for the saved exemplars from increasing. GPM [124], OGD [30] and OWM [171] construct a base for the space of gradient update. The components that might damage the performance of the previous tasks are removed from the base.

Replay. When storing few exemplars for the previous tasks is allowed, the idea of jointly training the new samples together with the old exemplars comes naturally. People mentioned such techniques data back to 1990s [118, 121] and they call it experience replay(ER) or rehearsal. Most methods that allow for exemplars use experience replay as an importance component of their method to recall past tasks, e.g. iCaRL [119], BIC [161]. However, because of the limitation of the usage of exemplars, there are also methods that train a generative model to implement generative replay. DGR [130], DGM [108] and DreamingReplay [135] generate and replay the whole image. On the other hand GFR [82] and PASS [182] generate and replay at feature level.

Model Generalization. Instead of preventing forgetting when learning new tasks, this type of techniques focuses on training a network that tends not to forget. There are three types of approaches in this category. The first type, called meta learning, defines it as a loss and optimize it directly [42, 57, 120]. The second type use self-supervised learning(SSL) for the feature extractor [111, 182]. The features learned by self-supervised learning are less limited to a certain task (especially when compared with supervised learning) and they typically tend to generalize well to new tasks. The third type focuses on the effect of the training regimes [55], e.g. studying the influence of weight decay, learning rate, batch size and optimizer, on continual learning.

Classification Specific. There are other techniques designed to solve specific problems that are highly related to the classification problem. BIC [161] and IL2M [10] solve the bias problem in CIL that the classifier always tend to predict classes from the latest task. iCaRL [119] use Nearest-Mean-of-Exemplars to replace the softmax classification. Finally, SDC [169] uses the triplet loss instead of cross-entropy loss for the training of the classifier.

In this thesis, we identify three main research areas in which continual learning has not yet researched in detail, including generative models, object re-identification and object counting. We will provide a more elaborate overview of continual learning methods in Chapter 2.

1.1.1 Continual learning of generative models

Image generation is an important and challenging task in computer vision. Since the rise of deep learning, to generate a realistic image is no longer an impossible task. People proposed different types of methods for this problem, including VAE [64], GAN [38], Flow-based [65] and diffusion models [139].

In this thesis, we focus on Generative Adversarial Network(GAN) for image generation. A GAN consists of two networks, a generator and a discriminator. The generator takes a noise vector as input and generates images. The discriminator takes a generated or real image as input and outputs a scalar value. The training of the GAN is a min-max game where the discriminator tries to maximize the difference between the generated image and the real one while the generator tries

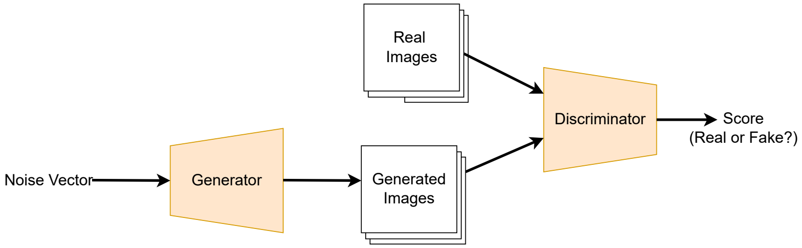


Figure 1.2: Generative Adversarial Network [38]. The generator takes a noise vector as input and generate images. The discriminator takes an image (generated or real) as input and outputs a score indicating if it is real or fake. The discriminator is trained to find the difference between the real and fake while the generator is trained to deceive the discriminator. Training of these models within a continual learning setup has received little research attention.

to minimize this difference. As a result, the generator will end up generating images that are similar to the training dataset.

Since the appearance of the original GAN, many improvements have been made to generate better quality images. The original GAN suffers from stability problems. The training process is sometimes compared with a boxing game, if one network is overwhelming another, the game is over. People proposed alternative losses [6, 93] or regularization methods [41, 101] to solve this problem. Another improvement comes from the architecture of the network, e.g. DCGAN [115], ProgressiveGAN [59] and BigGAN [11]. They design larger and better networks to generate larger and better images.

Another improvement of GAN is to control the generated images. In [115], they propose a conditional GAN where the image generation can be conditioned by a given class label. The class-conditioning is further improved by ACGAN [107] and projectionGAN [102]. The former uses an auxiliary classifier to assist the discriminator on the class label and the latter uses an additional projection layer. In [56], they propose an image conditioned GAN where the class label conditioning is replaced by an image. Zhang et al. [173] propose a text conditioned GAN which provide better convenience for user-oriented applications.

Despite this impressive progress, all these models require a joint training paradigm, which is not always available in the real life. In addition, a promising approach to continual learning of classification networks is pseudo-replay [130] and it also requires the continual learning of a generator. However, they propose a naive approach to continual training of the generator. *Therefore, in this thesis we investigate*

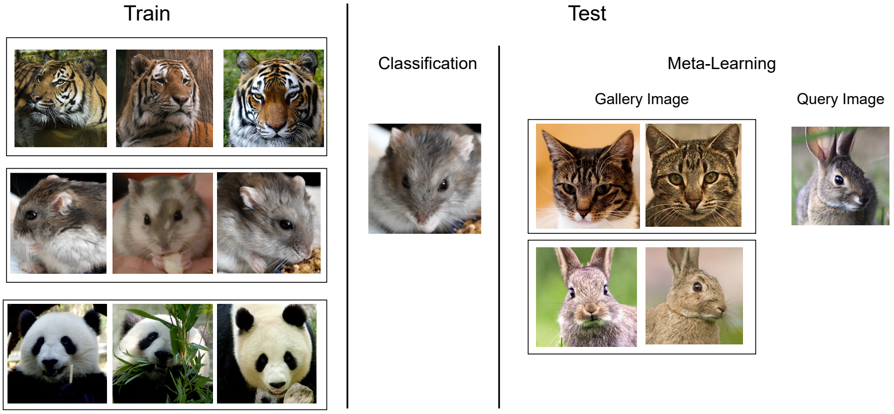


Figure 1.3: Classification vs Metric-learning and Meta-learning. For both classification and meta-learning problem, the model is trained with labeled images. During the inference, for the classification problem, the test sample is from one of the learned categories. However, for the meta-learning problem, few more labeled samples (gallery images) from new categories are given, and the new test sample (query image) is going to be predicted from these new categories.

how to extend continual learning theory to generative models, and more particularly to generative adversarial networks.

1.1.2 Continual learning for object re-identification

The goal of Object re-identification (ReID) is to identify individuals from the gallery image set that are identical to the query image [47, 174]. These images are usually from different cameras and viewpoints. Object ReID has been widely used in applications including person re-identification [17, 80, 166], vehicle re-identification [61, 85, 179], and face verification [148, 149].

A key aspect for object ReID is metric learning [88]. Metric learning has first been proposed by Chopra et al. [23]. Traditionally, the classification task classifies an unseen image to a seen category. A certain amount of samples for each category are needed for training the classifier. However, this paradigm is not suitable for scenarios like face recognition, where the number of categories is large and the number of samples per category is limited. To solve this problem, metric learning is proposed to learn a similarity metric from data. So, during the inference, whether several samples belong to a same category can be predicted by the distance to each other.

In [23], they use a contrasting loss, which bring the pair from the same category closer and push the pairs from different category farther. People extend it to triplet loss with triplet inputs [52] and N-pair loss with batch inputs [132]. More recently, deep metric learning approaches have been used to learn non-linear relationships [62, 103]. Metric learning has been widely applied to object re-identification [47, 174], mainly focusing on person ReID [17, 74, 166, 176], vehicle ReID [61, 85, 179] and face verification [148, 149, 176]).

Another similar research area is few-shot learning [9, 70, 78, 140, 153, 165], where the objective is also to classify unseen categories with only few samples guided. Several works use Meta-learning [8, 21, 32, 53, 106, 136, 144], which focuses on generalizing to unseen tasks, for few-shot learning. Meta learning is mainly metrics-based, e.g. ProtoNets [136] and RelationNets [141] or optimization-based, e.g. MAML [32] and Reptile [106].

However, most of the work mentioned previously is based on a joint training paradigm. There are few works in incremental metric learning, e.g. FGIR [20], and AKA [112]. However, they do not consider meta-learning to ensure improved generalizability to future tasks. Neither do they consider the intra-domain setting, where the similarity between the tasks is much smaller than in the inter-domain setting studied in their papers.

1.1.3 Continual learning of counting

The image-based counting task aims to infer the number of people, vehicles or any other objects present in images. It has a wide range of applications such as traffic control, environment survey and public safety.

Existing work can be divided into two main categories: point-based [77, 125, 138] and density-based [90, 147]. Point-based methods [138] aim to predict the position of each object which provides extra information in addition to the number itself. Some methods [76, 125] are based on object detectors. They benefit from the development of object detection, but inaccuracy is introduced by estimating the bounding box ground truth from the point ground truth. Liu et al. [77] propose to train a model to perform the counting and localization tasks at the same time, and they define an adaptive fusion scheme to make these two tasks complement each other. Song et al. [138] propose the Point to Point Network (P2PNet) that predicts the localization points directly. They develop a one-to-one matching strategy from the prediction to the ground truth based on the Hungarian algorithm.

However, the point-based approach does not perform well for images that are too dense and have a lot of occlusions. Density-based methods [90, 147, 175]

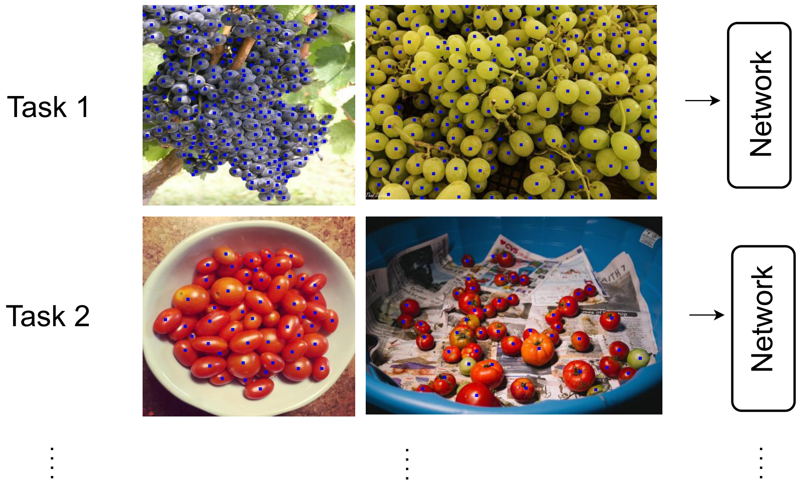


Figure 1.4: Continual learning for object counting. In each task, the model aims to learn a new type of object.

evaluate a density map for the image and the counting value is simply the sum of the density map. Several methods focus on identifying objects in different sizes. In [126, 175], they propose to use multi-column CNNs of different size. In [134], Sindagi et al. propose the Contextual Pyramid CNN that encodes local and global context together. Another line of research focuses on defining a better loss for the distance between the ground truth to the predicted density map. Ma et al. [90] propose a Bayesian loss and Wang et al. [147] propose to solve an Optimal Transport (OT) problem.

Most of the counting method is trained on one specific dataset each time. In [16, 89], the authors address the problem of training a model on multiple datasets at the same time. In [89], Ma et al. address the problem of model sensitivity to scale shift. They propose a closed-form solution of the optimal image rescaling factor given the scale distribution. The scale alignment is applied on the patches that divide from the image. They train a network to predict the spatial distribution and the scale distribution. Chen et al. [16] propose a Domain-specific Knowledge Propagating Network (DKPNet) to address the problem that the model tends to focus on learning in the dominant domain at the expense of the non-dominant domains.

In [35, 44, 152], the authors address the problem of domain adaptation in counting problem. The model is trained on the source dataset and the label of the target dataset is limited. Wang et al. [152] consider generating a synthetic dataset as a source set. And they propose using CycleGAN [183] for domain adaptation to the real dataset. In [35, 44], they use typical adversarial training [33] for the domain adaption. A discriminator is trained to identify the feature to source or target, and an adversarial loss is applied to the feature extractor to minimize the discrepancy of the discriminator.

These methods mentioned above are limited to counting people. In [87], Lu et al. propose class-agnostic counting, where the model can count any type of objects. They propose Generic Matching Network(GMN). The network is pretrained on video data for tracking, and they count instances by matching the instance on the test images with the specified exemplar patch. In [117], they propose to train a model that can count multiple type of objects, e.g., animals and fruits.

1.2 Objectives and approach

1.2.1 Continual learning of generative models

Most of previous work in generative model requires a joint training paradigm. Only few works, for example [127] study the continual learning of a generative model. They adapt EWC [66], which is a popular method for classification, to generative model. However, it does not achieve satisfying results. Therefore, we define the following objective:

> **Continual learning of generative models:** Incorporate the new developments of conditional GANs and regularization methods for the continual learning of generative models. We will focus on generative adversarial networks (GANs).

The idea of the proposed method is to save a copy of the old model to generate samples from previous tasks, and use these samples to prevent the new model from forgetting. We will consider two variations of our method. The first one is called Joint Training (JT) where we combine the generated samples with the new real samples to train a new GAN. The second one is called replay alignment (AL) where we try to align the old and the new model which means that when the input of the two model are identical the output should also be the same. We will apply a pixel-wise square loss given the sample input noise vector and class index.

1.2.2 Continual learning for object re-identification

Most of the methods for the re-identification(re-ID) problem require a large dataset for joint training. However, in reality the dataset come in a sequence and sometimes are not allowed to be stored. In Chapter 4, we address the problem of continual object re-identification problem. Previous state-of-the-art method of continual object re-ID problem AKA [112] use softmax-triplet loss of BoT [88], which follows a typical metric learning perspective. We also consider the object re-identification problem from a perspective of meta learning, which can improve generalization, following DMML [18]. We define the following objective for Chapter 4:

> **Continual learning of object reID:** We propose a novel approach for the continual learning of object ReID. We will consider an exemplar-free setting due to the privacy consideration in person ReID problem.

We observe that meta-metric learning [18] is a better approach when it comes to continual learning, compared with the typical triplet-loss like metric learning. To prevent forgetting, we aim to apply knowledge distillation loss [50] on the output probability. Furthermore, we evaluate the role of positive and negative pairs in the knowledge distillation loss. We will also consider a new incremental intra-domain object ReID benchmark, where the previous benchmark are mainly cross domain.

1.2.3 Continual learning of counting problem

Most of the previous work in counting focus on counting a single type of object. Only few works [87, 117] study the setting of counting different objects. However, these works are in the few-shot learning setting, which relies on training the model on a large dataset in a single step. In Chapter 5, we consider the following objective:

> **Continual learning of counting:** We propose a novel method for the continual learning of counting based on meta-metric learning. In each task, the model is trained to count a new type of object.

We propose to train a task-specific counter head for each object, and all the counter heads share the same feature extractor. After the training of each task, the counter head is fixed, but the feature extractor is still trained with new data. To prevent forgetting, we apply a regularization loss on the output of previous counter heads. In addition to that, we also investigate the usage of an adaptor that translates the feature from the new feature extractor to the old one. So the forgetting by the drifting of the feature extractor can be mitigated.

2 Related Work

The goal of continuous learning [26, 94] is to train a model to learn from a non-stationary distribution. Typically, for most deep learning methods, one assumes that data arrives in a sequence of batches (or datasets) each with a different data distribution. The vast majority of literature considers the stationary distribution case, where the model has access to all data, and it can train on this data by cycling through it (in epochs). This strategy, also known as joint training, is however not practical for many real-world applications. In many real-world applications, the data distribution can change over time. As an example, consider the COVID pandemic, when many people started to wear face-masks, requiring a fast update of all face recognition software. Continual learning develops theory, metrics, and algorithms to allow algorithms to learn from a sequence of data. In this chapter, we provide an overview of the main approaches that have been proposed.

One of the main problems of continual learning is *catastrophic forgetting* [39,66, 94]. When the model is trained with data from new tasks and the data from previous task is no longer available to the model, the performance of the previous tasks drop drastically. Related to this challenge is the stability-plasticity dilemma [99], which refers to the trade-off between retaining knowledge from previous tasks and the acquiring of new knowledge from the current data. Continual learning has seen a considerable increase in research attention in recent years. There are three main practical problems for which continual learning could be a solution [94]: Firstly, systems might have memory restrictions making it physically impossible to store much data from previous tasks (e.g., in robotics applications). Secondly, government legislation and privacy consideration can prohibit the storing (and sharing) of data from previous tasks. Thirdly, the paradigm of joint training, where a model is retrained from scratch when new data arrives is very energy inefficient, and continual learning which can build upon previously learned models can be a more sustainable alternative.

A large part of the current research on continuous learning is focused on image classification [66, 75, 119]. Several excellent surveys [26, 94] for this topic exist, and they introduce CL methods by several broad categories. In this chapter, we introduce the main techniques that are used in continual learning. We will start from the typical setting for continual learning for classification.

2.1 Continual learning settings for classification

In the continual learning setting, the model learns from a sequence of datasets $[D_1, D_2, ..., D_T]$, where T is the number of tasks. For the classification problem, each dataset contains a number of data pairs (x, y), where x is an image and $y \in Y_t$ is a label. In general, each dataset $[D_1, D_2, ..., D_T]$ is disjoint to each other and the labels from each dataset $[Y_1, Y_2, ..., Y_T]$ are also disjoint to each other.

Currently, there are three main different characteristics that define the continual learning setup. The first one is whether the task-ID is given during inference. Following the terminology of Van de Ven et al. [143], most of the work can be classified as Task Incremental Learning(TIL) [66, 84, 124] or Class Incremental Learning(CIL) [27, 54, 119]. TIL refers to the setting where, during testing, each data point comes with a task-ID τ which denotes from which dataset D_τ it comes. So the label can be picked from Y_τ. In CIL on the contrary, the task-ID τ is not given, so the label has to be chosen from all learned labels $[Y_1 \cup ... \cup Y_T]$. This makes the CIL setup more challenging than the TIL setup.

The second characteristic is whether the application allows for the storage of any data from previous tasks. For the more restrictive set up, the usage of exemplars (or a buffer) is not allowed. Methods which require exemplars such as iCaRL [84, 119] and GEM [84, 119] can not be applied to this setup, and instead one has to apply exemplar-free methods such as EWC [66] and MAS [3]. It should be noted that with the help of exemplars, it is much easier to prevent forgetting.

The third characteristics is whether the number of training epochs is limited. In *online* continual learning, one is only allowed to see all data once. Several methods have been proposed for this situation, including GEM [84] and StableSGD [55]. This situation more realistically mimics the streaming scenario, where a stream of data arrives, and the learner can only process every sample once (except for those samples that are selected for the buffer). However, the majority of the continual learning methods do not consider this restriction, and the number of training epochs is not limited just like for the regular classification problem.

In principle, when comparing two methods, all these three mentioned characteristics should be set to be the same, otherwise the comparison is not fair.

2.2 Techniques for continual learning

2.2.1 Parameter Regularization

There are two main regularization approaches in continual learning, namely parameter and data regularization. We will first focus on parameter regularization. These

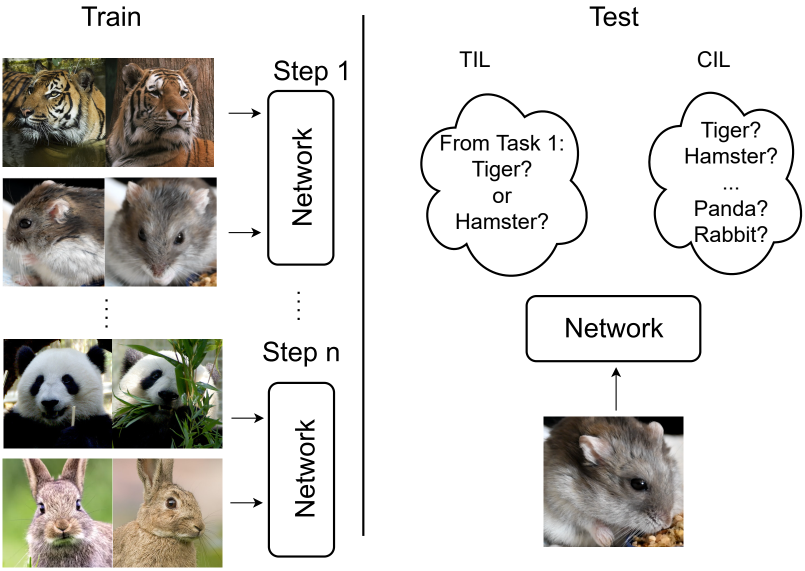


Figure 2.1: **Taks incremental Learning (TIL) vs Class Incremental Learning (CIL).** During testing, in the TIL setting, the model is aware of which task the sample comes from and in the CIL the model is agnostic of that. As a result, CIL is a more challenging setting than TIL.

works are in general based on the idea that one should look for optimal parameters on the new task that do not incur a high loss on previous tasks. Since the loss on previous tasks cannot be directly measured (we have no longer access to its data), we need to estimate it.

Kirkpatrick et al. [66] propose Elastic Weight Consolidation(EWC) to address the problem of catastrophic forgetting. They apply a quadratic penalty on the parameters of the network to constrain it to stay in the low error region for the previous task, as follows:

$$\mathcal{L}_{\text{reg}} = \frac{1}{2}\sum_i \Omega_i (\theta_i^{t\text{-}1} - \theta_i^t)^2, \tag{2.1}$$

They evaluate a stiffness (importance weights) Ω_i for each parameter θ_i, instead of applying a uniform penalty on all parameters($\Omega_i = 1$). The importance weights are

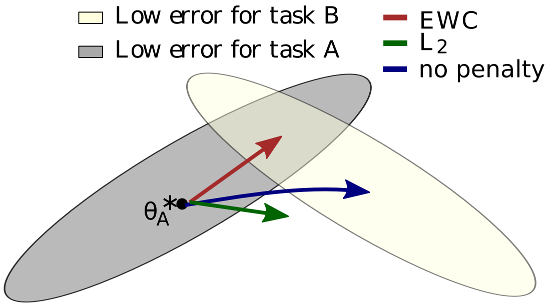


Figure 2.2: Parameter regularization methods in general aim to regularize the weights based on their importance for previous tasks. This figure shows a graphical overview for Elastic Weight Consolidation(EWC) where the ellipsoids represent the low error regions for the two tasks A and B. Parameter regularization prevents the learner from finding an optimal that is only good for task B and would incur catastrophic forgetting for task A (this figure is taken from [66]).

defined as the diagonal of the Fisher information matrix F. This can be computed from first-order derivatives according to [110]:

$$\Omega_i = E_{x \sim D_{t-1}} \left[\left(\frac{\partial L(x,\theta)}{\partial \theta_i} \right)^2 \right] \tag{2.2}$$

Here $L(x,\theta)$ is the loss of the task given the datapoint x and parameters θ. The basic idea is illustrated in Figure 2.2. In this paper, they show that EWC can prevent catastrophic forgetting on both visual classification problems, and on reinforcement learning problems where they learn a series of Atari games.

EWC is based on the assumption that the Fisher matrix can be approximated by a diagonal matrix. To improve upon EWC, rotated-EWC (or R-EWC) [81] has been proposed. The method rotates the parameter space to minimize the error introduced by diagonal-assumption of the Fisher Information Matrix approximation. Rotation of the parameter space is done by splitting each layer in such way to optimize approximation by a diagonal matrix on the resulting network.

Zenke et al. [172] proposed Synaptic Intelligence(SI). Similar to EWC [66]. They also apply a quadratic penalty on the changing of parameters, as Equation (2.1).

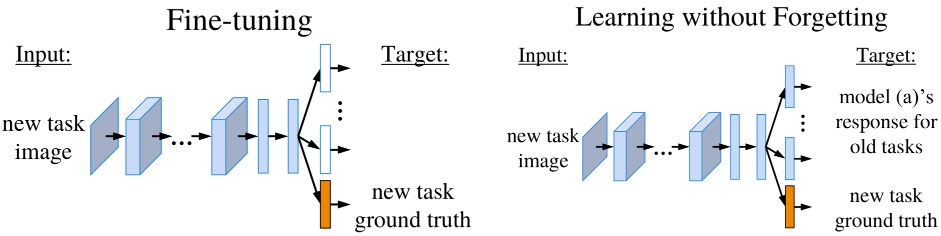


Figure 2.3: **Data regularization:** (1) Fine-Tuning(FT): The feature extractor and the new classifier are trained with new images and ground truth labels (2) Learning without Forgetting(Lwf) [75]: The output of the new images from the new classifier are trained with the ground truth labels as target like in FT. At the same time, a knowledge distillation loss is applied between the output of the new images from the old classifier and the output from the previous model (This figure is taken from [75]).

They compute the importance weights Ω_i by a path integral during the training.

$$\Omega_i^t = \sum_{\tau=1}^{t-1} \frac{w_i}{(\Delta_i^\tau)^2 + \xi} \tag{2.3}$$

where w_i stands for the contribution to a drop in the loss of each individual parameter. It can be approximated by the running sum of the gradient $\partial L / \partial \theta_i$. Δ_i^τ stands for the distance that parameters move from the previous task $\Delta_i^\tau = \theta_i^\tau - \theta_i^{\tau-1}$, and ξ is a damping parameter.

In [3], Aljundi proposed Memory Aware Synapses(MAS). They calculate the importance weights as the sensitivity of the learned function.

$$\Omega_i = E_{x \sim D_{t-1}} \left[\frac{\partial \| h_{t-1}(f_{t-1}(x)) \|_2^2}{\partial \theta_i} \right] \tag{2.4}$$

$f_{t-1}(\cdot)$ is the feature extractor for task $t-1$ and $h_{t-1}(\cdot)$ is the classifier for task $t-1$. They also provide a local version of the method, and they show that it is similar to the Hebbian learning theory [48] which provides an explanation for the phenomenon of synaptic plasticity in neuroscience.

2.2.2 Data Regularization

Li et al. [75] proposed Learning without Forgetting(LwF). In their method, they propose to use the knowledge distillation loss [50] to address the catastrophic forgetting problem in continual learning. More specifically, it is a modified cross entropy loss that encourage the new output probability to approximate the old one:

$$\mathscr{L}_{\text{reg}} = E_{x\sim D_t}\left[\sum_{\tau=1}^{t-1} \sigma(h_{t-1}^{\tau}(f_{t-1}(x)))\log\sigma(h_t^{\tau}(f_t(x)))\right], \tag{2.5}$$

$f_t(\cdot)$ is the feature extractor at task t. $h_t^{\tau}(\cdot)$ is the task-specific classifier for task τ when learning task t. $\sigma(\cdot)$ is the softmax activation function with a temperature.

The method is originally proposed for the task incremental learning(TIL) setting with no exemplars. However, it can also be easily adapted to CIL (Class Incremental Learning) with exemplars (see e.g. [161]):

$$\mathscr{L}_{\text{reg}} = E_{x\sim D_t \cup M}\left[\sigma(h_{t-1}(f_{t-1}(x)))\log\sigma(h_t(f_t(x)))\right], \tag{2.6}$$

Here the task specific classifiers $[h^1, h^2, ..., h^t]$ are replaced by a unified classifier h. h_t and h_{t-1} stand for the unified classifier at task t and task $t-1$ respectively. And the x is sampled from $D_t \cup M$ where M is the set of saved exemplars. The learning without forgetting method without exemplars was found to be very efficient in countering catastrophic forgetting, and performed among the best exemplar-free methods in a recent survey [94].

Ahn et al. [2] proposed Separated Softmax for Incremental Learning(SSIL). They apply a KL divergence loss (equivalent to the knowledge distillation loss, since the old output is not changing) on a task-wise probability instead of global probability. Previously in CIL, the probability is evaluated on a unified classifier $\sigma(h(\cdot))$, but here they separate the output of the classifier by task and then evaluate the probability $\left[\sigma(h^1(\cdot)), \sigma(h^2(\cdot)), ..., \sigma(h^t(\cdot))\right]$.

$$\mathscr{L}_{\text{reg}} = E_{x\sim D_t \cup M}\left[\sum_{\tau=1}^{t-1} D_{KL}(\sigma(h_{t-1}^{\tau}(f_{t-1}(x)))\,\|\,\sigma(h_t^{\tau}(f_t(x))))\right], \tag{2.7}$$

The regularization loss is applied on these task-wise probabilities. This small change of the original LwF methods leads to remarkable performance gains.

In [54], Hou et al. proposed LUCIR (Learning a Unified Classifier Incrementally via Rebalancing). They apply the regularization loss on the feature embedding $f(x)$ with a cosine similarity, instead of the output probabilities $\sigma(h(f(x)))$ as in LwF:

$$\mathscr{L}_{\text{reg}} = E_{x\sim D_t \cup M}\left[1 - cos(f_{t-1}(x_i), f_t(x_i))\right], \tag{2.8}$$

Another approach is proposed by Liu et al. [82]. They propose a method called Generative Feature Replay(GFR). They propose a feature distillation loss which is a

L_2 norm on the feature embedding.

$$\mathscr{L}_{\text{reg}} = E_{x \sim D_t}\left[\|(f_{t-1}(x) - f_t(x)\|_2\right], \tag{2.9}$$

GFR is an exemplar-free method, so the data is only sampled from current dataset D_t when evaluating regularization loss. Experiments show that feature distillation loss itself is not very effective but when it is combined with feature replay (explained further in section 2.2.8), it outperforms other exemplar free method with a large margin.

In [12], Buzzega et al. propose DER (Dark Experience Replay). The idea is to align the output between a new and old network given a data point x sampled from the memory M. Different from other methods where the output of old network are computed from the previously stored model, in DER, they store output logits $z = h_\tau(f_\tau(x))$ after task τ where these samples are firstly trained. The regularization loss is as follows:

$$\mathscr{L}_{\text{reg}} = E_{(x,z)\sim M}\left[\left\|z - h_t(f_t(x))\right\|_2^2\right]. \tag{2.10}$$

They also proposed DER++ where they add a standard cross entropy loss on the exemplars saved in the memory M with the ground truth label as described in section 2.2.6. In general, their method is a combination of knowledge distillation and rehearsal. And the method is also compatible with the setting of blurred task boundaries (where tasks are partially overlapping).

The previously discussed methods apply the distillation loss on the end of the network or feature extractor. Instead, Douillard et al. [27] use the regularization loss on intermediate feature layers. The feature of the intermediate layer is 3-dimensional(channel, height, width). Applying directly a L_2 loss is too rigid, leading to low plasticity. Therefore they propose Pooled Outputs Distillation(PoDNet) that applies a feature distillation loss after a pooling operator according to:

$$\mathscr{L}_{\text{reg-flat}} = E_{(x)\sim D_t \cup M}\left[\sum_{c=1}^{C}\left\|\sum_{h=1}^{H}\sum_{w=1}^{W} f_{t-1}^{l,c,w,h}(x) - f_t^{l,c,w,h}(x)\right\|^2\right]. \tag{2.11}$$

However, they also find this loss it too loose and a good trade-off between plasticity and prevention of forgetting is obtained when they take the sum of the result of applying pooling (summation) on the feature in the height or width direction respectively:

$$\mathscr{L}_{\text{reg-spatial}} = E_{(x)\sim D_t \cup M}\left[\sum_{l=1}^{L-1}\sum_{c=1}^{C}\sum_{h=1}^{H}\left\|\sum_{w=1}^{W} f_{t-1}^{l,c,w,h}(x) - f_{t}^{l,c,w,h}(x)\right\|^2\right] \tag{2.12}$$

$$+E_{(x)\sim D_t \cup M}\left[\sum_{l=1}^{L-1}\sum_{c=1}^{C}\sum_{w=1}^{W}\left\|\sum_{h=1}^{H} f_{t-1}^{l,c,w,h}(x) - f_{t}^{l,c,w,h}(x)\right\|^2\right], \tag{2.13}$$

here $f_t^{l,c,w,h}(x)$ is the feature at layer l, channel c and spatial position w,h. The final loss is defined as the sum of equation 2.11 and 2.12 where the flat loss is only applied on the last layer. This method outperforms all the previous method in the setting of CIL with half of all classes as the starting task(CIL Half).

Kang et al. [58] propose Adaptive Feature Consolidation(AFC). They apply the regularization loss on all the features $z_{l,c}$, where l and c are layer and channel indices respectively, with an importance factor $I_{l,c}$. The factors are calculated as the sensitivity of that feature according to:

$$I_{l,c} = E_{x,y\sim D_{\text{data}}^{t-1}}\left[\left\|\nabla_{Z_{l,c}} L(x,y)\right\|^2\right],$$

And the final regularization loss is given by:

$$\mathscr{L}_{\text{reg}} = E_{x\sim D_{\text{data}}^{t}}\left[\sum_{l=1}^{L}\sum_{c=1}^{C} I_{l,c}\left\|f_{t-1}^{l,c}(x) - f_{t}^{l,c}(x)\right\|^2\right], \tag{2.14}$$

The idea that they evaluate the importance for features is similar to those methods in Section 2.2.1 that evaluate importance to weights. This method further outperforms PoDNET [27], which is the previous state-of-the-art for CIL Half setting.

2.2.3 Gradient Update Modification

This type of techniques address the forgetting problem by modifying the update gradient g directly.

In [84], Lopez-Paz and Ranzato propose to ensure that the update gradient g does no increase the loss of previous task. They apply it as an inequality constraint to keep the inner product non-negative between the new gradient $\tilde{g}$ to all the gradient evaluated on the saved exemplar of the previous task $\tilde{g_k}$, according to:

$$\text{minimize}_{\tilde{g}} \quad \frac{1}{2}||g-\tilde{g}||_2^2 \quad \text{s.t.} \quad \tilde{g}^\top g_k \geq 0 \text{ for all } k < t. \tag{2.15}$$

The constraint is imposed on all previous tasks k (smaller than the current task t).

Following the idea of GEM [84], Chaudhry et al. [15] proposed Averaged GEM(A-GEM) to improve GEM by averaging all the previous gradients g_k to a single g_{ref}.

$$\text{minimize}_{\tilde{g}} \quad \frac{1}{2}||g-\tilde{g}||_2^2 \quad \text{s.t.} \quad \tilde{g}^\top g_{ref} \geq 0 \tag{2.16}$$

The experiments show that A-GEM is much faster than the original GEM while maintaining a similar accuracy.

Zeng et al. proposed Orthogonal Weights Modification(OWM) [171]. They aim to update the network in a space that is orthogonal to the input space A which consists of all previously trained input vectors. They construct a projector:

$$P = I - A\left(A^T A + \alpha I\right)^{-1} A \tag{2.17}$$

to project the gradient $\tilde{g} = Pg$. Here α is a small constant. To avoid saving all previous inputs A, they use Recursive Least Square(RLS) algorithm to calculate P iteratively.

Farajtabar et al. [30] proposed to store a subset of gradients from each task. They construct a basis from the store gradient using the Gram-Schmidt procedure. And the new update gradient that can be projected to the existing basis will be removed.

Finally, Saha et al. [124] proposed to use Singular Value Decomposition(SVD) on the network representations of each task and store only the important bases in the gradient projection memory M (GPM). When learning new tasks, the component existing in the GPM bases will be removed directly $\tilde{g} = g - MM^T g$. After the training, a few more bases that are extracted from the new representation and orthogonal to the existing M will be added to GPM.

2.2.4 Mask mechanism for parameter isolation

Mask mechanism shares a similar idea to parameter regularization (Section 2.2.1), where they aim to counter catastrophic forgetting by preventing the update of parameters which are relevant for previous tasks. However mask mechanism take this idea to a further step, parameter isolation. They use only a limited set of parameters for each task and not update these parameter in the future task.

Mallya et al. [92] propose Packnet that stands for "pack multiple tasks into a single network". The idea is to prune a certain amount of weights in the network to allow further learning of the new coming task. The pruning follows the method introduced by [43], for each layer they apply a binary mask that keeps only the highest weights(taking a fixed percentage) and the rest of them are set to zero. Then they finetune the network again with the mask, that is to say the weights that are set

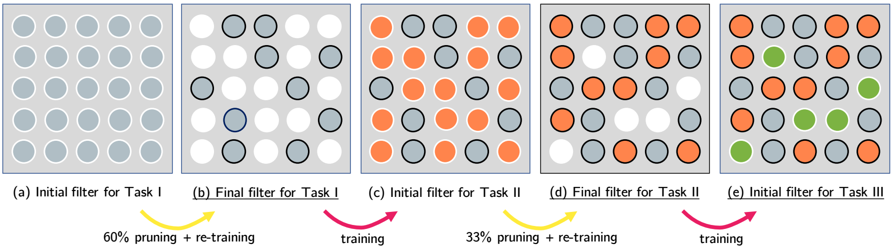

(a) Initial filter for Task I
(b) Final filter for Task I
(c) Initial filter for Task II
(d) Final filter for Task II
(e) Initial filter for Task III
60% pruning + re-training
training
33% pruning + re-training
training

Figure 2.4: Packnet: After training of each task, they apply a pruning to the filters (find low value filter and set them to zero). When training a new task, the weights left from the previous task are fixed and only the empty filters are trained. The new task can also use the features trained from previous task. (this figure is taken from [92]).

to zero remain as zero and only the preserved weights are trained. After the training of each task, they freeze the weights trained in the current stage. And in the later tasks, only the weights that are set to zero currently will be trained. In this paper, the ratio of the pruning are pre-defined depending on the number of task.

In [91], Mallya et al. follow up on the idea of Packnet and propose Piggyback. They learn a binary mask on an existing network where the weights are fixed. In the training, they define real-valued mask weights $\mathbf{m^r}$ and compress them to a binary value mask $\mathbf{m^r}$ by a selected threshold. They multiply the weights $\mathbf{W}$ by the mask $\mathbf{m}$ according to $\hat{\mathbf{W}} = \mathbf{W} \odot \mathbf{m^r}$ element-wise. The network weights $\mathbf{W}$ are fixed and only the mask weights $\mathbf{m^r}$ and a final classification layer are updated by back propagation. After the training, the real-valued mask $\mathbf{m^r}$ is no longer required and only the binary mask is stored for inference. Different from Packnet this work shows that the weights of the network do not have to be trained at all, and that by only training a binary mask on top of a pretrained network can achieve excellent results. Recently, Wortsman et al. [158] continued this research line. They show — surprisingly — that good performance can be obtained by learning masks upon a randomly initialized network.

Serra et al. [128] propose to move the mask from the network weights to the features (or activations) of the network. This has the large advantage that the memory overhead of the masks is significantly lower (e.g. AlexNet has 54M weights and only 10k features). Their method is called Hard Attention to the Task(HAT). They define the attention vector(mask) $\mathbf{a} \in [0,1]$ at the feature level. The features are multiplied by the mask $\mathbf{a}$ element-wise $\hat{\mathbf{h}} = \mathbf{h} \odot \mathbf{a}$. The mask $\mathbf{a}$ is computed by

$\mathbf{a} = \sigma(s\mathbf{e})$ where $\sigma(\cdot)$ is the sigmoid function, s is a scaling parameter and $\mathbf{e}$ is a task embedding vector. When obtained the attention vector $\mathbf{a}_l^t$ of layer l at task t, a cumulative attention vector $\mathbf{a}_l^t = \max\left(\mathbf{a}_l^t, \mathbf{a}_l^{t-1}\right)$ is calculated. During the training, they update less for those features that are more important for the previous task, defined by an accumulative attention vector. To be more specific, they modify the gradient $g_{l,ij}$, the weight for the $l-1$ layer's jth channel to the l layer's ith channel, as:

$$\tilde{g}_{l,ij} = \left[1 - \min\left(a_{l,i}^{\leq t}, a_{l-1,j}^{\leq t}\right)\right] g_{l,ij}, \tag{2.18}$$

Their method was further developed into binary masks [95] and extended to recurrent networks [25].

In [116], Rajasegaran proposed Random Path Selection(RPS). Instead of applying a mask on weights [91, 92] or features [128], they apply the mask on modules of the network. The RPS is based on a modification of ResNet [46]. For each layer, the single block between skip connections is replaced by M parallel modules(similar to the single ResNet block). $\mathbf{P} \in \{0,1\}^{L\times M}$ (L is number layers) is defined as binary masks for all layers and modules. It is called path in this paper. Similar to [91], if the value $P(l,m) = 0$, the module m at layer l is skipped and if it is equal to 1 the module is validated. The path when training task t denote as $\mathbf{P}_t^{tr}$. For each layer, one and only one module is activated

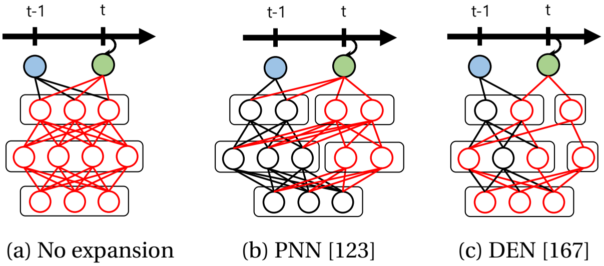
$$\sum_{m=1}^{M} \mathbf{P}_t^{tr}(l,m) = 1 \qquad \forall l = 1,2,\cdots,L, \tag{2.19}$$

The mask for the inference $\mathbf{P}_t^{ts}$ is the combination of all the modules that once selected as the training path. So it is capable of Class Incremental Learning(CIL) method because the task-id is not required in the inference.

$$\mathbf{P}_t^{ts}(l,m) = \mathbf{P}_0^{tr}(l,m) \vee \mathbf{P}_1^{tr}(l,m) \cdots \vee \mathbf{P}_t^{tr}(l,m), \tag{2.20}$$

During the training, N training paths are randomly initialized. And the modules that exist in the previous path $\mathbf{P}_{t-1}^{ts}$ are freezed. After the training, they select the best path based on the performance as the final training path $\mathbf{P}_t$. In their implementation, to improve the training speed, they perform the path selection for each J tasks, J is a fixed predefined hyperparameter, and experiments show that it does not jeopardize the performance.

In conclusion, masking methods in general aim to use only a limited set of network weights (or features) for a particular task. These methods allow for forward transfer, meaning that future tasks can exploit the features learned in previous tasks.



(a) No expansion (b) PNN [123] (c) DEN [167]

Figure 2.5: **Graphical explanation of Expandable Networks:** (a) No expansion: different tasks on trained on same parameters (b) Progressive Network [123]: After training a task, previous network is fixed and few new neurons are added to each layer. The new task is trained only on the new neurons while the previous features can also be utilized (c) DEN [167]: the previous network is retrained selectively on the new task. And the new neurons are added adaptively by adding a sparsity regularization. (This figure is taken from [167]).

However, they do not allow (or severely limit) the possibility of positive backward transfer, since the new knowledge cannot be exploited by previous tasks. They are typically applied in the task incremental learning setting, since most of them require a task-ID at inference time.

2.2.5 Expandable Networks

The techniques discussed in this section are closely related with those discussed in the previous section on masking methods. Instead, other than the methods introduced in previous sections, where the size of network is fixed from the first task till the last, in Rusu et al. [123], they propose Progressive Neural Networks(PNN) that expand the network for each new task. After training on a task, the existing parameters for the network are frozen and only a few neurons(features) are added to every layer of the network. When training the new task, old parameters are not updated but the new network can still use the features that output from the old network. The experiments are mainly on reinforcement learning and the results show transfer of knowledge exists in both low-level and high-level parts of the network.

Inspired by the PNN, Joon et al. [167] proposed Dynamically Expandable Networks(DEN). When learning a new task, they first train a new last layer using the feature output from the previous network. Then they fix part of the network and only retrain the part that has been selected as relevant weights using the newly trained last layer from previous step. They add new neurons adaptively for each layer with a group-sparsity regularization [5, 157] to prevent it from growing too much. They also measure the drift of parameter and apply a parameter regularization loss as introduced in Section 2.2.1. In addition to that, when a parameter drifts too much (more than a given threshold), they will make a duplicate of it. Experiments shows that it outperforms PNN [123].

Yan et al. [164] proposed Dynamically Expandable Representation(DER) which adapt HAT [128] to the CIL(Class Incremental Learning) scenario. In the vanilla version of HAT [128], a task-ID is needed in the inference to use the correct mask, so the method is only available for the TIL setting. In DER, when training a new task t, the feature extractor $F_t(\cdot)$ is trained with a channel mask $\mathbf{a} = \sigma(s\mathbf{e})$ where $\sigma(\cdot)$ is the sigmoid function, s is a scaling parameter and $\mathbf{e}$ is a task embedding vector. After the training they assign s a large value so the mask turns into a binary mask and they obtain a pruned network $F_t^P(\cdot)$ from $F_t(\cdot)$. They construct the super-feature extractor Φ_t by expanding the previous super-feature extractor Φ_{t-1} with the new feature extractor F_t^P.

$$\Phi_t(x) = [\Phi_{t-1}(x), F_t(x)], \tag{2.21}$$

where the $\Phi_{t-1}(x)$ is composed by a pruned network of previous feature extractors $\Phi_{t-1}(x) = [F_1^P(x), F_2^P(x), ..., F_{t-1}^P(x)]$. In addition, they also proposed a sparsity loss to encourage reducing the number of parameters.

$$\mathscr{L}_S = \frac{\sum_{l=1}^{L} K_l \|\mathbf{a}_{l-1}\|_1 \|\mathbf{a}_l\|_1}{\sum_{l=1}^{L} K_l c_{l-1} c_l}, \tag{2.22}$$

where K_l is the kernel size of the convolution layer l. Based on the expanded feature extractor, they re-train a classifier head for each task. Experiments show that the new method outperforms existing CIL method [27, 116] with fewer parameters.

2.2.6 Experience Replay

When storing few exemplars for the previous task is allowed, the idea of jointly training the new samples together with the old exemplars comes naturally. People mentioned such techniques dating back to 1990s [118, 121] and they call it experience replay(ER) or rehearsal. In deep learning era, Rebuffi et al. [119] introduced the new setting of CIL and they assumed that saving exemplars is allowed. They used rehearsal as a main strategy of their method, and most of the later works in CIL follow this setting and approach.

In [14], Chaudhry et al. study experience replay(ER) in online continual learning setting. They show that this simple technique outperforms several specially designed CL methods. They also find that even when the memory is very small (one example per class), ER still helps improve the performance. Over-fitting is not happening because the exemplars are not trained alone, but together with new data.

2.2.7 Exemplar Selection

The naive way of selecting exemplars is to select them randomly. In Rebuffi et al. [119] they adapt the concept of herding [156] for exemplar selection in continual learning. The idea is to select exemplars that best approximate the mean of all data in the feature space. The method is implemented in a greedy and iterative way. When adding or removing an exemplar, they select the exemplar that results in the minimal distance between the mean of the features of the exemplars of this class to the mean of features of all data of the class.

Other than using exemplar for replay, some methods use them to constrain the gradients. When using exemplars in the GEM [84] way (for each update, the component that increases the loss of exemplars with be removed), the method of selecting the exemplars becomes a different problem. In [4], Aljundi et al. proposed Gradient based Sample Selection (GSS). They explained that the problem is to find the intersection of the half spaces described by $\langle g, g_i \rangle \geq 0$ where g_i is the gradient calculated by the exemplars.

$$\tilde{C} = \bigcap_{g_i \in M} \{g | \langle g, g_i \rangle \geq 0\} \tag{2.23}$$

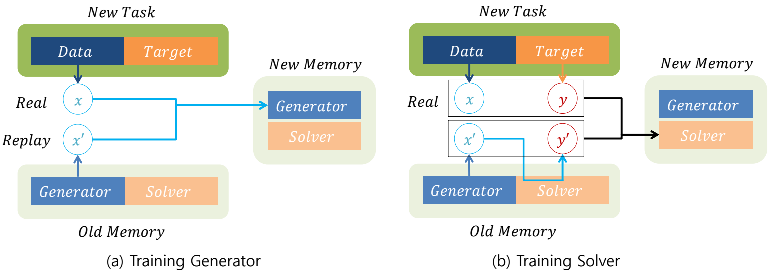
Since only part of previous samples are saved, to approximate the intersection evaluated by all the previous sample with limited exemplars, the best strategy is to find the set of exemplars that gives the smallest $\tilde{C}$. So they proposed to select the two solutions. The first one is to minimize the following equation with Integer Quadratic Programming(IQA):

$$\hat{M} \leftarrow \operatorname*{argmin}_{\hat{M}} \sum_{i,j \in \hat{M}} \frac{\langle g_i, g_j \rangle}{\|g_i\| \|g_j\|}, \tag{2.24}$$

The second one is a cheaper greedy strategy that they give a sample that less similar to the exemplars in the memory a higher chance to replace one in the memory. In the experiments they find that the second one is not worse than the IQA approach and sometimes even outperforms it.

2.2.8 Generative Replay

Because of the success of experience replay and the growing ability of generative models to generate pseudo samples that can approximate the distribution of real



(a) Training Generator

(b) Training Solver

Figure 2.6: Deep Generative Replay(DGR) [130]: The system contains a generator that generate images and a solver that predict a class label(classifier) from image. The generator is trained with the new real images and replay images generated by the previous generator. The solver(classifier) is also trained with combination while labels are provided by the previous classifer. (this figure is taken from DGR [130]).

samples, Shin et al. [130] propose Deep Generative Replay(DGR). In addition to training the classifier itself, they train a GAN (unconditionally) [38] to generate images to mimic the real data of the previous tasks. The generator and the classifier are trained separately. When learning a new task, the generator generates images of previous tasks. The new generator is trained with the real data of the new task and the generated images of previous tasks. The classifier is trained with the same images as the generator while the labels of the previous images are provided by the prediction of the previous classifier. They apply their method on MNIST [68] and the SVHN [104] dataset. Since the images from these two datasets are not difficult to generate, their method achieves very high performance, just slightly lower than exact replay of real data.

However, generation of images is computational expensive. So, in [82], liu et al. proposed Generative Feature Replay(GFR). They train a ACGAN [107] to generate features instead of entire images from a given label. Similar to DGR [130], the training of each task is divided into two stages. In the first stage, the network of ACGAN is fixed and the network of the classifer is trained with the current date sampled from D_t and the feature generated from the generator. In the second stage, the generator is trained with the feature output from the classifier and aligned with the previous generator (as we will also describe in more detail in chapter 3). The method is combined with feature distillation (explained in Section 2.2.2) and it outperforms all the previous exemplar free method and even few methods using exemplar in the setting of CIL with a large first task. In the paper, they also proposed a baseline method that the feature of every class are generated by a Gaussian distribution. It sacrifices the performance slightly but is computationally more efficient.

2.2.9 Model Generalization

Different from other techniques that prevent forgetting when learning new tasks, there are also some methods that try to learn a network that generalize better to future task thus does not tend to forget by itself. We have identified three types of methods in this scope.

Meta Learning

The first type tries to optimize this objective directly by defining a loss function. This type of methods are called meta-learning.

In [120], Riemer et al. formalized *transfer* from a sample (x_i, y_i) to another sample (x_j, y_j) as:

$$\frac{\partial L(x_i, y_i)}{\partial \theta} \cdot \frac{\partial L(x_j, y_j)}{\partial \theta} > 0, \tag{2.25}$$

Transfer means that learning data point i improves the performance of data point j and vice versa there is *interference* if the value is negative. Ideally, to encourage the transfer-ability of the network, one can optimize the following loss.

$$L = -\frac{\partial L(x_i, y_i)}{\partial \theta} \cdot \frac{\partial L(x_j, y_j)}{\partial \theta}, \tag{2.26}$$

However, the gradient of this loss requires a second derivative. So they followed the simplification introduced by Reptile [106] where only the first order Taylor expansion is considered. Meta-learning is combined with experience replay so the method is called Meta-Experience Replay(MER).

Another method was proposed by Javed et al. [57]. They proposed Online Meta Learning(OML). They divide the model into two parts: a Representation Learning Network (RLN) and a Prediction Learning Network (PLN). They apply the meta-learning method MAML [32] for RLN. In the inner loop, the RLN is frozen, and only the TLN are updated using a random sampled trajectory from the stream. Afterwards a meta-loss is computed with an additional validation batch. As the MAML simulates few-shot learning, the proposed method simulates continual learning in the inner loop and forgetting in the outer loop.

Finally, in [42], Gupta et al. proposed Lookahead-MAML(La-MAML). They introduced learnable learning rate for every parameter in the inner updates. The augmented learning objected is as follow:

$$\min_{\theta_0,\alpha} \sum_{S \sim D_t} [L(U_k(\alpha, \theta_0, S))] \tag{2.27}$$

They optimize the learning rate α together with the parameter θ_0. U_k denotes to operate the SGD operator U k times.

Self-Supervised Learning

Self-supervised learning(SSL) constructs an artificial task, e.g. rotation prediction [36], colorization [67] and contrastive learning [19], to train a feature extractor without using any label. In [182], Zhu et al. introduced SSL to continual learning because with SSL the model learns features that are not limited to current tasks but also potentially useful for future tasks. They use label augmentation for self-supervised learning proposed in [69]. For each class, images are rotated 90, 180 and 270 degrees to generate 3 extra classes. They combined this auxiliary task with Prototype Augmentation, a feature replay method they proposed which is similar to [82]. Experiments show that SSL task can help improve the performance of continual learning.

Pham et al. [111] put the SSL idea in a more central position. Inspired by the Complementary Learning Systems(CLS) theory [96] in neuroscience, they proposed a DualNet system which contains a fast learner network and a slow learner network. The slow network learns a feature representation that is more general and intrinsic using only self-supervised learning without any labels. They apply a contrastive learning method called Barlow Twins [170] in their method. For a batch of data points, they apply two different data transformations on all of them and then they calculate features for them. A cross-correlation matrix is evaluated on these two batches of features. The contrastive loss is applied on the correlation matrix to encourage the diagonal terms (same data point) to be equal to 1 and non-diagonal terms (different data points) to be equal to 0. The fast learner network learns to classify images using the features from the slow learner network and ground truth labels. They apply a mask mechanism and experience replay to prevent the forgetting of the fast learner. The fast and the slow learner complement each other, and experiments show that the method is effective and also robust to other contrastive learning objectives. Interestingly, this method can also be applied in a semi-supervised continual learning setting, since the slow learner does not require any labels, and it can exploit the information from the unlabeled data.

Training Regimes

In [55], Mirzadeh et al. make a hypothesis that forgetting is strongly affected by the flatness of the local minima. They state: "The wider the minima are, the less forgetting happens". Different from other papers that propose a method to solve the problem, they study the role of training regimes in continual learning. The effect of dropout, weight decay, learning rate, batch size and optimizer are analyzed in this paper with extensive experiments. In the end, they combine all the analysis and proposed a stableSGD that uses 1)dropout, 2)no weight decay, 3)high learning rate in the first task and decrease it across tasks, 4)small batch size, 5)SGD optimizer. The experiments show that it outperforms several existing methods.

2.2.10 Task Balancing

Wu et al. [161] point out that, in class incremental learning, a large part of the error is not introduced by the forgetting of the feature network, but just the last layer. They observe that the last fully connected layer is biased toward the new classes. So they propose to add a a simple linear model as the bias correction layer. The model only contains two trainable parameters α and β. They calculate the final output logits q_k as:

$$q_k = \begin{cases} o_k & k \text{ is an old class} \\ \alpha o_k + \beta & k \text{ is a new class} \end{cases}, \tag{2.28}$$

o_k are the output logits of the model $h(f(x))$ for class k. They split both the trainset and the exemplar dataset in to two parts. The first part is used to train the network as usual with a data regularization loss as in equation 2.6. After the training, they fix the network, and train the linear model with only the second part. This simple model is very effective for correcting the bias toward new classes and experiments showed that it was the new state of the art for the setting of CIL with a uniform division of tasks (instead of a large first task).

Similarly, in [10], Belouadah et al. propose Incremental Learning with Dual Memory(IL2M). They store the statistics of each class, in addition to the exemplars. During the inference, the prediction score is corrected by:

$$q_k = \begin{cases} o_k & k \text{ is an old class} \\ o_k \times \dfrac{\mu_k^O}{\mu_k^N} \times \dfrac{\mu^N}{\mu^O} & k \text{ is a new class} \end{cases}, \tag{2.29}$$

μ_k^O and μ_k^N stand for statistics of prediction score of class k in Old and New model respectively and μ^O and μ^N stand for averaged prediction scores of all data in Old and New model. BIC [161] and IL2M are proposed in the same period of time and share a similar idea. One of the main differences is that BIC uses data a regularization loss while IL2M does not. Despite this difference, according to experiments by Masana et al. [94], in the setting of CIL with a uniform division of tasks, both method are state-of-the-art and perform similarly. This further proves that dealing with the task imbalance is one of the most important challenges in this setting.

2.2.11 Alternative Classification Methods

A large part of the forgetting is introduced by the last classifier layer, as mentioned in Section 2.2.10. There are also methods that aim to avoid this problem by not using the standard classification approach. In [119], Rebuffi et al. propose Nearest-Mean-of-Exemplars(NME) classification. When predicting a class label for a datapoint x, they do not select the class directly from the output probability. Instead, they evaluate a prototype for each class by averaging the features of the exemplars of this class. $\mu_y = \frac{1}{|P_y|}\sum_{p\in P_y}\phi(p)$. Then they predict the class label by find the prototype with minimal distance to the feature of the datapoint.

$$y = \underset{y=1,\dots,t}{\operatorname{argmin}} \|\phi(x) - \mu_y\| \tag{2.30}$$

In [169], Yu et al. propose to train the feature extractor by triplet loss, instead of cross entropy loss. The idea is to maximize the distance between samples from different class and the minimize the distance between samples from the same class. They construct a triplet by the anchor z_a, a positive instance z_p and a negative instance z_n. The positive instance z_p and the anchor come from the same class and the negative instance z_n come from a different class.

$$L = \max(0, d_+, d_- + m) \tag{2.31}$$

where $d_+ = \|z_a - z_p\|$ and $d_- = \|z_a - z_n\|$. During the inference, they use nearest class mean (NCM) classifier.

3 Memory Replay GANs: learning to generate images from new categories without forgetting[*]

3.1 Introduction

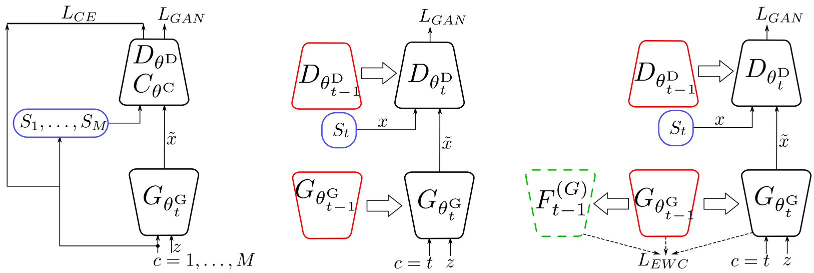
Generative adversarial networks (GANs) [38] are a popular framework for image generation due to their capability to learn a mapping between a low-dimensional latent space and a complex distribution of interest, such as natural images. The approach is based on an adversarial game between a generator that tries to generate good images and a discriminator that tries to discriminate between real training samples and generated. The original framework has been improved with new architectures [59, 114] and more robust losses [6, 41, 93].

GANs can be used to sample images by mapping a randomly sampled latent vector. While providing diversity, there is little control over the semantic properties of what is being generated. Conditional GANs [100] enable the use of semantic conditions as inputs, so the semantic properties and the inherent diversity can be decoupled. The simplest condition is just the category label, allowing to control the category of the generated image [107].

As most machine learning problems, image generation models have been studied in the conventional setting that assumes all training data is available at training time. This assumption can be unrealistic in practice, and modern neural networks face scenarios where tasks and data are not known in advance, requiring to continuously update their models upon the arrival of new data or new tasks. Unfortunately, neural networks suffer from severe degradation when they are updated in a sequential manner without revisiting data from previous tasks (known as *catastrophic forgetting* [97]). Most strategies to prevent forgetting in neural networks rely on regularizing weights [66, 81] or activations [75], keeping a small set of exemplars from previous categories [84, 119], or memory replay mechanisms [60, 121, 130].

While previous works study forgetting in discriminative tasks, in this paper we focus on forgetting in generative models (GANs in particular) through the problem of generating images when categories are presented sequentially as disjoint tasks. The closest related work is [127], that adapts elastic weight consolidation (EWC) [66] to GANs. In contrast, our method relies on memory replay and we describe two approaches to prevent forgetting by joint retraining and by aligning replays. The

[*] This chapter is based on a publication in NeurIPS 2018 [159]



(a) Joint training (b) Sequential fine tuning (c) GAN with EWC [127]

Figure 3.1: Baseline architectures.

former includes replayed samples in the training process, while the latter forces to synchronize the replays of the current generator with those generated by an auxiliary generator (a snapshot taken before starting to learn the new task). An advantage of studying forgetting in image generation is that the dynamics of forgetting and consolidation can be observed visually through the generated images themselves.

3.2 Sequential learning in GANs

3.2.1 Joint learning

We first introduce our conditional GAN framework in the non-sequential setting, where all categories are learned jointly. In particular, this first baseline is based on the AC-GAN framework [107] combined with the WGAN-GP loss for robust training [41]. Using category labels as conditions, the task is to learn from a training set $S = \{S_1, \dots, S_M\}$ to generate images given an image category c. Each set S_c represents the training images for a particular category.

The framework consists of three components: generator, discriminator and classifier. The discriminator and classifier share all layers but the last ones (task-specific layers). The conditional generator is parametrized by θ^G and generates an image $\tilde{x} = G_{\theta^G}(z, c)$ given a latent vector z and a category c. In our case the conditioning is implemented via conditional batch normalization [28], that dynamically switches between sets of batch normalization parameters depending on c. Note that, in contrast to unconditional GANs, the latent vector is completely agnostic to the category, and the same latent vector can be used to generate images of different categories just by using a different c.

Similarly, the discriminator (parametrized by θ^D) tries to discern whether an input image x is real (i.e. from the training set) or generated, while the generator tries to fool it by generating more realistic images. In addition, AC-GAN uses an auxiliary classifier C with parameters θ^C to predict the label $\tilde{c} = C_{\theta^C}(x)$, and thus forcing the generator to generate images that can be classified in the same way as real images. This additional task improves the performance in the original task [107]. For convenience we represent all the parameters in the conditional GAN as $\theta = \left(\theta^G, \theta^D, \theta^C\right)$.

During training, the network is trained to solve both the adversarial game (using the WGAN with gradient penalty loss [41]) and the classification task by alternating the optimization of the generator, and the discriminator and classifier. The generator optimizes the following problem:

$$\min_{\theta^{\mathrm{G}}} \left(L^{\mathrm{G}}_{\mathrm{GAN}}(\theta, S) + L^{\mathrm{G}}_{\mathrm{CLS}}(\theta, S) \right) \tag{3.1}$$

$$L^{\mathrm{G}}_{\mathrm{GAN}}(\theta, S) = -\mathbb{E}_{z\sim p_z, c\sim p_c}\left[D_{\theta^{\mathrm{D}}}\left(G_{\theta^{\mathrm{G}}}(z, c) \right) \right] \tag{3.2}$$

$$L^{\mathrm{G}}_{\mathrm{CLS}}(\theta, S) = -\mathbb{E}_{z\sim p_z, c\sim p_c}\left[y_c \log C_{\theta^{\mathrm{C}}}\left(G_{\theta^{\mathrm{G}}}(z, c) \right) \right] \tag{3.3}$$

where $L^{\mathrm{G}}_{\mathrm{GAN}}(\theta, S)$ and $\lambda_{\mathrm{CLS}} L^{\mathrm{G}}_{\mathrm{CLS}}(\theta, S)$ are the corresponding GAN and cross-entropy loss for classification, respectively, S is the training set, $p_c = \mathcal{U}\{1, M\}$, $p_z = \mathcal{N}(0, 1)$ are the sampling distributions (uniform and Gaussian, respectively), and y_c is the one-hot encoding of c for computing the cross-entropy. The GAN loss uses the WGAN formulation with gradient penalty [41]. Similarly, the optimization problem in the discriminator and classifier is

$$\min_{\theta^{\mathrm{D}}, \theta^{\mathrm{C}}} \left(L^{\mathrm{D}}_{\mathrm{GAN}}(\theta, S) + \lambda_{\mathrm{CLS}} L^{\mathrm{D}}_{\mathrm{CLS}}(\theta, S) \right) \tag{3.4}$$

$$\begin{aligned} L^{\mathrm{D}}_{\mathrm{GAN}}(\theta, S) &= -\mathbb{E}_{(x,c)\sim S}\left[D_{\theta^{\mathrm{D}}}(x) \right] + \mathbb{E}_{z\sim p_z, c\sim p_c}\left[D_{\theta^{\mathrm{D}}}\left(G_{\theta^{\mathrm{G}}}(z, c) \right) \right] \\ &+ \lambda_{\mathrm{GP}} \mathbb{E}_{x\sim S, z\sim p_z, c\sim p_c, \epsilon\sim p_\epsilon}\left[\left(\| \nabla D_{\theta^{\mathrm{D}}}\left(\epsilon x + (1-\epsilon) G_{\theta^{\mathrm{G}}}(z, c) \right) \|_2 - 1 \right)^2 \right] \end{aligned} \tag{3.5}$$

$$L^{\mathrm{D}}_{\mathrm{CLS}}(\theta, S) = -\mathbb{E}_{(x,c)\sim S}\left[C_{\theta^{\mathrm{C}}}\left(G_{\theta^{\mathrm{G}}}(z, c) \right) \right] \tag{3.6}$$

where ϵ are parameters of the gradient penalty term, sampled as $p_\epsilon = \mathcal{U}(0, 1)$. The last term of $L^{\mathrm{D}}_{\mathrm{GAN}}$ is the gradient penalty.

3.2.2 Sequential fine tuning

Now we modify the previous framework to address the sequential learning scenario. We define a sequence of tasks $\mathbf{T} = (1, \ldots, M)$, each of them corresponding to learning to generate images from a new training set S_t. For simplicity, we restrict each S_t to contain only images from a particular category c, i.e. $t = c$.

The joint training problem can be adapted easily to the sequential learning scenario as

$$\min_{\theta_t^{\mathrm{G}}} L_{\mathrm{GAN}}^{\mathrm{G}}\left(\theta_t, S_t\right) \tag{3.7}$$

$$\min_{\theta_t^{\mathrm{D}}} L_{\mathrm{GAN}}^{\mathrm{D}}\left(\theta_t, S_t\right) \tag{3.8}$$

where $\theta_t = \left(\theta_t^G, \theta_t^D\right)$ are the parameters during task t, which are initialized as $\theta_t = \theta_{t-1}$, i.e. the current task t is learned immediately after finishing the previous task $t-1$. Note that there is no classifier in this case since there is only data of the current category.

Unfortunately, when the network learns to adjust its parameters to generate images of the new domain via gradient descent, that very drifting away from the original solution for the previous task will cause catastrophic forgetting [97]. This has also been observed in GANs [127, 154] (shown later in Figures 3.3, 3.5 and 3.7 in the experiments section).

3.2.3 Preventing forgetting with Elastic Weight Consolidation

Catastrophic forgetting can be alleviated using samples from previous tasks [84, 119] or different types of regularization that result in penalizing large changes in parameters or activations [66, 75]. In particular, the elastic weight consolidation (EWC) regularization [66] has been adapted to prevent forgetting in GANs [127] and included as an augmented objective when training the generator as

$$\min_{\theta_t^{\mathrm{G}}} L_{\mathrm{GAN}}^{\mathrm{G}}\left(\theta_t, S_t\right) + \sum_i \frac{\lambda_{EWC}}{2} F_{t-1,i}\left(\theta_{t,i}^{\mathrm{G}} - \theta_{t-1,i}^{\mathrm{G}}\right)^2 \tag{3.9}$$

where $F_{t-1,i}$ is the Fisher information matrix that somewhat indicates how sensitive the parameter $\theta_{t,i}^{\mathrm{G}}$ is to forgetting, and λ_{EWC} is a hyperparameter. We will use this approach as a baseline.

3.3 Memory replay generative adversarial networks

Rather than regularizing the parameters to prevent forgetting, we propose that the generator has an active role by replaying memories of previous tasks (via generative sampling), and using them during the training of current task to prevent forgetting. Our framework is extended with a replay generator, and we describe two different methods to leverage memory replays.

This replay mechanism (also known as pseudorehearsal [121]) resembles the role of the hippocampus in replaying memories during memory consolidation [29], and has been used to prevent forgetting in classifiers [60, 130], but to our knowledge has not been used to prevent forgetting in image generation. Note also that image generation is a generative task and typically more complex than classification.

3.3.1 Joint retraining with replayed samples

Our first method to leverage memory replays creates an extended dataset $S'_t = S_c \bigcup_{c \in \{1,\ldots,t-1\}} \tilde{S}_c$ that contains both real training data for the current tasks and memory replays from previous tasks. The replay set $\tilde{S}_c$ for a given category c typically samples a fixed number for replays $\hat{x} = G_{\theta^{\mathrm{G}}_{t-1}}(z,c)$.

Once the extended dataset is created, the network is trained using joint training (see Fig. 3.2a) as

$$\min_{\theta_t^{\mathrm{G}}} \left(L_{\mathrm{GAN}}^{\mathrm{G}}\left(\theta_t, S'_t\right) + \lambda_{\mathrm{CLS}} L_{\mathrm{CLS}}^{\mathrm{G}}\left(\theta_t, S'_t\right)\right) \tag{3.10}$$

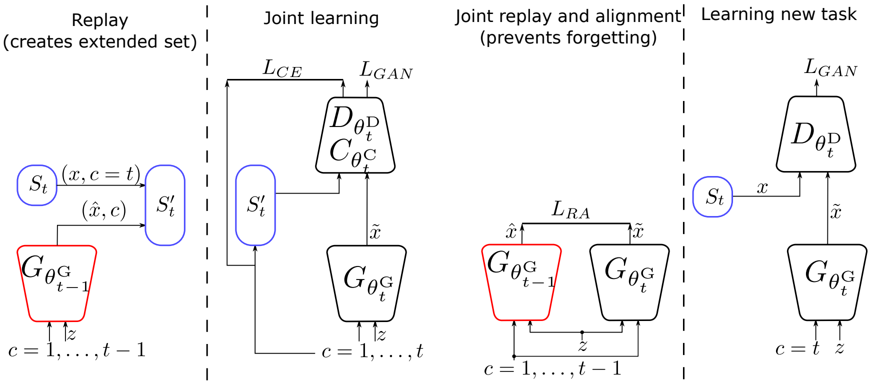
$$\min_{\theta_t^{\mathrm{D}}} \left(L_{\mathrm{GAN}}^{\mathrm{D}}\left(\theta_t, S'_t\right) + \lambda_{\mathrm{CLS}} L_{\mathrm{CLS}}^{\mathrm{D}}\left(\theta_t, S'_t\right)\right) \tag{3.11}$$

This method could be related to the deep generative replay in [130], where the authors use an unconditional GAN and the category is predicted with a classifier. In contrast, we use a conditional GAN where the category is an input, allowing us finer control of the replay process, with more reliable sampling of (x,c) pairs since we avoid potential classification errors and biased sampling towards recent categories.

3.3.2 Replay alignment

We can also take advantage of the fact that the current generator and the replay generator share the same architecture, inputs and outputs. Their condition spaces (i.e. categories), and, critically, their latent spaces (i.e. latent vector z) and parameter spaces are also initially aligned, since the current generator is initialized with the same parameters of the replay generator. Therefore, we can synchronize both the replay generator and current one to generate the same image by the same category c and latent vector z as inputs (see Fig. 3.2b). In these conditions, the generated images $\hat{x}$ and x should also be aligned pixelwise, so we can include a suitable pixelwise loss to prevent forgetting (we use L_2 loss).

In contrast to the previous method, in this case the discriminator is only trained with images of the current task, and there is no classification task. The problem



(a) Joint retraining with replay

(b) Replay alignment

Figure 3.2: Memory Replay GANs and mechanisms to prevent forgetting (for a given current task t).

optimized by the generator includes a replay alignment loss

$$\min_{\theta_t^{\mathrm{G}}} L_{\mathrm{GAN}}^{\mathrm{G}}\left(\theta_t, S_t\right) + \lambda_{\mathrm{RA}} L_{\mathrm{RA}}\left(\theta_t, S_t\right) \tag{3.12}$$

$$L_{\mathrm{RA}}\left(\theta_t, S_t\right) = \mathbb{E}_{x\sim S, z\sim p_z, c\sim\mathcal{U}\{1,t-1\}}\left[\left\|G_{\theta_t^{\mathrm{G}}}(z,c) - G_{\theta_{t-1}^{\mathrm{G}}}(z,c)\right\|^2\right] \tag{3.13}$$

Note that in this case both generators engage in memory replay for all previous tasks. The corresponding problem in the discriminator is simply $\min_{\theta_t^{\mathrm{D}}} L_{\mathrm{GAN}}^{\mathrm{D}}\left(\theta_t, S_t\right)$.

Our approach can be seen as *aligned distillation*, where distillation requires spatially aligned data. Note that in that way it could be related to the *learning without forgetting* approach [75] to prevent forgetting. However, we want to emphasize several subtle yet important differences:

Different tasks and data Our task is image generation where outputs have a spatial structure (i.e. images), while in [75] the task is classification and the output is a vector of category probabilities.

Spatial alignment Image generation is a one-to-many task with many latent factors of variability (e.g. pose, location, color) that can result in completely different images yet sharing the same input category. The latent vector z somewhat captures those factors and allows an unique solution for a given (z, c). However, pixelwise comparison of the generated images requires that not only the input but also the output representations are aligned, which is ensured in our case since at the beginning of training both have the same parameters. Therefore we can use a pixelwise loss.

Task-agnostic inputs and seen categories In [75], images of the current classification task are used as inputs to extract output features for distillation. Note that this implicitly involves a domain shift, since a particular input image is always linked to an unseen category (by definition, in the sequential learning problem the network cannot be presented with images of previous tasks), and therefore the outputs for the old task suffer from domain shift. In contrast, our approach does not suffer from that problem since the inputs are not real data, but a category-agnostic latent vector z and a category label c. In addition, we only replay seen categories for both generators, i.e. 1 to $t-1$.

3.4 Experimental results

We evaluated the proposed approaches on different datasets with different level of complexity. The architecture and settings are set accordingly. We use the Tensorflow [1] framework with Adam optimizer [63], learning rate 1e-4, batch size 64 and fixed parameters for all experiments: λ_{EWC} = 1e9, λ_{RA} = 1e-3 and λ_{CLS} = 1 except for λ_{RA} = 1e-2 on SVHN dataset.

3.4.1 Digit generation

We first consider the digit generation problem in two standard digit datasets. Learning to generate a digit category is considered as a separate task. MNIST [68] consists of images of handwritten digits which are resized 32 × 32 pixels in our experiment. SVHN [104] contains cropped digits of house numbers from real-world street images. The generation task is more challenging since SVHN contains much more variability than MNIST, with diverse backgrounds, variable illumination, font types, rotated digits, etc.

The architecture used in the experiments is based on the combination of AC-GAN and Wasserstein loss described in Section 3.2.1. We evaluated the two variants of the proposed memory replay GANs: joint training with replay (MeRGAN-JTR) and replay alignment (MeRGAN-RA). As upper and lower bounds we also evaluated joint training (JT) with all data (i.e. non-sequential) and sequential fine tuning (SFT). We also implemented two additional methods based on related works: the adaptation of EWC to conditional GANs proposed by [127], and the deep generative replay (DGR) module of [130], implemented as an unconditional GAN followed by a classifier to predict the label. For experiments with memory replay we generate one

Table 3.1: Average classification accuracy (%) in digit generation (ten sequential tasks).

	5 tasks (0-4)						10 tasks (0-9)					
	Baselines		Others		MeRGAN		Baselines		Others		MeRGAN	
	JT	SFT	EWC [127]	DGR [130]	JTR	RA	JT	SFT	EWC [127]	DGR [130]	JTR	RA
MNIST	97.66	19.87	70.62	90.39	**97.93**	**98.19**	96.92	10.06	77.03	85.40	**97.00**	**97.01**
SVHN	85.30	19.35	39.84	61.29	**80.90**	**76.05**	84.82	10.10	33.02	47.28	**66.50**	**66.78**

batch of replayed samples (including all tasks) for every batch of real data. We use a three layer DCGAN [114] for both datasets. In order to compare the methods in a more challenging setting, we keep the capacity of the network relatively limited for SVHN.

Figure 3.3 compares the images generated by the different methods after sequentially training the ten tasks. Since DGR is unconditional, the category for visualization is the one predicted by its classifier. We observe that SFT completely forgets previous tasks in both datasets, while the other methods show different degrees of forgetting. The four methods are able to generate MNIST digits properly, although both MeRGANs show sharper ones. In the more challenging setting of SVHN (note that the JT baseline also struggles to generate realistic images), the digits generated by EWC are hardly recognizable, while DGR is more unpredictable, sometimes generates good images but often generating images with ambiguous digits. Those generated by MeRGANs are in general clear and more recognizable, but still showing some degradation due to the limited capacity of the network.

We also trained a classifier with real data, using classification accuracy as a proxy to evaluate forgetting. The rationale behind is that in general bad quality images will confuse the classifier and result in lower classification rates. Table 3.1 shows the classification accuracy after the first five tasks (digits 0 to 4) and after the ten tasks. SFT forgets previous tasks so the accuracy is very low. As expected, EWC performs worse than DGR since it does not leverage replays, however it significantly mitigates the phenomenon of catastrophic forgetting by increasing the accuracy from 19.87 to 70.62 on MNIST, and from 19.35 to 39.84 on SVHN compared to SFT in the case of 5 tasks. The same conclusion can be drawn in the case of 10 tasks. By using the memory replay mechanism, MeRGANs obtain significant improvement compared to the baselines and the others related methods. Especially, our approach performs about 8% better on MNIST and about 21% better on SVHN compared to the strong baseline DGR in the case of 5 tasks. Note that our approach achieves about 12% gain in the case of 10 tasks, which shows that our approach is much more stable with increasing number of tasks. In the more challenging SVHN dataset, all methods decrease in terms of accuracy, however MerGAN are able to mitigate forgetting and

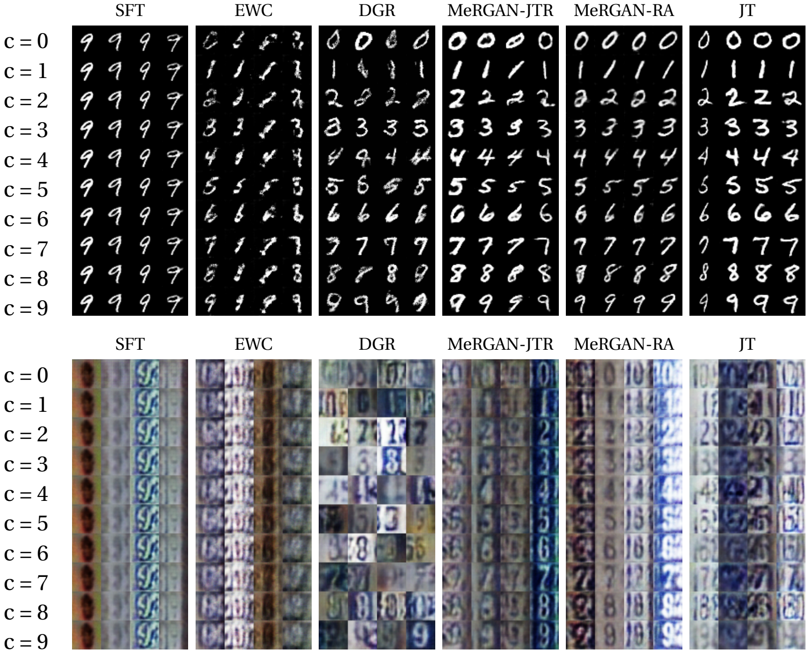


Figure 3.3: Images generated for MNIST and SVHN after learning the ten tasks. Rows are different conditions (i.e. categories), and columns are different latent vectors.

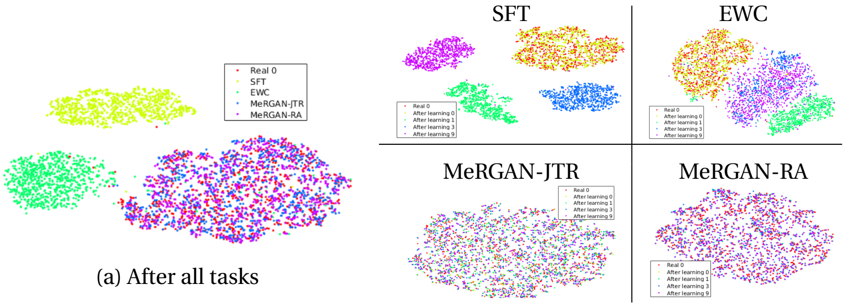
obtain comparable results to JT.

In addition to that, we also evaluated the reverse accuracy by training a classifier with the data generated by the MeRGAN and testing it on the real dataset. The result are shown in table 3.2.

On the MNIST dataset, both methods, MeRGAN-JTR and MeRGAN-RA, produce very high accuracy in 5 tasks setting (digits 0 to 4), 0.992 and 0.985 respectively, and keep relatively high accuracy in 10 tasks setting, 0.968 and 0.939 respectively. However, on the SVHN dataset there is a huge drop compared with the direct classification accuracy. In the 10 tasks setting, the reverse classification accuracy of JTR is only 0.201. We found that this drop in performance can be reduced by improving the architecture of the GAN. Therefore, we also trained a GAN with ResNet-18 network [46]. This network was further improved by replacing the AC-GAN with cGAN with projection discriminator [102] and using one-hot conditioning [100].

Table 3.2: Direct and reverse classification accuracy (%) in digit generation (ten sequential tasks).

	5 tasks (0-4)				10 tasks (0-9)			
	MeRGAN-JTR		MeRGAN-RA		MeRGAN-JTR		MeRGAN-RA	
	direct	reverse	direct	reverse	direct	reverse	direct	reverse
MNIST	97.93	99.15	98.19	98.55	97.00	96.83	97.01	93.86
SVHN	80.90	40.71	76.05	74.83	66.50	20.07	66.78	49.43
SVHN(ResNet-18)	82.30	81.12	81.74	82.30	73.29	51.00	78.56	70.98



(a) After all tasks

(b) After tasks 0,1,3,9

Figure 3.4: t-SNE visualization of generated 0s. Real 0s correspond to red dots. Please view in electronic format with zooming.

The results of JTR and RA are 51.0 and 71.0 respectively. It shows that result of the reverse classification accuracy can be improved by using a better GAN. It also shows that in a more complex case, the RA works better than JTR.

Another interesting way to compare the different methods is through t-SNE visualizations. We use a classifier trained with real digits to extract embeddings of the methods to compare. Fig. 3.4a shows real 0s from MNIST and generated 0s from the different methods after training 10 tasks (i.e. the first task, and therefore the most difficult to remember). In contrast to SFT and EWC, the distributions of 0s generated by MeRGANs greatly overlap with the distribution of real 0s (in red) and no isolated clusters of real samples are observed, which suggests that MeRGANs prevent forgetting better while keeping diversity (at least in the t-SNE visualizations). Fig. 3.4b shows the t-SNE visualizations of real and 0s generated after learning 0,1,3 and 9, with similar conclusions.

3.4.2 Scene generation

We also evaluated MeRGANs in a more challenging domain and on higher resolution images (64 × 64 pixels) using four scene categories of the LSUN dataset [168]. The experiment consists of a sequence of tasks, each one involving learning the generative distribution of a new category. The sequence of categories is *bedroom*, *kitchen*, *church (outdoors)* and *tower*, in this order. This sequence allows us to have two indoor and outdoor categories, and transitions between relatively similar categories (*bedroom* to *kitchen* and *church* to *tower*) and also a transition between very different categories (*kitchen* to *church*). Each category is represented by a set of 100000 training images, and the network is trained during 20000 iterations for every task. The architectures are based on [41] with 18-layer ResNet [46] generator and discriminator, and for every batch of training data for the new category we generate a batch of replayed images per category.

Figure 3.5 shows examples of generated images. Each block column corresponds to a different method, and inside, each row shows images generated for a particular condition (i.e. category) and each column corresponds to images generated after learning a particular task, using the same latent vector. Note that we excluded DGR since the generation is not conditioned on the category. We can observe that SFT completely forgets the previous task, and essentially ignores the category condition. EWC generates images that have characteristics of both new and previous tasks (e.g. bluish outdoor colors, indoor shapes), being unable to neither successfully learn new tasks nor remember previous ones. In contrast both variants of MeRGAN are able to generate competitive images of new categories while still remembering to generate images of previous categories.

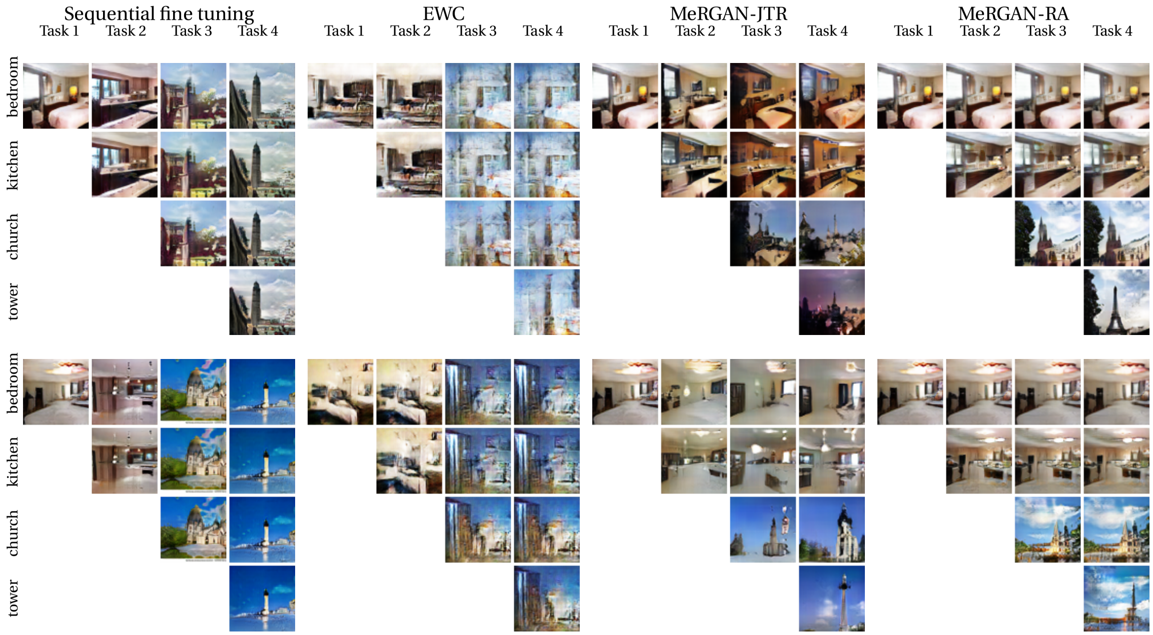


Figure 3.5: Images generated after sequentially learning each task (column within each block) for different methods (block column), two different latent vectors z (block row) and different conditions c (row within each block). The network learned after the first task is the same in all methods. Note that fine tuning forgets previous tasks completely, while the proposed methods still remember them.

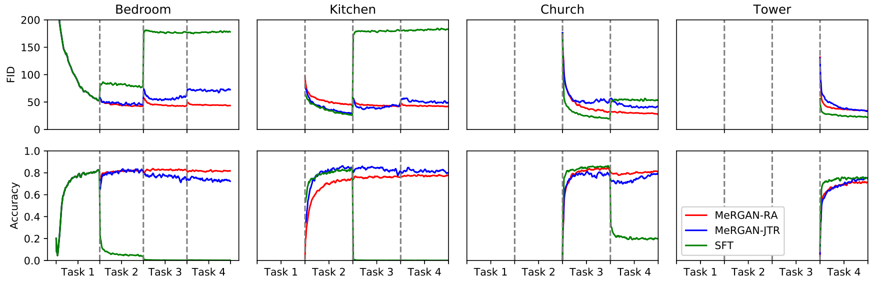


Figure 3.6: Evolution of FID and classification accuracy (%). Best viewed in color.

Table 3.3: FID and average classification accuracy (%) on LSUN after the 4th task

	SFT	EWC	DGR	MeRGAN-JTR	MeRGAN-RA
Acc.(%)	15.02	14.28	15.40	79.19	**81.03**
Rev acc.(%)	28.0	63.35	26.17	70.00	**83.62**
FID	110.12	178.05	93.70	49.69	**37.73**

In addition to classification accuracy (using a VGG [133] trained over the ten categories in LSUN), for this dataset we add two additional metrics. The first one is reverse accuracy measured by a classifier trained with generated data and evaluated with real data. The second one is the Frechet Inception Distance (FID), which is widely used to evaluate the images generated by GANs. Note that FID is sensitive to both quality and diversity [49]. Table 3.3 shows these metrics after the four tasks are learned. MeRGANs perform better in this more complex and challenging setting, where EWC and DGR are severely degraded.

Figure 3.6 shows the evolution of these metrics during the whole training process, including transitions to new tasks (the curves have been smoothed for easier visualization). We can observe not only that sequential fine tuning forgets the task completely, but also that it happens early during the first few iterations. This also allows the network to exploit its full capacity to focus on the new task and learn it quickly. MeRGANs experience forgetting during the initial iterations of a new task but then tend to recover during the training process. In this experiment MeRGAN-RA seems to be more stable and slightly more effective than MeRGAN-JTR.

Figure 3.6 provides useful insight about the dynamics of learning and forgetting in sequential learning. The evolution of generated images also provides complementary insight, as in the *bedroom* images shown in Figure 3.7, where we pay special attention to the first iterations. The transition between task 2 to 3 (i.e. *kitchen* to *church*) is particularly revealing, since this new task requires the network to learn to generate many completely new visual patterns found in outdoor scenes. The most clear example is the need to develop filters that can generate the blue sky regions, that are not found in the previous indoor categories seen during task 1 and 2. Since the network is not equipped with knowledge to generate the blue sky, the new task has to reuse and adapt previous one, interfering with previous tasks and causing forgetting. This interference can be observed clearly in the first iterations of task 3 where the walls of bedroom (and kitchen) images turn blue (also related with the peak in forgetting observed at the same iterations in Figure 3.6). MeRGANs provide mechanisms that penalize forgetting, forcing the network to develop separate filters for the different patterns (e.g. separated filters for wall and sky). MeRGAN-JTR seems to effectively decouple both patterns, since we do not observe the same "blue walls" interference during task 4. Interestingly, the same interference seems to be milder in MeRGAN-RA, but recurrent, since it also appears again during task 4. Nevertheless, the interference is still temporary and disappears after a few iterations more.

Another interesting observation from Figures 3.5 and 3.7 is that MeRGAN-RA remembers the *same* bedroom (e.g. same point of view, colors, objects), which is related to the replay alignment mechanism that enforces remembering the instance. On the other hand, MeRGAN-JTR remembers bedrooms *in general* as the generated image still resembles a bedroom but not exactly the same one as in previous steps. This can be explained by the fact that the classifier and the joint training mechanism enforce the not-forgetting constraint at the category level.

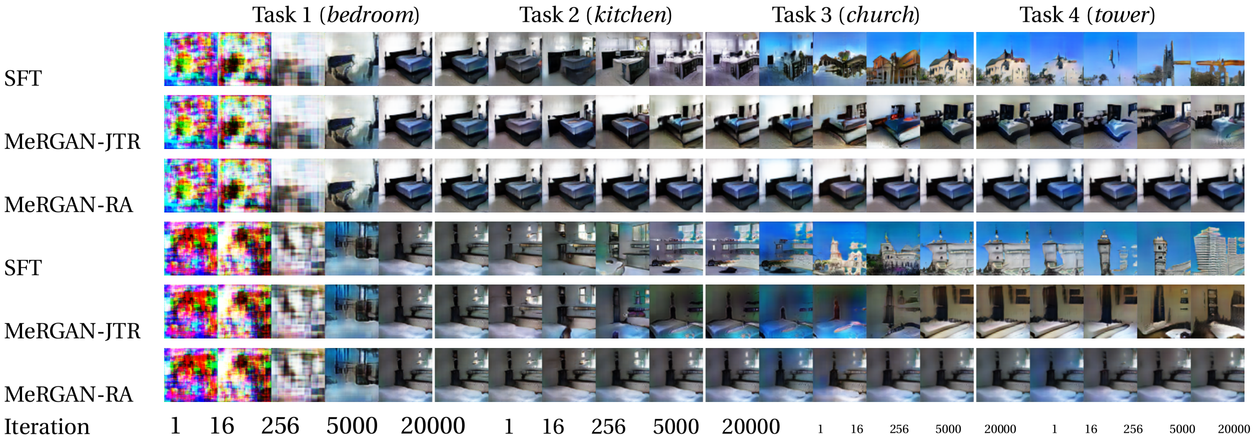


Figure 3.7: Evolution of the generated images (category *bedroom* and two different values of z) during the sequential learning process (rows). Sequential fine tuning forgets the previous task after just a few iterations (iterations within each task are sampled in a logarithmic fashion). Note that fine tuning forgets previous tasks completely, while the MeRGANs still remember them.

3.5 Conclusions

We have studied the problem of sequential learning in the context of image generation with GANs, where the main challenge is to effectively address catastrophic forgetting. MeRGANs incorporate memory replay as the main mechanism to prevent forgetting, which is then enforced through either joint training or replay alignment. Our results show their effectiveness in retaining the ability to generate competitive images of previous tasks even after learning several new ones. In addition to the application in pure image generation, we believe MeRGANs and generative models robust to forgetting in general, could have important application in many other tasks. We also showed that image generation provides an interesting way to visualize the interference between tasks and potential forgetting by directly observing generated images.

4 Positive Pair Distillation Considered Harmful: Continual Meta Metric Learning for Lifelong Object Re-Identification[*]

4.1 Introduction

Object re-identification (ReID) aims to associate the identity of a query image with those in a gallery set [47, 174]. It is applied to many applications, including person re-identification [17, 74, 166], vehicle re-identification [61, 85, 179], and face verification [148, 149]. Most existing approaches assume that the test and training dataset are drawn from the same distribution and that all training data is available jointly when training the network [74, 85, 88, 174, 179]. In domain generalization ReID [7, 22, 24, 105, 137] all source domain data is assumed available during training. This assumption is not realistic for many applications as all training data might not be available from the start and its distribution could vary over time. In addition, the trained system could be applied at inference time to new data never seen during training. Only recently, the problem of Lifelong ReID has been proposed [112]. This setting requires learning from a *sequence* of domains, and evaluates the algorithm on *unseen* domains.

Continual learning [26, 94, 97] addresses the problem of learning from non-stationary streams of data. It has developed several techniques including regularization-based methods [3, 66, 81, 172], parameter-isolation [91, 92, 128], and replay-based methods [45, 159, 161, 164]. In this chapter we consider exemplar-free continual learning where it is not allowed to save any samples (exemplars) of previous tasks for the problem of object re-identification. This requirement is out of the privacy considerations in person ReID problems.

Most continual learning methods specifically consider the incremental learning of classification problems. The considered setup for object re-identification (Fig. 4.1) is different in two main aspects. Firstly, they usually do not incrementally learn a *classifier*, instead they incrementally learn a feature representation. Secondly, the aim is to perform evaluation on new unseen tasks. So the real goal is to incrementally learn a metric space that generalizes to previously unseen tasks. Pu et al. [112] propose a method to address the first problem but ignore the second consideration: *the representation should generalize to unseen tasks.*

Meta-learning [8, 21, 32, 53, 106, 136, 144] focus on generalising to unseen tasks

[*] This chapter is based on a publication in BMVC 2022 [151].

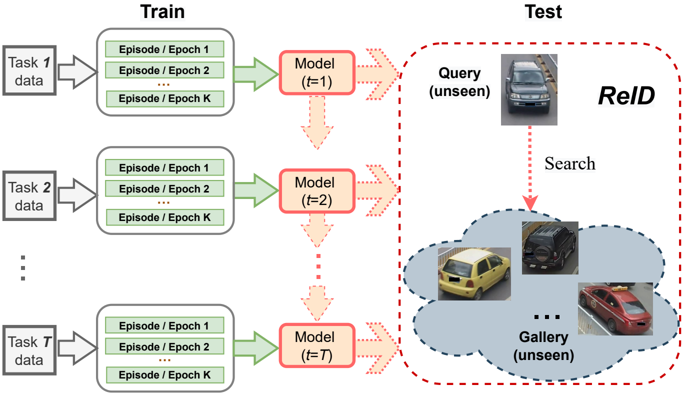


Figure 4.1: Lifelong Object ReID with continual meta-metric learning. Unlike conventional object re-identification, data are presented sequentially in discrete tasks of disjoint classes. Data from previous tasks are unavailable in successive ones and the learner must incrementally update when a new task arrives. Furthermore, in object re-identification the test identities are not seen during training, which demands generalization of the learned metric.

and has been applied to few-shot learning [9, 70, 78, 140, 153, 165]. Object ReID can be considered a few-shot learning problem, since the object identities at test time are not shown during the training and we only have few support images. To exploit the generalization capability of meta learning, Chen et al. [18] propose Deep Meta Metric Learning (DMML) that formulates the deep metric learning as a meta learning problem. Since the main challenges of object re-identification are learning from a sequences and generalization to previously unseen domains, in this chapter we propose *Continual Meta Metric Learning* to address this problem.

To further endow continual meta metric learning with a mechanism to mitigate forgetting knowledge from previous tasks, we introduce a temporary classifier for the support set and study the potential of directly applying knowledge distillation [50, 75]. However, we find that the distillation and metric learning losses are antagonistic. We therefore propose **D**istillation **w**ith**o**ut **P**ositive **P**airs (DwoPP). DwoPP, different from naive distillation, which distills knowledge from the previous to the current task classifier over *all* classes in the current task, distills only using *negative* examples. In this way, we avoid the antagonistic relationship between the metric and distillation losses which is from positive pairs distillation.

The main contributions of this chapter are: **1)** we show that meta metric learning is superior to global metric learning for object re-identification; **2)** we explicitly explore the roles of positive and negative pairs in distillation and propose a novel distillation scheme called DwoPP for Continual Meta Metric Learning; **3)** we propose task splits for evaluation of continual metric learning methods on intra-domain object ReID for three ReID datasets and evaluate on much longer sequences than the existing benchmark; and **4)** we perform extensive experimental analysis demonstrating that, DwoPP achieves significantly better performance on person and vehicle ReID, as well as on the existing LReID benchmark [112].

4.2 Related work

Object re-identification and metric learning. Metric learning has been widely applied to object re-identification [47, 174], mainly focusing on person ReID [17, 74, 166, 176], vehicle ReID [61, 85, 179] and face verification [148, 149, 176]). Deep metric learning methods can be divided into three categories based on the loss used: contrastive loss with pairwise inputs [23], triplet loss with triplet inputs [52], and N-pair loss with batch inputs [132]. In general, deep metric learning works well but does not take generalization of the learned metrics into account and neglects relationships between inter-class samples. DMML [18] formulates metric learning for object re-identification from a meta learning perspective. We build upon DMML for our *continual learning* view of meta metric learning.

Continual learning. Continual learning methods can be categorized into three groups: parameter-isolation, regularization-based and replay-based methods [26]. The most relevant to our work are regularization-based methods [3, 66, 75, 81, 169, 172]. Knowledge distillation is a widely used regularization method which decreases forgetting by either aligning features [82, 159] or the predicted probabilities [75]. To adapt knowledge distillation to Continual Meta Metric Learning, we propose a variant of knowledge distillation by introducing a temporary classifier for the current support set, and more importantly the distillation in the chapter is without considering positive pairs. Replay-based continual learning overcomes forgetting by saving a set of exemplars from each task [45, 54, 159, 161, 164]. We focus on exemplar-free continual learning. And continual learning applied to persons in particular has privacy considerations which makes retaining data problematic.

(Incremental) Meta learning. Meta learning based on metrics or optimization-based approaches are the main directions of current research [153]. ProtoNets [136] and RelationNets [141] are canonical representatives of metric-based approaches, while MAML [32] and Reptile [106] are representative optimization-based methods. Incremental meta learning (IDA [79], ERD [150]) methods have been mainly developed for incremental few-shot learning, however, they can also be applied to lifelong object ReID and we will compare to them in the experimental section. There are a few methods on incremental metric learning which approach the problem as one of representation learning with a metric-based classification loss. Examples include CRL [176], FGIR [20], and AKA [112]. However, these works all focus on distillation over seen classes and thus neglect the need to recognize unseen identities.

4.3 Methodology

4.3.1 Preliminaries

There are two main approaches to metric learning applied to object ReID: those based on global optimization of a metric embedding over the training set, and those based on episodic meta learning. Most global optimization metric learning methods minimize a metric loss over the whole dataset $D = (\mathbf{X}, \mathbf{Y})$ of inputs $\mathbf{X}$ and corresponding labels $\mathbf{Y}$. For comparison in this chapter we use the popular softmax-triplet loss as used in BoT [88].

Deep meta metric learning (DMML). In DMML [18], the authors instead formulate metric learning as a meta learning problem. They decompose the training data into a series of sub-tasks, called *episodes* in meta learning, and then learn a meta metric that generalizes well to all sub-tasks. Assuming the unseen test task is drawn from the same distribution of sub-tasks from the training set, this learned meta metric should generalize to this unseen test task.

Assume that we sample K episodes in total for training, that each episode E_k is composed of N classes, and that each class contains n_s images in the support set S_k and n_q images in the query set Q_k. In each episode, we learn the meta metric to correctly predict the query samples from support samples. The learning problem for DMML is:

$$\theta^* = \arg\min_{\theta} \mathbb{E}_{k\in[1,K]} \left[\mathscr{L}_{\text{eps}}(\theta; S_k, Q_k) \right] \tag{4.1}$$

where $\mathscr{L}_{\text{eps}}$ is the episode level hard-mining metric loss proposed in DMML [18].

An illustration of the DMML loss [18] is shown in Fig. 4.2. The hard-mining DMML loss finds the largest distance to a positive example and the smallest distance to a negative sample to compute the metric loss with a margin.

The episodic loss $\mathscr{L}_{\text{eps}}$ is defined in terms of positive and negative pairs. In the current episode E_k with the class set $\mathbb{C}$, a query point $q_c \in Q_k$ is drawn from a specific class $c \in \mathbb{C}$. We construct the *positive pairs* $[q_c, s_c]$ from the query point and support points $s_c \in S_k$ from the class c, and negative pairs $[q_c, s_{c'}]$ from the query point and

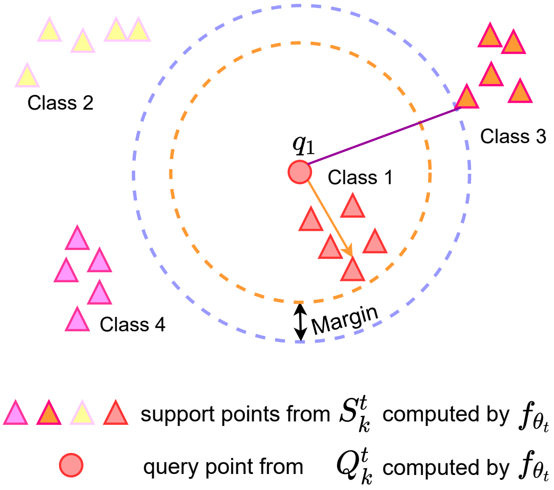

Class 2
Class 3
q_1
Class 1
Class 4
Margin
support points from S_k^t computed by f_{θ_t}
query point from Q_k^t computed by f_{θ_t}

Figure 4.2: Supposing the query point is from class 1, the hard-mining DMML loss selects the farthest positive point and nearest negative point to compute the distance. It forces a margin between negative and positive distances.

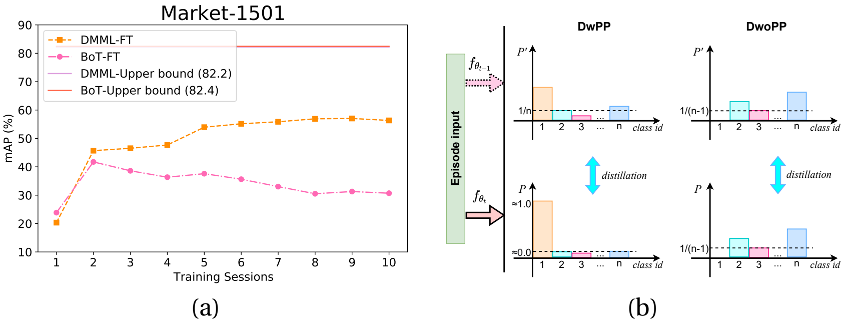
support points $s_{c'} \in S_k$ from *different* classes $c' \neq c$. Hard mining is performed over the positive pairs by finding largest Euclidean distance from q_c to a positive support sample $d_c = \max_{s_c \in S_k} d(q_c, s_c)$, and over negative pairs by finding the smallest distance from q_c to the a negative support sample $d_{c'} = \min_{s_{c'} \in S_k} d(q_c, s_{c'})$. $\mathscr{L}_{\text{eps}}$ is defined in terms of these hard-mined distances (τ is a margin):

$$\mathscr{L}_{\text{eps}}(\theta; S_k, Q_k) = \sum_{q_c \in Q_k} \log(1 + \sum_{c' \in \mathbb{C} \setminus \{c\}} \exp(d_{c'} - d_c + \tau)), \tag{4.2}$$

4.3.2 Continual Metric Learning

In continual metric learning, tasks $t \in [1, T]$ arrive sequentially as disjoint datasets D_t. The aim is to learn θ_t incrementally in a *training session* for each task t and to ensure it accumulates knowledge from the previous tasks so as to generalize better to unseen test tasks:

$$\theta_t^* = \arg\min_{\theta_t} \mathscr{L}_{\text{cml}}(\theta_t; D_t). \tag{4.3}$$



(a)

(b)

Figure 4.3: (a) Comparing continual meta-metric learning (DMML-FT [18]) with continual metric learning (BoT-FT [88]). We finetune on 10 equally split Market-1501 tasks. Upper bounds are joint training on all data. (b) Comparison between DwPP and DwoPP (class 1 is the positive class). The old model has never seen class 1 and so likely produces an output less than 1 although we want positive pairs to map to the exact same point in latent space. Also, the dominance of the positive class inhibits distillation of negative pair information.

And the data from previous tasks (i.e. $D_{t'}$ for $t' < t$) are *not* available to the learner at task t.

There are two approaches to defining $\mathscr{L}_{cml}$ in Eq. 4.3: meta-learning methods (DMML [18]) or softmax-based methods (BoT [88]). The majority of Person Re-Identification approaches (including the LReID benchmark [112]) are based on the softmax-triplet loss. We compare these methods in the Continual Metric Learning setting on Market-1501 in Fig. 4.3(a) by simply applying fine-tuning (FT) without any mitigation of forgetting. We clearly see that continual metric learning is quickly surpassed by continual meta-metric learning. The underlying reason for this marked improvement is that re-identification aims to recognize *unseen* objects (each object identity is represented by only one query image at test time). This is the central characteristic of few-shot recognition. Instead, the conventional softmax-triplet loss easily overfits to the current task and neglects the relationships between inter-class samples. The meta learning DMML loss, however, tends to learn a better representation space that generalizes to future unseen tasks and thus suffers less from forgetting. In brief, DMML is a more principled approach for continual metric learning than the softmax-triplet loss and we propose to use DMML as the basis.

4.3.3 Distillation without Positive Pairs (DwoPP)

To adapt the DMML loss defined in Eq. 4.2 to continual metric learning, we compute it for task t over episodes E_k^t drawn only from the current task data D_t. We denote the support set and query set of each episode during task t as S_k^t and Q_k^t. Then the DMML loss is defined with the current model f_{θ_t} as $\mathscr{L}_{\text{eps}}(\theta_t; S_k^t, Q_k^t)$ (see Eq. 4.2).

Episodic meta learning with the DMML loss will not mitigate forgetting in a continual metric learning. Knowledge Distillation [50, 75] is a common technique for alleviating catastrophic forgetting when learning over a sequence of tasks. Note, however, distillation assumes that a classifier over classes from the previous tasks is available on which to perform knowledge distillation – something that for continual meta metric learning we do not have. However, based on the sampled episodes we can construct two temporary classifiers, one based on the previous and one based on the current tasks' feature extractor. We can then define a new distillation loss in terms of these temporary classifiers.

Class Prototypes. To construct the temporary classifier, we compute prototypes as the centroid of embedded samples of each class $\mathbf{u}_c$ (c is class label):

$$\mathbf{u}_c = \frac{1}{n} \sum_{(x_i, y_i) \in S_k^t} f_\theta(x_i) \delta_c(y_i), \tag{4.4}$$

where $\delta_c(y) = 1 \Leftrightarrow y = c$ is an indicator function.

DwPP: Distillation with Positive Pairs. With the class prototypes $\mathbf{u}_c$, the prediction for class $c \in \mathbb{C}$ of query image $\hat{x} \in Q_k^t$ with the model f_{θ_t} is given by:

$$g_c(S_k^t, \hat{x}; \theta) = \frac{[\exp(-d(f_\theta(\hat{x}), \mathbf{u}_c))]^{1/T}}{\sum_{c' \in \mathbb{C}} [\exp(-d(f_\theta(\hat{x}), \mathbf{u}_{c'}))]^{1/T}}, \tag{4.5}$$

where T is the temperature and d is the Euclidean distance. These predictions are used to distill knowledge from task t-1 into task t by constructing two temporary classifiers, one using θ_t and another using θ_{t-1}, and considering all negative and positive pairs:

$$\mathscr{L}_{\text{DwPP}}(\theta_t; \theta_{t-1}, S_k^t, Q_k^t) = \sum_{\hat{x} \in Q_k^t} KL\left[\mathbf{g}(S_k^t, \hat{x}; \theta_{t-1}) \,||\, \mathbf{g}(S_k^t, \hat{x}; \theta_t)\right]. \tag{4.6}$$

Here $\mathbf{g}$ is a classifier constructed by concatenating the predictions g_c defined in Eq. 4.5 for all classes in the episode.

Knowledge distillation for continual meta metric learning requires careful attention to which pairs are included in the distillation loss. Consider the hypothetical case illustrated in Fig. 4.3(b) where we show the predictions of the two temporary classifiers (class 1 is the query class). In task t, the new classes from D_t are not well-discriminated from each other – that is, the margin between positive and negative pairs in D_t is not guaranteed by the model from task $t-1$ and the predicted probabilities are distributed as in the upper left column of Fig. 4.3(b). After learning task t we would like it to be a peaked distribution around the correct class, and simultaneously we also wish to maintain the relative probabilities of all classes (via knowledge distillation). Although this distillation will maintain model stability and mitigate forgetting, the estimate of the old model for the correct label is likely to be unreliable and will prevent the metric loss from pushing similar labels to the same position in the embedding space. Furthermore, the dominance of the positive class prevents distillation of the relevant negative pair information (also known as dark knowledge [50]), which weakens the alignment of classes in the feature space.

In essence, the metric and distillation losses are *antagonistic* due to the inclusion of positive pairs in knowledge distillation. Thus we propose to remove positive pairs from distillation. As shown in the right column of Fig. 4.3(b), since the other classes are negatives for class 1, they can be easily aligned with the previous probabilities to overcome forgetting. At the same time, the peaked distribution in the bottom left of Fig. 4.3(b) can also be achieved by the metric loss. To further analyze the role of positive and negative pairs, we decouple the KL divergence into positive and negative pair distillation as proposed by DKD [177], showing that positive pair distillation leads to performance degradation (see Table 4.5).

DwoPP: Distillation without Positive Pairs. To remove positive pairs from DwPP distillation, we exclude class $\hat{y}$ which is the class label of query image $\hat{x} \in Q_k^t$ from the temporary classifier and rewrite the Eq. 4.5 as:

$$g'_c(S_k^t, \hat{x}, \hat{y}; \theta) = \frac{[\exp(-d(f_\theta(\hat{x}), \mathbf{u}_c))]^{1/T}}{\sum_{c' \in \mathbb{C} \setminus \{\hat{y}\}} [\exp(-d(f_\theta(\hat{x}), \mathbf{u}_{c'}))]^{1/T}} \tag{4.7}$$

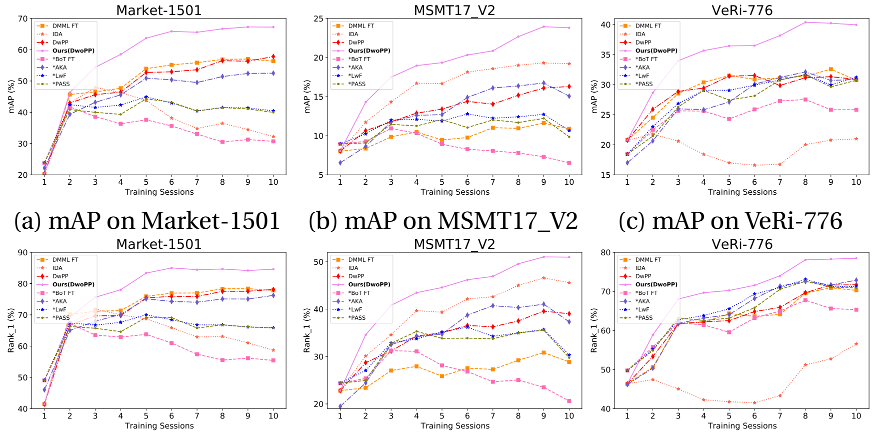
Then the DwoPP distillation can be rewritten as:

$$\mathcal{L}_{\text{DwoPP}}(\theta_t; \theta_{t-1}, S_k^t, Q_k^t) = \sum_{(\hat{x}, \hat{y}) \in Q_k^t} KL[\mathbf{g}'(S_k^t, \hat{x}, \hat{y}; \theta_{t-1}) \,||\, \mathbf{g}'(S_k^t, \hat{x}, \hat{y}; \theta_t)]. \tag{4.8}$$

With the above defined DwoPP distillation loss and episode DMML loss, the continual metric learning loss function for each episode is defined as:

$$\mathcal{L}_{\text{cml}}(\theta_t; \theta_{t-1}, S_k^t, Q_k^t) = \mathcal{L}_{\text{eps}}(\theta_t; S_k^t, Q_k^t) + \lambda \mathcal{L}_{\text{DwoPP}}(\theta_t; \theta_{t-1}, S_k^t, Q_k^t). \tag{4.9}$$

To demonstrate the necessity of removing positive pairs from the distillation, we



(a) mAP on Market-1501 (b) mAP on MSMT17_V2 (c) mAP on VeRi-776

(d) Rank-1 on Market-1501 (e) Rank-1 on MSMT17_V2 (f) Rank-1 on VeRi-776

Figure 4.4: mAP and Rank-1 performance. Methods with "*" use the softmax-triplet loss.

compare DwPP and DwoPP in Sec. 4.4 and perform an ablation on T in both.

4.4 Experimental Results

4.4.1 Experimental setup

Datasets for Intra-domain Object ReID. We propose continual metric learning splits for two Person ReID datasets and one vehicle ReID dataset. **(1) Market-1501 [180]** consists of 32,668 images of 1,501 identities captured by 6 cameras. The dataset is divided into a training set with 12,968 images of 751 identities and a test set containing 3,368 query images and 19,732 gallery images of 750 identities. For continual metric learning setup, we split the 751 training identities into 10 disjoint tasks, each with 75 identities (the first with 76). **(2) MSMT17_V2 [155]** consists of 126,441 images of 4101 persons captured by 15 cameras. Its training set includes 30,248 images of 1041 persons, and its test set covers the remaining 3060 persons with 11,659 query images and 82,161 gallery images. For MSMT17_V2, we split the training persons into 10 tasks also, each task with 104 persons (the first task with 105 persons). **(3) VeRi-776 [83]** contains 49,357 images of 776 vehicles, which are captured by 20 cameras. Among them, 576 vehicles are used for training

		continual metric learning training tasks				Unseen-test task
Task id:		1	2	...	10	
Identities per task:	Market-1501	76	75	...	75	750
	MSMT17_V2	105	104	...	104	3060
	VeRi-776	63	57	...	57	200

Table 4.1: Our proposed 10-task split of two Person ReID datasets and one Vehicle ReID dataset for Continual Metric Learning.

and the remaining 200 are used for testing. In total, VeRi-776 consists of 37,778 training images, 1,678 query images, and 11,579 gallery images. For continual metric learning, we split the training 576 vehicles into 10 tasks, each task with 57 vehicles (the first task with 63 vehicles). Our splits of three dataset are shown in Table. 4.1. For all three datasets, we try to uniformly distribute the training identities into 10 continual metric learning tasks. The query and gallery set are fixed and serve as the unseen-test task.

The Lifelong ReID (LReID) benchmark. We adapt the train set of the inter-domain LReID benchmark by building it from four datasets: Market-1501 [180], CUHK-SYSU ReID [163], MSMT17_V2 [155], and CUHK03 [73]. We removed the DukeMTMC-ReID dataset from the original LReID benchmark [112] due to its retraction on account of privacy issues. Except for this change, we keep the same training order as LReID Order-1: Market-1501 [180]→ CUHK-SYSU [163] → MSMT17_V2 [155] → CUHK03 [73]. After training, the model is evaluated on the test query and gallery sets LReID-Seen of these four datasets (i.e. over *seen* domains). We also test on LReID-Unseen test set which combines seven person ReID datasets: VIPeR [40], PRID [51], GRID [86], i-LIDS [181], CUHK01 [72], CUHK02 [71], and SenseReID [178].

Implementation details. We follow the same network structure and training strategy as DMML [18] for methods based on the DMML loss, and use the network and training protocol of BoT [88] for methods based on the softmax-triplet. For our person ReID experiments, we use ResNet-50 [46] pretrained on ImageNet [122] as our feature extractor. The last spatial downsampling operation in the network is removed to maintain high resolution. We resize input images to 256×128 for all methods. For vehicle ReID, we also use a ResNet-50 backbone pretrained on ImageNet as the embedding architecture, and use input images of size 224×224 augmented with random horizontal flips. We use the Adam optimizer [63] with a base learning rate of $LR = 0.0002$ and weight decay of 0.0001. All models are trained for 600 epochs with fixed learning rate of 0.0002 for the first 300 epochs, after which the learning rate is reduced by a factor of $0.005^{1/300}$ each epoch until the end. We set the trade-off coefficient to $\lambda = 1.0$, the margin as $\tau = 0.4$ as in DMML [18], and the temperature to $T = 1.0$ for DwoPP and $T = 10.0$ for DwPP. The number of classes, support images, and query images are in each episode are $N = 32, n_s = 5, n_q = 1$, respectively.

Compared methods and metrics. Our evaluation is divided into two parts: **(1)** To compare with conventional continual learning methods, we train models with the softmax-triplet loss of BoT [88]. For methods using this loss without exemplars, we selected AKA [112], PASS [182], and LwF [75]. For methods using exemplars, we selected FT+, iCaRL [119], LUCIR [54], and LwF+ [75]. **(2)** For comparison with incremental meta learning methods, we build upon the DMML loss [18]. For methods without exemplars we selected IDA [79]. For methods with exemplars, we selected ERD [150]. Note that AKA is the state-of-the-art in LifelongReID and IDA is the state-of-the-art in incremental meta learning. For all exemplar-based methods we store 500 exemplars for all experiments. We use mean Average Precision (mAP) and Accuracy at Rank-1 as metrics [166]. We compute the mAP and Rank-1 Accuracy of the model on the unseen test set after each task. All results are averages over three runs.

4.4.2 Comparative performance evaluation

Intra-domain Lifelong Object ReID. Fig. 4.4 gives the mAP and Rank-1 curves on Market-1501, MSMT17_V2, and VeRI-776. We report the performance of all methods after task $t = 10$ and the average metrics over all training sessions in Table 4.2. On all three datasets, finetuning with softmax-triplet loss is always sub-optimal to the finetuning with the meta metric loss. The performance gap between the mAP for the DMML-FT and BoT-FT after the last task is 25.8, 4.3, and 5.0 on three datasets, respectively. Note that the two losses result in a similar joint training performance. This demonstrates that meta metric learning is more suitable to the Continual Metric Learning problem, as we discussed in Sec. 4.3.2. For continual learning methods without exemplars, our method DwoPP performs best on all datasets. Compared to the DMML-FT metrics after task 10, DwoPP improves by between 9.1 to 12.9 in mAP. Note that on Market-1501 and VeRi-776 DMML-FT outperforms most of the methods that actively counter fogetting. Furthermore, we also include a comparison with rehearsal methods in Table 4.2. The methods iCaRL, LwF+, FT+ and ERD obtain similar results, and improved performance compared to DMML-FT. LUCIR performs the worst and that could be because the cosine embedding is used for re-balancing the old and new tasks. Our exemplar-free method DwoPP performs better than exemplar-based methods on these three datasets (only marginally worse in average Rank-1 Accuracy on VeRi-776). We also ablate our distillation and report results for distillation with all pairs (DwPP). The results of DwPP show that naive

Metric:	mAP						Rank-1 Accuracy					
Dataset:	Market		MSMT17		VeRi-776		Market		MSMT17		VeRi-776	
Based on episodic optimization with DMML loss [18]												
Joint training:	82.2		44.7		73.6		92.6		68.9		92.1	
Sessions:	last	avg	last	avg	last	avg	last	avg	last	avg	last	avg
without exemplars												
DMML-FT (ICCV'19)	56.3	49.1	10.9	10.0	30.8	29.3	77.8	71.5	28.9	27.4	70.3	62.4
IDA (ECCV'20)	32.2	37.8	19.2	16.8	21.0	18.4	58.7	63.1	45.6	38.2	56.6	45.4
DwPP	57.8	48.4	16.3	13.3	30.9	28.9	78.1	70.7	39.0	33.9	71.7	63.3
Ours (DwoPP)	**67.2**	**57.6**	**23.8**	**19.1**	**39.9**	**35.3**	**84.6**	**77.1**	**51.0**	**42.6**	**78.5**	**69.3**
with 500 exemplars in total												
ERD (CVPRW'22)	63.5	53.9	21.7	17.2	38.2	33.8	81.8	74.5	46.6	39.4	72.9	65.5
Based on global optimization with softmax-triplet loss from BoT [88]												
Joint training:	82.4		43.2		69.2		93.0		71.1		92.7	
Sessions:	last	avg	last	avg	last	avg	last	avg	last	avg	last	avg
without exemplars												
BoT-FT (CVPR'19)	30.7	33.5	6.6	8.4	25.8	24.9	55.4	58.8	20.6	25.4	65.3	62.1
LwF (ECCV'16)	40.5	40.2	10.7	11.8	31.2	28.1	65.9	65.4	30.3	32.3	71.3	65.8
PASS (CVPR'21)	40.0	40.1	9.9	11.6	30.7	27.3	65.8	64.2	29.7	31.9	70.9	64.4
AKA (CVPR'21)	52.5	45.6	15.1	13.3	30.9	27.1	76.2	69.9	37.3	34.6	72.9	64.4
with 500 exemplars in total												
BoT-FT+ (CVPR'19)	61.5	52.4	21.5	17.5	36.7	32.3	81.0	74.4	47.7	41.3	76.2	69.6
iCaRL (CVPR'17)	58.0	52.2	21.6	18.3	38.0	33.3	78.7	74.5	47.5	42.2	78.1	70.9
LwF+ (ECCV'16)	60.7	54.0	20.8	17.5	38.3	33.3	80.3	75.4	46.6	40.8	77.9	70.1
LUCIR (CVPR'19)	8.3	6.8	3.0	2.5	10.1	11.6	27.7	22.3	10.6	9.1	42.9	41.4

Table 4.2: Results in mAP and Rank-1 Accuracy (in %) after last task and average over all tasks. The top half reports results for meta metric learning, and the lower half for global optimization methods using the softmax-triplet loss (BoT [88]). Results are further split into methods with and without exemplars. The best exemplar-free results are highlighted in **bold**.

application of knowledge distillation to continual meta metric learning does hardly improve results. The removal of positive pairs (DwoPP) results in large performance gains after the last task: gains between 7.5 to 9.4 in mAP. To further analyze the results we measure forgetting and plasticity on the Market 1501 dataset. Continual learning aims to counter forgetting (stability) while optimally learning new tasks (plasticity). To measure these, we track the change in mAP for each identity in the unseen test set after each task: a drop is added to forgetting, an increase to plasticity. In Table 4.3 we report the plasticity and forgetting averaged over tasks. We see that DwoPP has greatly reduced forgetting at the price of only a small decrease in plasticity.

Inter-Domain Lifelong Person ReID (LReID). In Table 4.4, we compare DwoPP with other methods on the LReID [112] benchmark. Similar to the results for the intra-domain ReID setting, the DMML-FT baseline outperforms BoT-FT by a large

Continual learning metrics on Market 1501							
	BoT FT	LwF	AKA	IDA	DMML FT	DwPP	DwoPP
Plasticity	9.7	9.5	9.7	7.2	10.8	10.7	8.1
Forgetting	-8.9	-7.7	-6.5	-5.9	-6.8	-6.5	-2.9
Overall	0.8	1.8	3.2	1.3	4.0	4.2	5.2

Table 4.3: Average forgetting and plasticity in mAP (%) on Market-1501 together with the overall mAP change (defined as plasticity plus forgetting).

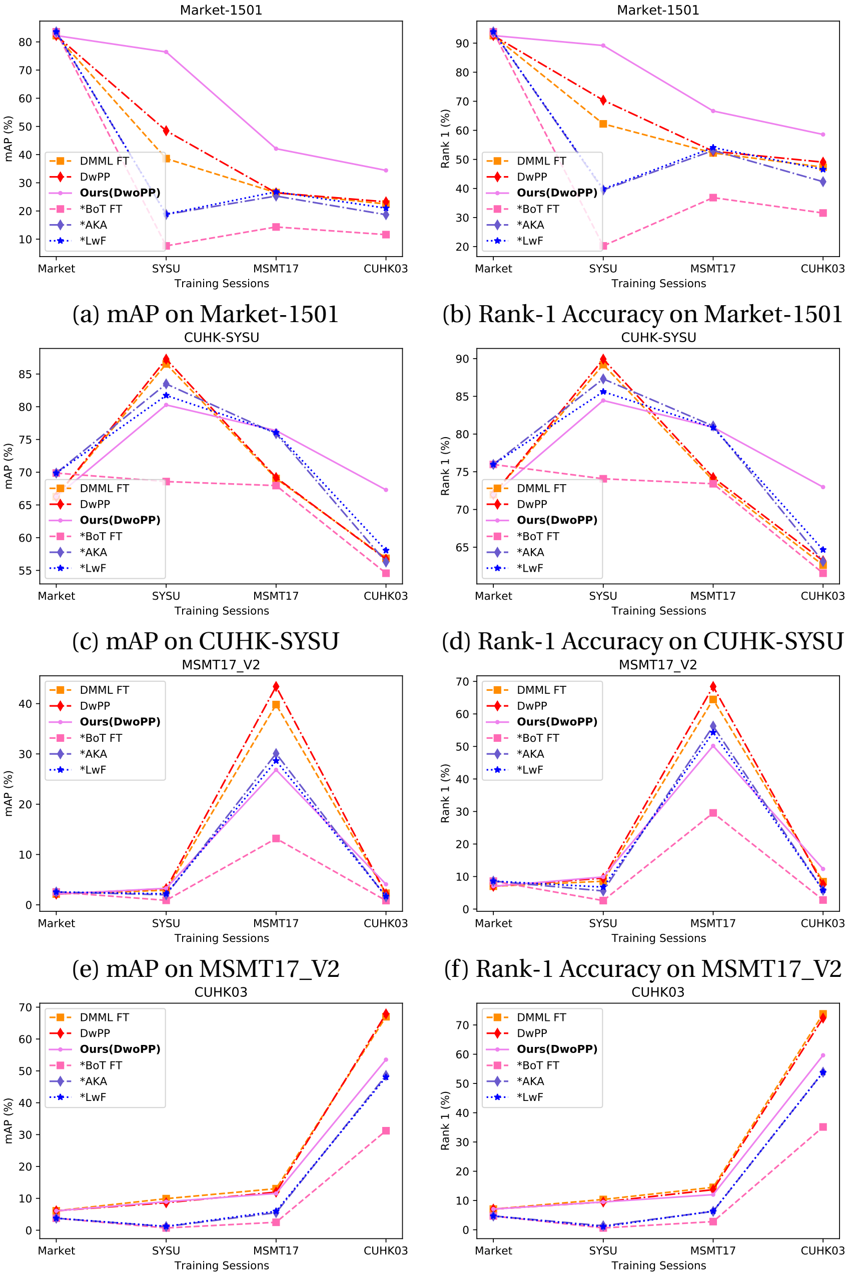
	mAP						**Rank-1 Accuracy**					
	market	sysu	msmt17	cuhk03	***seen avg.***	***unseen***	market	sysu	msmt17	cuhk03	***seen avg.***	***unseen***
BoT-FT	11.6	54.6	0.8	31.2	24.6	32.4	31.6	61.6	2.8	35.1	32.8	32.8
LwF	21.0	58.0	1.7	48.0	32.2	43.3	46.5	64.7	5.8	53.8	42.7	42.9
AKA	18.7	56.3	1.6	48.6	31.3	43.6	42.3	63.1	5.8	53.9	41.3	43.6
DMML-FT	22.5	56.8	2.3	67.0	37.2	42.8	47.3	62.6	8.4	**73.8**	48.0	42.6
DwPP	23.2	56.7	2.2	**67.9**	37.5	44.7	49.1	63.2	7.5	72.4	48.0	44.2
Ours (DwoPP)	**34.4**	**67.3**	**4.1**	53.5	**39.8**	**48.5**	**58.6**	**73.0**	**12.3**	59.6	**50.9**	**47.8**

Table 4.4: Results after learning the last task. BoT [88] (above) and DMML [18] (below).

margin for both seen and unseen tasks. Our method performs best, outperforming AKA by 8.5/9.6 (mAP/Rank-1 Accuracy) on seen tasks and 4.9/4.2 (mAP/Rank-1 Accuracy) on unseen tasks. The difference between DwPP and DwoPP on LReID further highlights the importance of removing positive pairs from knowledge distillation. An interesting phenomenon we observed is that DwPP is always better in the current task evaluation. We assume this is because DwPP forces the predictions to be aligned with the probability distributions of the old model, which contain some information about the relative distances of these identities. This extra information further enhances representation learning in the current task, thus leading to better performance on the current task even compared to the finetuning baseline (which is usually better on the current task). All the performance curves on LReID-Seen are shown in Fig. 4.5 and the curves on LReID-Unseen are shown in Fig. 4.6.

Influence of positive pairs on distillation. To better understand the role of positive pairs (PP) and negative pairs (NP) in knowledge distillation, we decouple the knowledge distillation (following DKD [177]) from Eq. 4.6 into PPKD and NPKD by $\mathscr{L}_{\text{DwPP}^*} = \alpha * PPKD + \beta * NPKD, \alpha + \beta = 1.0$ (here we use $T = 1.0$). Note that $\mathscr{L}_{\text{DwPP}} = PPKD + \rho * NPKD$ (see Supplementary Material for further explanations). In Table 4.5, we observe that the performance drastically decreases with higher participation of positive pairs.

Ablation on λ in DwoPP and temperature T in both DwoPP and DwPP. In Fig. 4.7(a) we vary λ which controls the tradeoff between metric and distillation



(a) mAP on Market-1501

(b) Rank-1 Accuracy on Market-1501

(c) mAP on CUHK-SYSU

(d) Rank-1 Accuracy on CUHK-SYSU

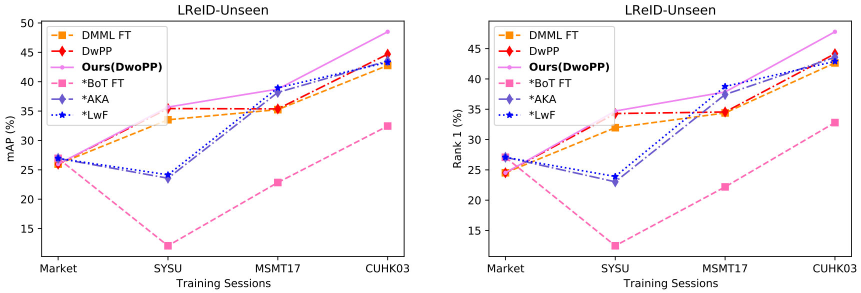
(e) mAP on MSMT17_V2

(f) Rank-1 Accuracy on MSMT17_V2

(g) mAP on CUHK03

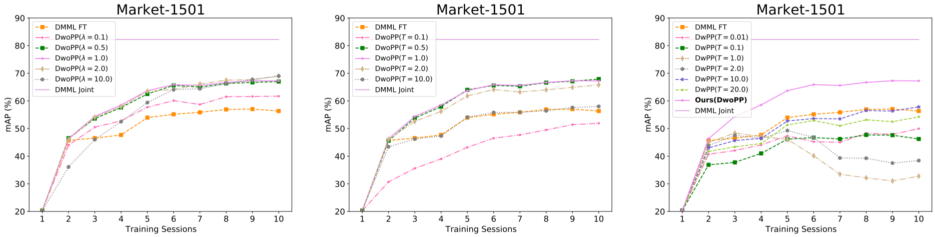
(h) Rank-1 Accuracy on CUHK03

Figure 4.5: Results in mAP and Rank-1 Accuracy on the LReID benchmark. The training order is (Market-1501→ CUHK-SYSU → MSMT17_V2 → CUHK03). The first four rows show the evaluation on these four tasks respectively.



(a) mAP on LReID-Unseen (b) Rank-1 Accuracy on LReID-Unseen

Figure 4.6: Results in mAP and Rank-1 Accuracy the LReID-Unseen test set of the LReID benchmark. The training order is (Market-1501→ CUHK-SYSU → MSMT17_V2 → CUHK03).



(a) λ for DwoPP (b) T for DwoPP (c) T for DwPP

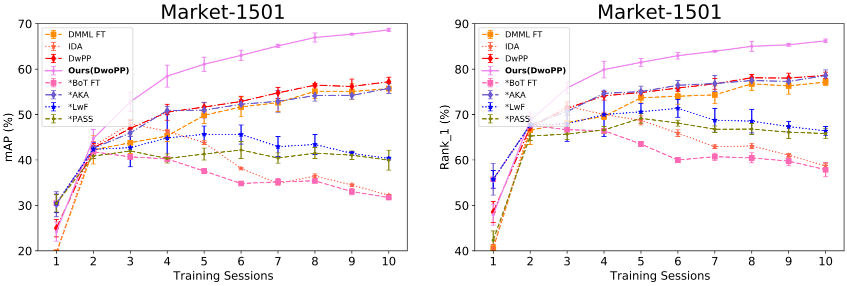
Figure 4.7: Ablation study on hyperparameters λ and T.

losses. Except for $\lambda = 10.0$ and $\lambda = 0.1$, DwoPP performance is stable to changing λ. We set $\lambda = 1.0$ for DwoPP in all experiments. In Fig. 4.7(b), we vary the temperature hyperparameter T in DwoPP. A high temperature smooths the distribution and decreases the influence of the dominant class. For DwoPP $T = 10.0$ performs similarly to finetuning, and $T = 0.1$ causes the model to focus only on the highest probability. Thus we set $T = 1.0$ for DwoPP. In Fig. 4.7(c) we vary the temperature T in DwPP to determine if larger temperatures benefit it. However, even with the best $T = 10.0$, DwPP performs similarly to DMML-FT and much worse than DwoPP. Again showing that naive knowledge distillation does not improve results for continual meta metric learning. We use $T = 10$ for DwPP in all experiments.

More random orders on Market-1501 dataset. In previous experiments, we split

		DwoPP	DKD [177]				DwPP
	α	0.0	0.1	0.3	0.5	1.0	1.0
	β	1.0	0.9	0.7	0.5	0.0	1-ρ
mAP	last	**67.2**	62.9	48.2	36.0	25.9	32.8
	avg	**57.6**	53.7	46.8	39.1	32.1	37.8

Table 4.5: Decoupling Eq. 4.6 into PPKD and NPKD with coefficients α and β on Market-1501 with temperature $T = 1.0$. ρ is the positive probabilities as in DKD [177].



(a) mAP on Market-1501

(b) Rank-1 on Market-1501

Figure 4.8: Performance on Market-1501 averaged over three random ID orders with standard deviation.

tasks according to the *object IDs*. Thus, for the purpose of verifying the robustness of our proposed method to various orderings of the tasks, we randomly generate three different orderings of person IDs from Market-1501 to split the tasks (see Fig. 4.8). The results show that the trends are similar as those reported in Table 1. Results are averages and standard deviations in mAP and Rank-1 Accuracy over these three runs.

4.5 Conclusions

We observed that meta learning approaches perform better than those based on global metric loss optimization for Object ReID, and thus based our approach on Continual Meta Metric Learning. To overcome forgetting, we proposed Distillation without Positive Pairs (DwoPP) as an approach that eliminates positive samples from distillation. This distillation makes the metric learning model accumulate knowledge from the previous and current task, and generalizes better to unseen tasks. Extensive experiments on newly proposed intra-task object re-identification datasets and the existing LReID benchmark demonstrate the effectiveness of our approach. Furthermore, the experiments confirm that naive knowledge distillation does not improve results for continual meta metric learning, and only after the removal of positive pairs forgetting of previous tasks is efficiently countered.

Limitations and ethical considerations Person ReID is fraught with ethical concerns over its potential to violate the privacy of observed subjects. Although continual learning for Person ReID offers the possibility of learning and updating models without the need for long-term retention of sensitive data, it also runs the risk of "baking" biases into the model that, due to mitigation of forgetting, become difficult to remove. For real applications there is still a large gap between joint and continual training for object ReID, and a limitation of the experiments in this work is the relatively short task sequences we consider.

5 Density map distillation for incremental object counting

5.1 Introduction

The image-based counting task aims to infer the number of people, vehicles or any other objects present in images. It has a wide range of applications such as traffic control, environment survey and public safety. Most of existing research focus on learning a model from a single dataset. Only [16] and [89] propose to train a model on multiple datasets simultaneously in a multi-task setting. In this paper, we propose a method to incrementally learn to count new objects or to count in a new domain.

Continual learning (CL) addresses the problem of training a model from a non-stationary distribution. It is important because the data in the real world does not come together, and often the previous data cannot be revisited due to the privacy or storage restrictions. Researchers have explored continual learning in many tasks, e.g. classification [66, 75], segmentation [13, 32], and object detection [131]. However, continual learning for counting systems, has to the best of our knowledge, not yet been studied.

One of the main challenges of continual learning is catastrophic forgetting. After training on new data, models tend to forget the knowledge from the previous data. In the past few years, people tried to alleviate this issue by using replay examples [119, 161], expand networks [167] and regularization [66, 75]. As one of the most promising methods, regularization can be further categorized as weight regularization [66] and data regularization [75]. The former apply regularization on weight to prevent them from drifting too far from the old model, while the later apply it on the output of the network with given input data. Due to their success for classification tasks, the fact that they do not require exemplars, and because they scale well with the number of tasks, we will here explore data regularization for object counting.

However, these methods are mainly designed for the classification problem, which aims to predict the category for a given sample. For the counting problem, which is a regression problem where the output is a scalar value, directly applying any existing CL method is suboptimal. We therefore propose a new method called Density Map Distillation (DMD). For each new object, we train a separate counter head that maps the feature extraction backbone to an object-specific density map. After the training of each task, the counter head is fixed and only the feature extractor is trained during future tasks. When training the new task, we use the new data to apply a distillation on the output of all previous counter heads. Since the counter head is fixed and the feature extractor is drifting, we propose to train an adaptor to project the new features to the old features. This mechanism allows us to keep plasticity while maintaining stability (i.e., prevent forgetting).

The contribution of this chapter include: (1) We set up a benchmark of incremental learning for counting new objects. We define metrics for evaluating incremental counting problems. (2) We propose Density Map Distillation (DMD) for the incremental counting problem. The novelty includes fixing the task-specific counter head and training an adaptor for the feature extractor. Our method prevents forgetting, while maintaining plasticity to learn new tasks. (3) We adapt several existing methods of incremental learning in our benchmark. Experiments shows that our new methods outperforms these existing methods.

5.2 Related Work

5.2.1 Incremental Learning

Incremental learning aims to develop methods that can learn new knowledge from new data while not forgetting previous knowledge learned from the previous training stages. The existing methods can mainly be categorized as three types: distillation based, dynamic model based and rehearsal based [26]. Distillation based methods focus on how to limit the change of the model by applying a loss on the weights directly [3, 66], or on the output features [54] and probabilities [75]. Dynamic model based methods [167] extend the architecture of the network to learn new knowledge from the new incoming data distribution. Rehearsal based methods [119] save a few exemplars from the previous dataset and replay them or use them to constrain the model during the new training sessions. For a more complete overview of incremental learning literature, we refer to Chapter 2.

Previously, incremental learning mainly focused on image classification problems. Recently, the community also developed incremental learning algorithms for other problems such as image generation [160], segmentation [13], object detection [113], video classification [109]. But to the best of our knowledge, there is no work for incremental learning of counting problems yet.

5.2.2 Crowd Counting

There are two categories of crowd counting methods, density map based methods [90, 147] and localization based methods [138].

Localization based methods count by locating each individual's position. Some methods [76, 125] are driven by an object detector. However, since most of the counting datasets have only point annotations available, inaccuracy is introduced by estimation of the ground truth bounding boxes. Liu et al. [77] propose Recurrent Attentive Zooming Network(RAZ-Net) that recurrently detect high density regions and zoom in for re-inspection. The network performs the counting and localization task at the same time, and they define an adaptive fusion scheme to make this two tasks complement each other. Song et al. [138] propose the Point to Point Network (P2PNet) that predicts the localization points directly. They surpass the state-of-the-art by using a new one-to-one matching strategy from the prediction to the ground truth based on the Hungarian algorithm.

For the localization based method, it is hard to predict each location where the crowd density is very high [146]. Most of the research in counting mainly focuses on predicting a density map and then count by the summing it. In [126, 175], they propose to use several parallel CNNs of different sizes to address the problem of scale variation. Another line of research focuses on the loss function. In [90], Ma et al. propose a Bayesian loss to measure the distance between the predicted and the ground truth density map. They construct a smoothed density map for each of the annotated points, where the value is the posterior probability of this point at the corresponding position. Wang et al. [147] measure the similarity between the predicted density map to the ground truth density map by solving an Optimal Transport (OT) problem.

In the above methods, the model is always trained with one dataset. In [16] and [89], they propose to train a model on multiple datasets simultaneously. Ma et al. [89] deal with the issue that the model is sensitive to scale shift. They divide each image into non-overlapping patches and apply scale alignment. They derive a closed-form solution of the optimal image rescaling factor given the scale distribution. A CNN network is trained to predict the spatial distribution and the scale distribution. In [16], Chen et al. deal with the problem that the model tends to focus on learning the dominant domains and ignores the non-dominant domains. They propose Variational Attention (VA) to model the domain specific attention distribution. In addition, they also propose Intrisic Variational Attention (InVA) for the concern of domain overlapp across different datasets and sub-domains within a single dataset. Both Va and InVA refine the propagating knowledge so that the data from each dataset can be learned without bias.

There are also some research [35, 44, 152] focusing on counting problem in domain adaptation setting, where the model is trained on the source dataset and the label of the target dataset is limited. Wang et al. [152] consider using synthetic dataset as a source set and adapt the model to real dataset. They propose using CycleGAN [183] for domain adaptation. In [35,44], they use adversarial training for the domain adaption [33], where a discriminator is trained to classify the feature to source or target and an adversarial loss is applied on the feature extractor to fool the discriminator.

Previous works in counting focus on one category. In [87], Lu et al. propose class-agnostic counting. The aim is to train a network that counts the number of instances in an image by specifying an exemplar patch. They propose Generic Matching Network (GMN) which is pretrained on video data for tracking, and they count instances by matching the instance on the test images. In [117], Ranjan et al. proposed the Few-shot Adaptation and Matching Network (FamNet). It contains a density prediction module and a multi-scale feature extraction module, which is a fixed pretrained ResNet-50 [46]. They compute the correlation maps of the features between the images and the exemplars at different scales. The density prediction layer uses these outputs to predict a final density map. They also propose a test-time adaptation loss, consisting of a min-count loss and a perturbation loss. Instead of a fixed similarity measure, Shi et al. [129] propose a trainable bilinear similarity metric. Furthermore, they also extend it to a dynamic similarity metric that captures the key pattern for each few-shot exemplar specifically.

5.3 Method

Counting is an integral part of many real-life applications. To alleviate the human costs of manual counting, many methods have been developed for the counting of objects [90,138,147]. As discussed in the introduction, these methods generally assume that all training that is jointly available. However, for many applications this assumption is not realistic and the algorithm would only be able to have access to a batch of data at each time step.

A naive approach to learning from a sequence of tasks would be to just continue finetuning the model on the available data of consecutive tasks. However, this would lead to the *catastrophic forgetting* phenomenon. An illustration of this is provided in Figure 5.7 where we show that after learning several tasks with finetuning, the method has lost its ability to count the first-task *grapes* class. In this section, we explore distillation-based methods for incremental learning of object counting to prevent the effect of catastrophic forgetting.

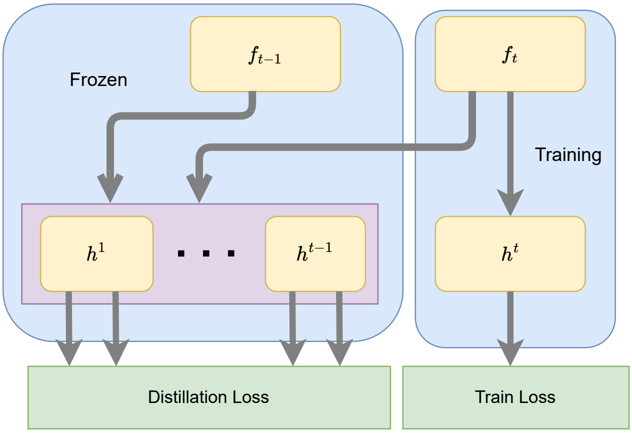


Figure 5.1: Density Map Distillation (DMD) without Adaptor. While training new tasks the distillation loss is applied on the output density map using the previous counter heads, between the previous and the new feature extractors. Different from LwF [75], previous counter heads are fixed when training new task.

5.3.1 Notation

In a typical counting problem, images X_i are annotated for a single object class $c \in C$, for example annotations of persons, cars, or apples are given. Existing works do not consider counting various classes of objects simultaneously. Typically, objects are annotated with a single point in the center of the object at positions $p_{ij}, j \in 1, ..., N_i$ where there are N_i objects for the image x_i. We will use the notation P_i to refer to the set of locations in image x_i. Object counting learns a model for a single object class that given an input image maps to a density map which predicts the number of object instances per pixel [90, 147] or which directly predicts the object coordinates [138].

In incremental learning for counting problems, the data is spli in various tasks, where each task $t \in [1, T]$ arrives sequentially. For each task, the dataset $D_t = \{c_t, \big((x_1, P_1), (x_2, P_2), \cdots, (x_M, P_M)\big)\}$ contains the class category c_t and images with ground truth position annotations. We consider the scenarios where each task has a single object category c_t different from the other tasks. After training on all T tasks, the model is evaluated on a test set Y that contains images of all objects C seen in the various tasks. The task-ID of the test images is available to the algorithm at inference time (this setting is also known as task-incremental learning).

For the training of the object counting network, we propose to use a network which can be divided into a feature extractor $f : R^{w \times h \times 3} \rightarrow R^{w_d \times h_d \times d}$ where d is the number of output channels of the feature extractor, and an object-specific *counter head* given by $h : R^{w_d \times h_d \times d} \rightarrow R^{w_d \times h_d \times 1}$. The counter head maps from the feature space to a density map. The prediction of a network for an image x is then given by:

$$\hat{y} = \sum_{w=1}^{w_d} \sum_{h=1}^{h_d} \hat{d}(x) = \sum_{w=1}^{w_d} \sum_{h=1}^{h_d} h \circ f(x) \tag{5.1}$$

where $\hat{d} = h \circ f$ is the predicted density map and the summation is over the spatial coordinates of the density map.

For training the new task, we use the loss proposed by Wang et al. [147]

$$\mathscr{L}_{\text{train}} = \left| \|d\|_1 - \left\|\hat{d}\right\|_1 \right| - \lambda_1 \mathscr{W}\left(\frac{d}{\|d\|_1} - \frac{\hat{d}}{\left\|\hat{d}\right\|_1} \right) + \lambda_2 \frac{1}{2} \left\| \frac{d}{\|d\|_1} - \frac{\hat{d}}{\left\|\hat{d}\right\|_1} \right\| \tag{5.2}$$

The first term is the counting loss for the final counting number. The second term is the optimal transport loss, where $\mathscr{W}$ is the Monge-Kantorovich's Optimal Transport (OT) cost [145]. The third term is the Total Variation (TV) loss, and λ_1 and λ_2 are the hyperparameters for the OT and TV losses.

To extend the above described method to incremental object counting, we use the following notations. The network contains a feature extractor after learning task t given by f_t. For each of the learned tasks, we have a task specific counter head h_t for each object. At the beginning of the task, the feature extractor f_t is initialized from the previous feature extractor f_{t-1}. The previous feature extractor f_{t-1} is then fixed and stored as a reference. Other older feature extractors like f_{t-2} are not kept.

When training task t we use h_t^τ to refer to the previous counter heads for the object that was learned at task τ. At inference time, we combine the last feature extractor with any of the previously learned counter heads, so for example to get the solution for class c_τ after training task t we apply $h_t^\tau \circ f_t$. We also consider fixing the previous task specific counter, i.e. we do not update it when learning new tasks, so $h_\tau^\tau = h_{\tau+1}^\tau = \cdots$, and we simply refer to it as h^τ.

5.3.2 Data regularization for regression problems

One of the popular approaches to prevent *catastrophic forgetting* in continual learning is by means of regularization methods [26]. Compared with the other two main approaches to continual learning, regularization methods have the advantage over rehearsal methods that they do not require the storage of any data from previous tasks, and they do not have an increased memory footprint when training on larger task sequences like isolation methods typically have. Regularization methods can

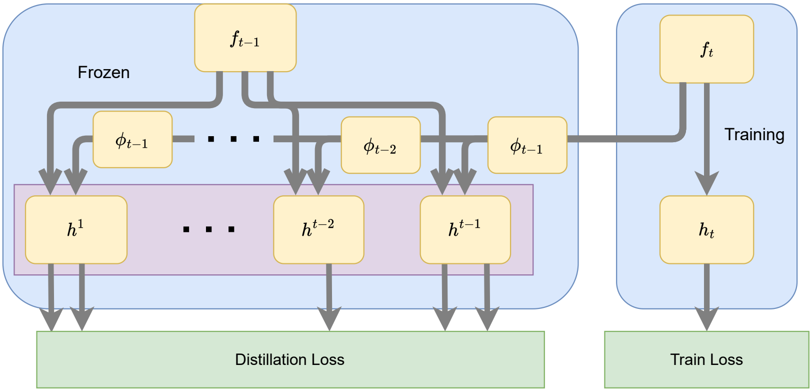


Figure 5.2: Density Map Distillation (DMD). In addition to the distillation loss, we train an adaptors (ϕ) to project the new features to the old features, since the previous counter heads are fixed and the feature extractor is still training.

be differentiated in data and parameter regularization methods.

Data regularization for classification networks is proposed by [75] and it is one of the most popular methods for exemplar-free continual learning. Different from the parameter regularization methods [3, 66] which apply the regularization loss on the parameters of the network, data regularization apply the regularization on the output of network layers. Other than parameter regularization, it is dependent upon the data on which the distillation is applied. This idea has been further extended by [27, 54]. The former apply the regularization loss on the feature output and the output after the cosine normalization. The latter apply them on several intermediate layers and study various marginalization to improve the plasticity of the method.

However, the most common data regularization methods, LwF [75] cannot be applied directly to the counting problem. In LwF [75], Li et al. proposed to apply a knowledge distillation loss between the new and the old output. Given the image from the new dataset as the input, both models give a prediction of the probability and a cross entropy loss is applied as a regularization. However, an object counting network does not output a probability, and therefore the cross entropy loss cannot be applied. An adaption to the counting problem is to apply a L_2 loss on the density map:

$$\mathscr{L}_{\text{reg}} = \sum_{\tau \in [1,t-1]} \left\| h_t^\tau \circ f_t(x) - h_{t-1}^\tau \circ f_{t-1}(x) \right\|_2 . \tag{5.3}$$

We will identify this method with Learning without Forgetting (LwF) in our results section. However, we found that such an adaptation leads to suboptimal results. We hypothesize that this method suffers from overfitting.

Another typical data regularization method is to apply regularization on the feature level [27,54] according to:

$$\mathcal{L}_{\text{reg}} = \sum_{\tau \in [1, t-1]} \left\| f_t(x) - f_{t-1}(x) \right\|_2 . \tag{5.4}$$

We call this method Feature Distillation (FD). This prevents the feature extractor from drifting too far from the old one. But this regularization does not consider the difference between the new and old task. So it is either too rigid so that the model cannot learn from new tasks or too flexible and the model forgets previous knowledge. This was also observed by PODNet [27].

5.3.3 Density Map Regularization with Cross-Task adaptors

To address the shortcomings of data regularization for regression tasks, and to prevent the overfitting of the previous counter heads, we propose a further adaptation. After training of each task, the counter head for this task will be fixed. So the notation h_t^τ (the counter head for task τ during or after the learning of the task t) can be simplified as h^τ, because the counter head is not changed after the training, and hence $h_\tau^\tau = h_{\tau+1}^\tau = \cdots$. We also store the previous feature extractor f_{t-1} as a reference for the regularization loss. Earlier feature extractor are not needed, so the memory requirement does not scale linearly with the number of tasks. Then we apply the following regularization loss on the density map output from the old and new models:

$$\mathcal{L}_{\text{reg}} = \sum_{\tau \in [1, t-1]} \left\| h^\tau \circ f_t(x) - h^\tau \circ f_{t-1}(x) \right\|_2 . \tag{5.5}$$

This method is an exemplar-free method, since images from previous tasks are not used. As shown in Figure 5.1, both old and new feature extractors use the same image $x \in D_t$ as input and extract a feature $f_{t-1}(x)$ and $f_t(x)$. We use L_2 distance as the regularization loss. It is applied to the output of each counter head for all previous tasks $h_1, \cdots, h_{t-1}$, which encourages the new model output to yield the same result when counting previous objects.

As we fixed the previous counter head, this might prevent the feature extractor from learning new knowledge. Therefore, in addition, we propose to train an adaptor ϕ to project the features from the new feature extractor to the old one. The adaptor is trained together with the feature extractor using the distillation loss. As illustrated in Figure 5.2, when training task t, the adaptor ϕ_{t-1} projects the features generated by f_t to approximate those by f_{t-1}. Similarly, by cascading several previous adaptors $\phi_{t-2}, \cdots, \phi_1$, the features can be projected to those in earlier stages. So the distillation loss with adaptor is given by:

$$\mathscr{L}_{\text{reg}} = \sum_{\tau \in [1,t-1]} \left\| h^\tau \circ \phi_\tau \circ \cdots \circ \phi_{t-1} \circ f_t(x) - h^\tau \circ \phi_\tau \circ \cdots \circ \phi_{t-2} \circ f_{t-1}(x) \right\|_2. \quad (5.6)$$

We call our method *density map distillation* (DMD). To identify, the version defined by Eq. 5.5 without the adaptor, we will use the name *DMD w/o Adapt*). The learning of adaptors between backbone networks in continual learning has been studied recently for continual self-supervised learning [31,37]. However, its usage in combination with supervised heads, as is done in this chapter, has not been studied before.

$$\mathscr{L} = \mathscr{L}_{\text{train}} + \lambda \mathscr{L}_{\text{reg}}, \quad (5.7)$$

where λ is the hyperparameter to balance the training loss and the regularization loss.

During the inference, the feature is extract by the new feature extractor f_t. To count the object c_τ, the feature needs to be adapted through all the adaptors learned after that task, $\phi_{t-1}, \phi_{t-2}, \cdots, \phi_\tau$. Then counter head h^τ uses the adapted feature to predict the density map for the given object according to:

$$\hat{d}(x) = h^\tau \circ \phi_\tau \circ \cdots \circ \phi_{t-1} \circ f_t(x). \quad (5.8)$$

5.4 Experimental Results

In this section, we introduce the experimental setup and evaluate the proposed method on several benchmark counting datasets.

5.4.1 Dataset and evaluation

Datasets. The RSOC dataset is a counting dataset of aerial images proposed by [34] involving *buildings, small vehicles, large vehicles,* and *ships.* In this paper, we will consider learning to count these classes incrementally in the before mentioned order. The images of buildings are collected from Google Earth, while the rest are from the DOTA dataset [162]. The DOTA dataset is an object detection dataset of aerial images. The original labels of bounding boxes are replaced by their central location for the counting problem. There are 2468 images for buildings, 280 images

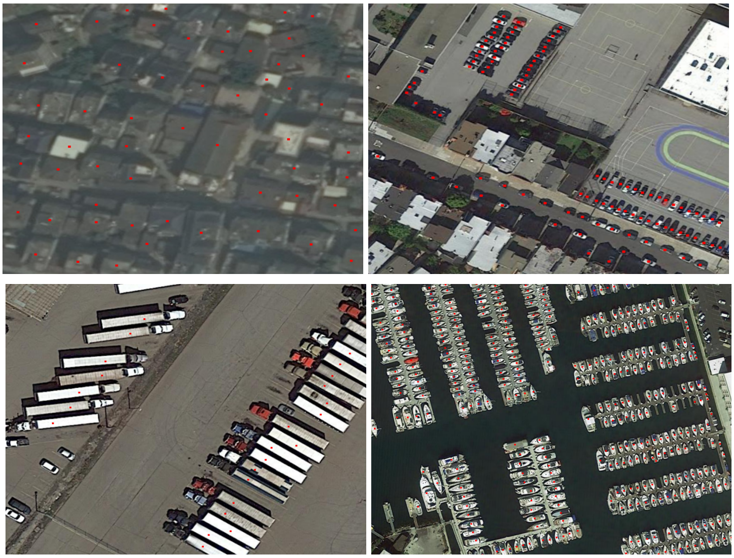


Figure 5.3: Sample images from RSOC dataset.

for small vehicles, 172 images for large vehicles and 137 images for ships

The FSC147 [117] dataset is a counting dataset for few-shot learning, containing 147 categories. For most categories, there are less than 100 images per category. For our incremental learning, we chose several categories that contain a significant number of images. To better share the knowledge across the learning process, we select similar categories for the learning sequence. We consider two sequences of counting of four tasks. The first sequence, called *FSC-fruits*, contains *grapes*, *tomatoes*, *strawberries* and *apples*, containing 116, 117, 126 and 165 images, respectively. The second sequence, called *FSC-birds*, is *flamingos*, *pigeons*, *cranes* and *geese*, containing 76, 81, 108 and 162 images, respectively.

Evaluation Metric. Following previous methods, we use rooted Mean Squared Error (MSE), Mean Absolute Errors (MAE) and mean Normalized Absolute Errors (NAE) as metric to evaluate the performance of the model.

Mean Squared Error (MSE) is defined as:

$$\mathrm{MSE} = \frac{1}{N}\sum_{i=1}^{N} \left\| \hat{y} - y \right\|_2, \tag{5.9}$$

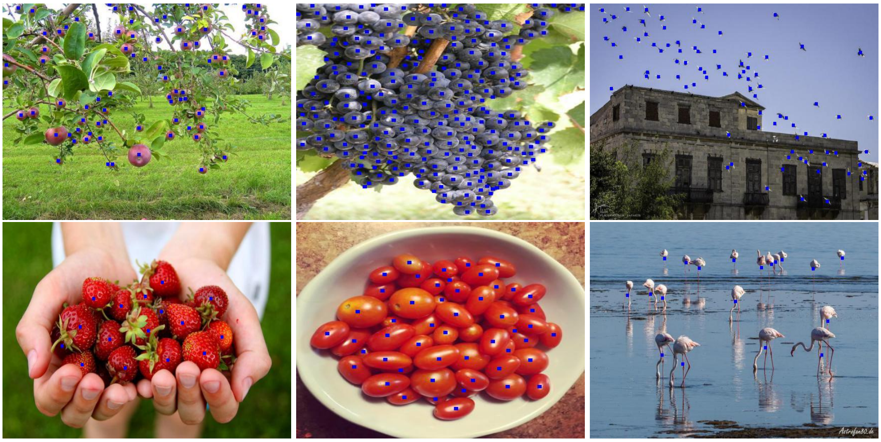


Figure 5.4: Sample images from FSC147 dataset.

where $\hat{y}$ is the predicted count number, y is the ground truth count number and N is the size for the testset.

Mean Absolute Errors (MAE) is defined as:

$$\mathrm{MAE} = \frac{1}{N} \sum_{i=1}^{N} \left\| \hat{y} - y \right\|_1, \tag{5.10}$$

and mean Absolute Errors (NAE) is defined as:

$$\mathrm{NAE} = \frac{1}{N} \sum_{i=1}^{N} \frac{\left\| \hat{y} - y \right\|_1}{y}. \tag{5.11}$$

When evaluating the average performance of all the dataset, we use NAE because its values can be compared over datasets which have varying number of objects in them.

5.4.2 Implementation Details

Our implementation is based on the official code of DM-Count [147]. The feature extractor is the convolutional layers of VGG19 with 512 output channels. The counter head contains of two 3 × 3 convolutional layers with 256 and 128 output channels respectively and a 1 × 1 convolutional layers with 128 output channels. The adapter is a one-layer 1 × 1 convolutional layer with the same number of channels.

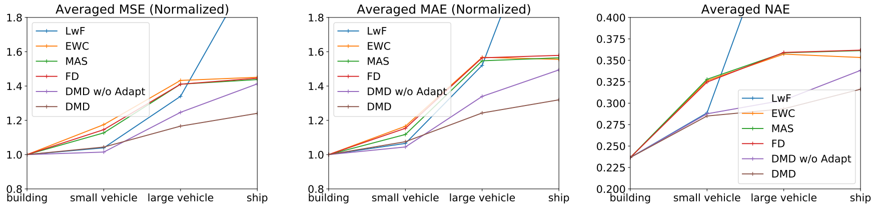


Figure 5.5: Result for RSOC (satellite) Dataset. The performances are evaluated after training of each task. We report the averaged value for all the previously seen tasks. The value is normalized to remove the dataset-scale. Lower value indicate better performance.

We train the model with the Adam optimizer, using batch size 10, learning rate 1e-5, weight-decay 1e-4 and beta 0.9 and 0.999. For each stage, we train the model for 1000 epochs and for the next stage the training is started from the previous model. The hyperparameters $\lambda = 100$ for RSOC dataset and $\lambda = 10$ for both FSC-fruits and FSC-birds.

5.4.3 Results on satellite images

For satellite images, we train our model with four classes *buildings*, *small vehicles*, *large vehicles* and *ships* in sequence from the RSOC dataset. Table 5.1 shows the performance at the end of the incremental learning process, after training all four classes. The performance is evaluated in three metric: MSE, MAE and NMAE. Smaller values indicates better performance. The average performance of the four classes is evaluated with NMAE because of dataset-scale invariance of the NMAE metric.

Finetuning (FT) achieves the best performance on the last task and worst on the first task, as expected. Feature Distillation (FD), EWC [66] and MAS [3] show a similar pattern: they are good at remembering the first task, but have difficulties to learn subsequent tasks. However, they also often perform good in the last task. This might be because the *ships* class is more similar to the first task of *buildings* when comparing to the middle *vehicle* tasks: the stability which prevents these methods from adapting to the *vehicles* tasks, helps it to get acceptable results for *buildings*. LwF [75] performs good on the first task. But it fails in the second and third task due to its very flexible counter head.

Our method DMD w/o Adapt improved the result compared with existing methods, as shown in the averaged value. The good performance in the second and

Dataset:	**building**			**small vehicle**			**large vehicle**			**ship**			**Avg**
Metric:	MSE	MAE	NMAE	MSE	MAE	NMAE	MSE	MAE	NMAE	MSE	MAE	NMAE	NMAE
FT	31.10	27.05	0.926	1266.89	430.81	0.609	52.34	38.70	0.624	86.40	59.51	0.296	0.614
LwF	14.21	10.70	0.360	1326.12	505.10	1.000	79.65	62.78	1.000	137.76	108.27	0.499	0.715
FD	**10.79**	**7.48**	**0.263**	1108.06	356.01	0.379	39.16	27.17	0.423	122.86	88.16	0.382	0.362
EWC	10.94	7.58	0.268	1150.12	360.78	0.383	39.75	27.33	0.418	117.46	80.90	0.345	0.352
MAS	11.07	7.71	0.271	1068.67	333.95	0.380	40.23	27.75	0.419	117.94	85.17	0.375	0.361
DMD w/o Adapt	13.36	9.90	0.336	**929.75**	**291.62**	**0.288**	33.92	22.52	0.345	130.56	87.13	0.384	0.338
DMD	12.63	9.25	0.315	988.63	320.97	0.315	**25.78**	**16.53**	**0.269**	**107.84**	**76.40**	**0.367**	**0.316**

Table 5.1: Performance of several incremental learning methods after learning four tasks of RSOC dataset. In **bold** we show the best results for each column excluding the FT method.

the third task shows that it can learn new task while not forgetting the previous one. After adding the adaptor for the feature extractor, our method DMD further improved the performance, especially on the last task. In conclusion, the proposed density map distillation obtains around a 4% improvement over the best parameter distillation method (EWC).

Additional results are provided in Figure 5.5. Other than Table 5.1 here we provide results after learning each of the tasks. We report the averaged MSE and MAE (normalized for the performance that would be obtained if we only train that task) for all previously seen tasks. For example, the scores reported at task 2 in the graph are the average of normalized MSE obtained on *building* and *small vehicle*) based on the network after training task 2. In the figure, we can observe that the parameter regularization methods EWC and MAS significantly outperform the FT baseline. Next, we observe that our method DMD w/o Adapt obtains significantly better results, especially for averaged NAE. Next, we see that for only two tasks, the proposed DMD method does perform similarly to DMD w/o Adapt. However, for more tasks, DMD does significantly better, and outperforms all methods after four tasks.

5.4.4 Results of counting fruits and birds

Here we consider two incremental learning sequences based on the FSC147 dataset. The first one, FSC-fruits, contains the following tasks *grapes, tomatoes, strawberries,* and *apples.* The second one, FSC-birds, considers the consecutive classification tasks of *flamingos, pigeons, cranes,* and *geese.* Table 5.2, and Table 5.3 summarize the results on FSC-fruits and FSC-birds, respectively.

Similar to the result in RSOC dataset, Finetuning (FT) achieves the best performance on the last task and forgets previous tasks. LwF [75] gives relatively good result in the first and the last task, but failed in the second and third task. MAS [3]

Dataset:	grapes			tomatoes			strawberries			apples			Avg
Metric:	MSE	MAE	NMAE	MSE	MAE	NMAE	MSE	MAE	NMAE	MSE	MAE	NMAE	NMAE
FT	67.11	53.25	0.671	28.49	20.28	0.362	28.11	21.18	0.312	12.85	7.89	0.119	0.366
LwF	27.29	19.74	0.260	63.09	49.54	1.000	73.45	61.02	1.000	**16.50**	**9.77**	**0.135**	0.599
FD	17.26	11.99	0.149	16.44	12.76	0.319	19.83	13.79	0.225	29.35	15.19	0.179	0.218
EWC	17.91	12.40	0.154	17.86	14.07	0.328	21.42	15.16	0.252	22.03	13.69	0.190	0.231
MAS	**16.12**	**11.37**	**0.142**	15.79	12.19	0.298	19.23	13.31	0.212	27.32	15.61	0.211	0.216
DMD w/o Adapt	25.35	18.28	0.240	14.29	11.50	0.260	**16.09**	11.25	**0.177**	23.96	12.33	0.141	0.207
DMD	26.63	18.83	0.250	**11.35**	**8.86**	**0.221**	16.27	**11.09**	0.178	18.56	10.84	0.140	**0.197**

Table 5.2: Performance of several incremental learning methods after learning four tasks on FSC-fruits. In **bold** we show the best results for each column excluding the FT method.

Dataset:	flamingos			pigeons			cranes			geese			Avg
Metric:	MSE	MAE	NMAE	MSE	MAE	NMAE	MSE	MAE	NMAE	MSE	MAE	NMAE	NMAE
FT	74.55	37.77	0.587	41.54	27.12	0.617	12.79	7.63	0.218	7.90	3.54	0.102	0.381
LWF	66.35	27.73	0.326	60.12	42.93	1.000	54.75	32.30	1.000	10.59	5.21	0.153	0.620
FD	64.01	24.85	0.246	27.42	15.30	0.350	13.14	7.09	0.192	13.19	6.78	0.198	0.247
EWC	**62.90**	**23.94**	0.233	23.33	12.06	0.284	12.76	6.49	0.163	12.25	6.41	0.193	0.217
MAS	63.38	24.10	**0.226**	25.07	13.70	0.308	12.96	6.64	0.168	12.04	6.03	0.178	0.220
DMD w/o Adapt	67.35	28.19	0.330	25.96	11.60	0.204	7.78	4.41	0.128	10.72	5.56	0.166	0.207
DMD	67.21	28.66	0.354	**22.52**	**10.56**	**0.198**	**7.23**	**4.16**	**0.123**	**9.01**	**4.37**	**0.133**	**0.202**

Table 5.3: Performance of several incremental learning methods after learning four tasks on FSC-birds. In **bold** we show the best results for each column excluding the FT method.

and EWC [66] give the best result in the first task in FSC-fruits and FSC-birds respectively, but they fail to learn new tasks. Feature Distillation (FD) also performs similarly. FD and MAS work slightly better in FSC-fruits, and EWC works better in FSC-birds.

Our method DMD w/o Adapt improves the result over the above-mentioned methods. Especially, it gets better performance in the new tasks, on both the FSC-fruit and FSC-bird sequence. DMD further improves the result than DMD w/o Adapt, with the feature translation by the adaptor. In FSC-fruits, the performance drops slightly in the first task *grapes* and improves by a large margin in the second task *tomatoes*, compared with DMD w/o Adapt. In FSC-birds, the performance improves in both *pigeons* (second) and *geese* (last) tasks.

Figure 5.6 shows the averaged performance after training of each task. In FSC-fruits, our method DMD outperforms other methods. In FSC-birds, both DMD and DMD w/o Adapt outperform other existing method with a large margin after the third task.

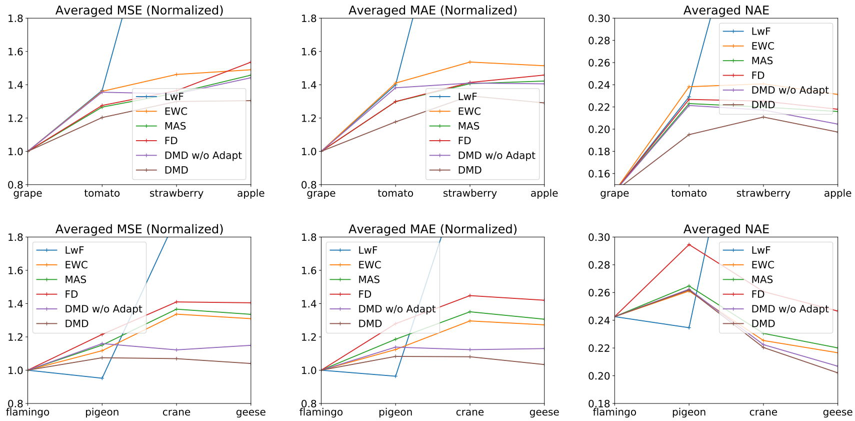


Figure 5.6: Averaged performance for FSC147 Dataset. The performances are evaluated after training of each task. We report the averaged value for all the previously seen tasks. The value is normalized to remove the dataset-scale. Lower value indicate better performance.

Dataset:	grapes			tomatoes			strawberries			apples			Avg
Metric:	MSE	MAE	NMAE	MSE	MAE	NMAE	MSE	MAE	NMAE	MSE	MAE	NMAE	NMAE
DMD w/o Adapt(1)	24.35	15.85	0.200	17.15	13.38	0.302	18.82	14.70	0.266	31.26	16.47	0.195	0.241
DMD w/o Adapt(3)	25.35	18.28	0.240	14.29	11.50	0.260	16.09	11.25	0.177	23.96	12.33	0.141	0.207
DMD(1)	22.82	15.38	0.196	15.40	11.70	0.266	17.76	13.52	0.243	32.43	17.01	0.192	0.224
DMD(3)	26.63	18.83	0.250	11.35	8.86	0.221	16.27	11.09	0.178	18.56	10.84	0.140	0.197

Table 5.4: Ablation of varying number of layers in the counter head on the FSC-fruits sequence.

5.4.5 Ablation Study

Layers of the counter head. We study the effect of using different layers for the counter head in FSC-fruits benchmark. For the comparison, the size of the total network is fixed, so to increase the size of the counter head means that we move few layers from the feature extractor to the counter head. Table 5.4 shows the result of our methods, DMD w/o Adapt and DMD, with 1 or 3 counter head layers. It shows that compared with 1 layer, using 3 layers for the counter head achieves a better performance on newer tasks and a better overall performance.

Hyper parameter for regularization. We study the effect of using different hyperparameter for regularization in FSC (fruits) benchmark. Table 5.4 shows the

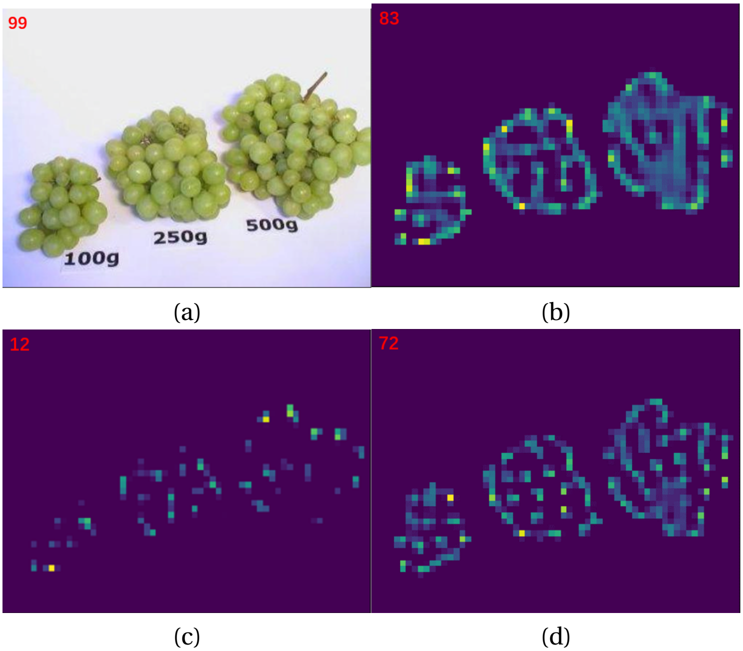
Dataset:	**grapes**			**tomatoes**			**strawberries**			**apples**			**Avg**
Metric:	MSE	MAE	NMAE	MSE	MAE	NMAE	MSE	MAE	NMAE	MSE	MAE	NMAE	NMAE
DMD w/o Adapt(10)	25.35	18.28	0.240	14.29	11.50	0.260	16.09	11.25	0.177	23.96	12.33	0.141	0.207
DMD w/o Adapt(100)	22.47	15.00	0.190	14.98	11.25	0.274	18.14	12.35	0.210	23.65	14.23	0.194	0.217
DMD(10)	26.63	18.83	0.250	11.35	8.86	0.221	16.27	11.09	0.178	18.56	10.84	0.140	0.197
DMD(100)	22.03	15.09	0.193	14.93	11.26	0.272	17.56	11.69	0.196	27.44	15.50	0.190	0.213

Table 5.5: Ablation of different regularization hyperparameter for regularization on FSC-fruits.

result of our methods, DMD w/o Adapt and DMD. With a higher regularization, the performance for the latter tasks drop because it is too rigid for learning new task and the model remembers the first task better.

5.5 Conclusions

We have studied the problem of incremental learning for the object counting problem, and we mainly focus on the density based method. The challenge is to prevent forgetting while learning to count new object categories for new tasks. We propose an exemplar-free method, called Density Map Distillation (DMD). For counting each object, we train a new counter head and all tasks share a feature extractor. We propose to fix the task counter and apply a distillation loss computed with new data on the output of the old counter head. To adapt the changed feature extractor for the fixed counter head, we introduced an adaptor to project the new output feature to the old one. Experiments shows that our method DMD w/o Adapt outperforms those methods adapted from continual learning for classification problems. And with the adaptor, our DMD further improve the performance.



(a) (b) (c) (d)

Figure 5.7: (a) Input Image (b) Density map after training only on the *grapes* data. Density map after learning two additional tasks (*tomatoes* and *strawberries*) with (c) Fine Tuning (FT) and (d) after learning by our proposed method (DMD). In the upper-left corner we show the ground truth number of grapes in (a) and the estimation of the algorithms respectively. Note that naive fine-tuning leads to catastrophic forgetting and the method loses its ability to count *grapes*. (d) Our method manages to get a considerable better count prediction even though there is some performance loss.

6 Conclusions and Future Work

6.1 Conclusions

In this thesis, we aimed to develop continual learning methods for various computer vision applications, including image generation, object re-identification and object counting.

- **Chapter 2: Related Work**

  In this chapter, we reviewed previous work in continual learning for classification problem. We discussed the settings and techniques used in continual learning. Most works discuss these methods in terms of a relatively large category, such as data regularization, parameter regularization, replay, and parameter isolation. We present techniques that belong to these categories, and we also present some other techniques that do not clearly fall into these broad categories, including gradient update modification, task balancing, model generalization, exemplar selection, generative replay and alternative classification methods.

- **Chapter 3: Memory Replay GANs: learning to generate images from new categories without forgetting**

  Previous works on sequential learning address the problem of forgetting in discriminative models. In this chapter, we considered the case of generative models. In particular, we investigated generative adversarial networks (GANs) in the task of learning new categories sequentially. We first showed that sequential fine-tuning renders the network unable to properly generate images from previous categories (i.e. it suffers from forgetting). Addressing this problem, we propose Memory Replay GANs (MeRGANs), a conditional GAN framework that integrates a memory replay generator. We study two methods to prevent forgetting by leveraging these replays, namely joint training with replay and replay alignment. Qualitative and quantitative experimental results in MNIST, SVHN and LSUN datasets show that our memory replay approach can generate competitive images while significantly mitigating the forgetting of previous categories.

- **Chapter 4: Positive Pair Distillation Considered Harmful: Continual Meta Metric Learning for Lifelong Object Re-Identification**

  Lifelong object re-identification incrementally learns from a stream of re-identification tasks. The objective is to learn a representation that can be applied to all tasks and that moreover generalizes to previously unseen re-identification tasks. The main challenge, which distinguishes continual re-identification from standard continual learning scenarios, is the fact that at inference time the representation must generalize to previously unseen identities. To address this problem, we proposed to apply continual meta metric learning to lifelong object re-identification. To prevent forgetting of previous tasks, we used knowledge distillation and explored the roles of positive and negative pairs. Based on our observation that the distillation and metric losses are antagonistic, we proposed to remove positive pairs from distillation to robustify model updates. We call this approach Distillation without Positive Pairs (DwoPP). To verify the effectiveness of DwoPP, we performed extensive intra-domain experimental analysis on person and vehicle re-identification datasets (Market-1501, MSMT17 V2, VeRi-776), as well as inter-domain experiments on the LReID benchmark. Our experiments demonstrated that DwoPP significantly outperforms the state-of-the-art.

- **Chapter 5: Incremental learning of counting new objects by density map distillation**

  In this chapter, we presented a new problem: learn to count a series of new objects incrementally. The challenge is to prevent forgetting of the learned task, while also learning to count new objects. We considered the density map-based method, which is the mainstream counting method. We set up a benchmark that contains a variety of objects, including buildings, vehicles, and boats from satellite, and a variety of different fruits and birds. Existing methods that were directly adapted from continual learning for classification task cannot get optimal performance. Therefore, we proposed an exemplar-free method, Density Map Distillation (DMD). This approach has two main contributions: 1) We found that it is important to fix the old counter head in the task of counting 2) Since the counter head is fixed and the feature will drift, we proposed to use an adaptor to project features from new to old. Experiments showed that our method is effective and outperformed other methods adapted from existing approaches.

6.2 Future work

As presented in Chapter 2, many methods and techniques for continuous learning have been proposed. In the future, we are interested in analyzing these methods and techniques because we have at least two questions in mind.

The first question is why many of the existing techniques for continuous learning do not appear to interfere with each other, but in reality, we do not improve result by simply using them simultaneously. One explanation may come from the perspective of the plasticity-flexibility tradeoff. Some of the different approaches may be similar in nature; they simply place a point on the plasticity-flexibility tradeoff in different ways. But at the same time, we also believe that a more in-depth study may reveal new insights into the combination of methods. The second question is on how these method work in an environment other than the experiments considered in the original papers. For most works, experiments are presented for which the method outperforms other methods. It would be very interesting to show how these methods work in other environments and analyze why certain types of methods work or do not work in certain situations.

Currently, most of the research on continuous learning focuses on the setting where memory is limited or forbidden. In addition to practical considerations, another reason why people are so fascinated by this topic is that human memory is very limited and unstable. We want to develop an artificial intelligence system that works in a similar way to humans themselves. However, memory is actually very cheap and stable for a computer system. Therefore, another interesting direction is to study continuous learning without memory limitations, but taking into account training time or energy consumption.

Publications

1. **Wu, C.**, Herranz, L., Liu, X., Wang, Y., van de Weijer, J., Raducanu, B. (2018). Memory Replay GANs: learning to generate images from new categories without forgetting. (NeurIPS 2018)

2. Wang, K.*, **Wu, C.***, Bagdanov, A., Liu, X., Yang, S., Jui, S., van de Weijer, J.(2022). Positive Pair Distillation Considered Harmful: Continual Meta Metric Learning for Lifelong Object Re-Identification. (BMVC 2022)

3. Liu, X.*, **Wu, C.***, Menta, M., Herranz, L., Raducanu, B., Bagdanov, A., Jui, S.,van de Weijer, J.(2020). Generative feature replay for class-incremental learning. (CVPR 2020 Workshop on Continual Learning)

4. Wang, Y., **Wu, C.**, Herranz, L., van de Weijer, J., Gonzalez-Garcia, A., Raducanu, B. (2018). Transferring GANs: generating images from limited data. (ECCV 2018)

5. Wang, Y, Gonzalez-Garcia, A., **Wu, C.**, Herranz, L., Khan, F.S., Jui, S., Van de Weijer, J. (2021). MineGAN++: Mining Generative Models for Efficient Knowledge Transfer to Limited Data Domains. (Submitted to IJCV)

[*] These authors contributed equally to this work.

Bibliography

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.

[2] Hongjoon Ahn, Jihwan Kwak, Subin Lim, Hyeonsu Bang, Hyojun Kim, and Taesup Moon. Ss-il: Separated softmax for incremental learning. In *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[3] 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)*, pages 139–154, 2018.

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

[5] Jose M Alvarez and Mathieu Salzmann. Learning the number of neurons in deep networks. In *Advances in Neural Information Processing Systems (NeuIPS)*, 2016.

[6] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

[7] Yan Bai, Jile Jiao, Wang Ce, Jun Liu, Yihang Lou, Xuetao Feng, and Ling-Yu Duan. Person30k: A dual-meta generalization network for person re-identification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2123–2132, 2021.

[8] Sungyong Baik, Janghoon Choi, Heewon Kim, Dohee Cho, Jaesik Min, and Kyoung Mu Lee. Meta-learning with task-adaptive loss function for few-shot learning. In *IEEE International Conference on Computer Vision (ICCV)*, pages 9465–9474, 2021.

[9] Peyman Bateni, Raghav Goyal, Vaden Masrani, Frank Wood, and Leonid Sigal. Improved few-shot visual classification. In *Proceedings of the IEEE/CVF*

Conference on Computer Vision and Pattern Recognition (CVPR), pages 14493–14502, 2020.

[10] Eden Belouadah and Adrian Popescu. Il2m: Class incremental learning with dual memory. In *IEEE International Conference on Computer Vision (ICCV)*, pages 583–592, 2019.

[11] Andrew Brock, Jeff Donahue Deepmind, and Karen Simonyan Deepmind. Large scale gan training for high fidelity natural images synthesis. In *International Conference on Learning representations (ICLR)*, 2019.

[12] Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark experience for general continual learning: a strong, simple baseline. In *Advances in Neural Information Processing Systems (NeuIPS)*, 2020.

[13] Fabio Cermelli, Massimiliano Mancini, Samuel Rota Bul'o, Elisa Ricci, and Barbara Caputo. Modeling the background for incremental learning in semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[14] Arslan Chaudhry, Marcus Rohrbach Facebook, A I Research, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip H S Torr, and Marc ' Aurelio Ranzato. On tiny episodic memories in continual learning. In *ICML Workshop*, 2019.

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

[16] Binghui Chen, Zhaoyi Yan, Pengyu Li, Biao Wang, Wangmeng Zuo, and Lei Zhang. Variational attention: Propagating domain-specific knowledge for multi-domain learning in crowd counting. In *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[17] Guangyi Chen, Jiwen Lu, Ming Yang, and Jie Zhou. Spatial-temporal attention-aware learning for video-based person re-identification. *IEEE Transactions on Image Processing*, 28(9):4192–4205, 2019.

[18] Guangyi Chen, Tianren Zhang, Jiwen Lu, and Jie Zhou. Deep meta metric learning. In *IEEE International Conference on Computer Vision (ICCV)*, pages 9547–9556, 2019.

[19] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning (ICML)*, 2020.

[20] Wei Chen, Yu Liu, Weiping Wang, Tinne Tuytelaars, Erwin M Bakker, and Michael Lew. On the exploration of incremental learning for fine-grained image retrieval. *BMVA British Machine Vision Conference (BMVC)*, 2020.

[21] Yinbo Chen, Zhuang Liu, Huijuan Xu, Trevor Darrell, and Xiaolong Wang. Meta-baseline: Exploring simple meta-learning for few-shot learning. In *IEEE International Conference on Computer Vision (ICCV)*, pages 9062–9071, 2021.

[22] Seokeon Choi, Taekyung Kim, Minki Jeong, Hyoungseob Park, and Changick Kim. Meta batch-instance normalization for generalizable person re-identification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3425–3435, 2021.

[23] Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 539–546. IEEE, 2005.

[24] Yongxing Dai, Xiaotong Li, Jun Liu, Zekun Tong, and Ling-Yu Duan. Generalizable person re-identification with relevance-aware mixture of experts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16145–16154, 2021.

[25] Riccardo Del Chiaro, Bartłomiej Twardowski, Andrew Bagdanov, and Joost Van de Weijer. Ratt: Recurrent attention to transient tasks for continual image captioning. In *Advances in Neural Information Processing Systems (NeuIPS)*, 2020.

[26] 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.

[27] 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.

[28] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A learned representation for artistic style. In *International Conference on Learning representations (ICLR)*, 2017.

[29] Steffen Gais et al. Sleep transforms the cerebral trace of declarative memories. *Proceedings of the National Academy of Sciences*, 104(47):18778–18783, 2007.

[30] Mehrdad Farajtabar, Navid Azizan, Alex Mott, Ang Li, Deepmind Caltech, and Deepmind Deepmind. Orthogonal gradient descent for continual learning. In *AISTATS*, 2020.

[31] Enrico Fini, Victor G Turrisi da Costa, Xavier Alameda-Pineda, Elisa Ricci, Karteek Alahari, and Julien Mairal. Self-supervised models are continual learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9621–9630, 2022.

[32] 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, 2017.

[33] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 2015.

[34] Guangshuai Gao and Qingjie Liu. Counting from sky: A large-scale dataset for remote sensing object counting and a benchmark method. *IEEE Transactions on geoscience and remote sensing*, 2020.

[35] Junyu Gao, Yuan Yuan, and Qi Wang. Feature-aware adaptation and density alignment for crowd counting in video surveillance. *IEEE Transactions on Cybernetics*, 2020.

[36] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. In *International Conference on Learning representations (ICLR)*, 2018.

[37] Alex Gomez-Villa, Bartlomiej Twardowski, Lu Yu, Andrew D Bagdanov, and Joost van de Weijer. Continually learning self-supervised representations with projected functional regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3867–3877, 2022.

[38] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems (NeuIPS)*, 2014.

[39] Ian J Goodfellow, Mehdi Mirza, Da Xiao, Aaron Courville, and Yoshua Bengio. An empirical investigation of catastrophic forgetting in gradient-based neural networks. In *International Conference on Learning representations (ICLR)*, 2014.

[40] Douglas Gray and Hai Tao. Viewpoint invariant pedestrian recognition with an ensemble of localized features. In *European Conference on Computer Vision (ECCV)*, pages 262–275. Springer, 2008.

[41] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems (NeuIPS)*, pages 5769–5779, 2017.

[42] Gunshi Gupta, Karmesh Yadav, and Liam Paull. La-maml: Look-ahead meta learning for continual learning. In *Advances in Neural Information Processing Systems (NeuIPS)*, 2020.

[43] Song Han, Huizi Mao, Enhao Gong, Shijian Tang, William J Dally, Jeff Pool, John Tran, Bryan Catanzaro, Sharan Narang, Erich Elsen, Peter Vajda, and Manohar Paluri. Dsd: Dense-sparse-dense training for deep neural networks. In *International Conference on Learning representations (ICLR)*, 2017.

[44] Tao Han, Junyu Gao, Yuan Yuan, and Qi Wang. Focus on semantic consistency for cross-domain crowd understanding. In *ICASSP*, 2020.

[45] 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.

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

[47] Shuting He, Hao Luo, Pichao Wang, Fan Wang, Hao Li, and Wei Jiang. Transreid: Transformer-based object re-identification. *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[48] D.O. Hebb. *The Organization of Behavior: A Neuropsychological Theory*. New York Wiely, 2002.

[49] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, Günter Klambauer, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a nash equilibrium. *arXiv preprint arXiv:1706.08500*, 2017.

[50] Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. In *NIPS Workshop*, 2015.

[51] Martin Hirzer, Csaba Beleznai, Peter M Roth, and Horst Bischof. Person re-identification by descriptive and discriminative classification. In *Scandinavian conference on Image analysis*, pages 91–102. Springer, 2011.

[52] Elad Hoffer and Nir Ailon. Deep metric learning using triplet network. In *International workshop on similarity-based pattern recognition*, pages 84–92. Springer, 2015.

[53] 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.

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

[55] Seyed Iman Mirzadeh and Hassan Ghasemzadeh. Understanding the role of training regimes in continual learning. In *Advances in Neural Information Processing Systems (NeuIPS)*, 2020.

[56] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[57] Khurram Javed and Martha White. Meta-learning representations for continual learning. In *Advances in Neural Information Processing Systems (NeuIPS)*, 2019.

[58] Minsoo Kang, Jaeyoo Park, and Bohyung Han. Class-incremental learning by knowledge distillation with adaptive feature consolidation. In *Proceedings of*

the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022.

[59] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. In *International Conference on Learning representations (ICLR)*, 2018.

[60] Ronald Kemker and Christopher Kanan. Fearnet: Brain-inspired model for incremental learning. *arXiv preprint arXiv:1711.10563*, 2017.

[61] Pirazh Khorramshahi, Amit Kumar, Neehar Peri, Sai Saketh Rambhatla, Jun-Cheng Chen, and Rama Chellappa. A dual-path model with adaptive attention for vehicle re-identification. In *IEEE International Conference on Computer Vision (ICCV)*, 2019.

[62] Sungyeon Kim, Dongwon Kim, Minsu Cho, and Suha Kwak. Proxy anchor loss for deep metric learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3238–3247, 2020.

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

[64] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning representations (ICLR)*, 2013.

[65] Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in Neural Information Processing Systems (NeuIPS)*, volume 31. Curran Associates, Inc., 2018.

[66] 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.

[67] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Learning representations for automatic colorization. In *European Conference on Computer Vision (ECCV)*, 2016.

[68] Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

[69] Hankook Lee, Sung Ju Hwang, and Jinwoo Shin. Self-supervised label augmentation via input transformations. In *International Conference on Machine Learning (ICML)*, pages 5670–5680, 2020.

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

[71] Wei Li and Xiaogang Wang. Locally aligned feature transforms across views. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3594–3601, 2013.

[72] Wei Li, Rui Zhao, and Xiaogang Wang. Human reidentification with transferred metric learning. In *Asian Conference on Computer Vision (ACCV)*, pages 31–44. Springer, 2012.

[73] Wei Li, Rui Zhao, Tong Xiao, and Xiaogang Wang. Deepreid: Deep filter pairing neural network for person re-identification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 152–159, 2014.

[74] Yulin Li, Jianfeng He, Tianzhu Zhang, Xiang Liu, Yongdong Zhang, and Feng Wu. Diverse part discovery: Occluded person re-identification with part-aware transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2898–2907, 2021.

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

[76] Dongze Lian, Jing Li, Jia Zheng, Weixin Luo, and Shenghua Gao. Density map regression guided detection network for rgb-d crowd counting and localization. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1821–1830, 2019.

[77] Chenchen Liu, Xinyu Weng, and Yadong Mu. Recurrent attentive zooming for joint crowd counting and precise localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

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

[79] 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.

[80] 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.

[81] 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.

[82] 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 Workshops*, pages 226–227, 2020.

[83] Xinchen Liu, Wu Liu, Huadong Ma, and Huiyuan Fu. Large-scale vehicle re-identification in urban surveillance videos. In *2016 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6. IEEE, 2016.

[84] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. *Advances in Neural Information Processing Systems (NeuIPS)*, 2017.

[85] Yihang Lou, Yan Bai, Jun Liu, Shiqi Wang, and Ling-Yu Duan. Embedding adversarial learning for vehicle re-identification. *IEEE Transactions on Image Processing*, 28(8):3794–3807, 2019.

[86] Chen Change Loy, Tao Xiang, and Shaogang Gong. Time-delayed correlation analysis for multi-camera activity understanding. *International Journal of Computer Vision*, 90(1):106–129, 2010.

[87] Erika Lu, Weidi Xie, and Andrew Zisserman. Class-agnostic counting. In *Asian Conference on Computer Vision (ACCV)*, 2018.

[88] Hao Luo, Youzhi Gu, Xingyu Liao, Shenqi Lai, and Wei Jiang. Bag of tricks and a strong baseline for deep person re-identification. In *Proceedings of the*

IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, 2019.

[89] Zhiheng Ma, Xiaopeng Hong, Xing Wei, Yunfeng Qiu, and Yihong Gong. Towards a universal model for cross-dataset crowd counting. In *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[90] Zhiheng Ma, Xing Wei, Xiaopeng Hong, and Gong Yihong. Bayesian loss for crowd count estimation with point supervision. In *IEEE International Conference on Computer Vision (ICCV)*, 2019.

[91] 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.

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

[93] Xudong Mao, Qing Li, Haoran Xie, Raymond Y K Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[94] 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.

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

[96] James L McClelland, Bruce L McNaughton, and Randall C O'Reilly. Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory. *Psychological review*, 102(3):419, 1995.

[97] 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.

[98] 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.

[99] Martial Mermillod, Aurélia Bugaiska, and Patrick Bonin. The stability-plasticity dilemma: Investigating the continuum from catastrophic forgetting to age-limited learning effects, 2013.

[100] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[101] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning representations (ICLR)*, 2018.

[102] Takeru Miyato and Masanori Koyama. cgans with projection discriminator. In *International Conference on Learning representations (ICLR)*, 2018.

[103] Kevin Musgrave, Serge Belongie, and Ser-Nam Lim. A metric learning reality check. In *European Conference on Computer Vision (ECCV)*, pages 681–699. Springer, 2020.

[104] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

[105] Hao Ni, Jingkuan Song, Xiaopeng Luo, Feng Zheng, Wen Li, and Heng Tao Shen. Meta distribution alignment for generalizable person re-identification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2487–2496, 2022.

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

[107] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. In *International Conference on Machine Learning (ICML)*, pages 2642–2651, 2017.

[108] Oleksiy Ostapenko, Mihai Puscas, Tassilo Klein, Patrick Jahnichen, and Moin Nabi. Learning to remember: A synaptic plasticity driven framework for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[109] Jaeyoo Park, Minsoo Kang, and Bohyung Han. Class-incremental learning for action recognition in videos. In *IEEE International Conference on Computer Vision (ICCV)*, pages 13698–13707, October 2021.

[110] Razvan Pascanu and Yoshua Bengio. Revisiting natural gradient for deep networks. *arXiv preprint arXiv:1301.3584*, 2013.

[111] Quang Pham, Chenghao Liu, and Steven C H Hoi. Dualnet: Continual learning, fast and slow. In *Advances in Neural Information Processing Systems (NeuIPS)*, 2021.

[112] 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 (CVPR)*, pages 7901–7910, 2021.

[113] Juan-Manuel Pérez-Rúa, Xiatian Zhu, Timothy Hospedales, and Tao Xiang. Incremental few-shot object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[114] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional. In *International Conference on Learning representations (ICLR)*, 2014.

[115] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *International Conference on Learning representations (ICLR)*, 2016.

[116] 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.

[117] Viresh Ranjan, Udbhav Sharma, Thu Nguyen, and Minh Hoai. Learning to count everything. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[118] Roger Ratcliff. Connectionist models of recognition memory: constraints imposed by learning and forgetting functions. *Psychological review*, 97(2):285, 1990.

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

[120] Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, and Gerald Tesauro. Learning to learn without forgetting by maximizing transfer and minimizing interference. In *International Conference on Learning representations (ICLR)*, 2019.

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

[122] 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.

[123] 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.

[124] Gobinda Saha, Isha Garg, and Kaushik Roy. Gradient projection memory for continual learning. In *International Conference on Learning representations (ICLR)*, 2021.

[125] Deepak Babu Sam, Skand Vishwanath Peri, Mukuntha Narayanan Sundararaman, Amogh Kamath, and R. Venkatesh Babu. Locate, size and count: Accurately resolving people in dense crowds via detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[126] Deepak Babu Sam, Shiv Surya, and R Venkatesh Babu. Switching convolutional neural network for crowd counting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

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

[128] Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *International Conference on Machine Learning (ICML)*, 2018.

[129] Min Shi, Hao Lu, Chen Feng, Chengxin Liu, and Zhiguo Cao. Represent, compare, and learn: A similarity-aware framework for class-agnostic counting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[130] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. In *Advances in Neural Information Processing Systems (NeuIPS)*, 2017.

[131] K. Shmelkov, C. Schmid, and K. Alahari. Incremental learning of object detectors without catastrophic forgetting. In *IEEE International Conference on Computer Vision (ICCV)*, pages 3420–3429, 2017.

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

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

[134] Vishwanath A Sindagi and Vishal M Patel. Generating high-quality crowd density maps using contextual pyramid cnns. In *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[135] James Smith, Yen-Chang Hsu, Jonathan Balloch, Yilin Shen, Hongxia Jin, and Zsolt Kira. Always be dreaming: A new approach for data-free class-incremental learning. In *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[136] Jake Snell, Kevin Swersky, and Richard S Zemel. Prototypical networks for few-shot learning. *Advances in Neural Information Processing Systems (NeuIPS)*, 2017.

[137] Jifei Song, Yongxin Yang, Yi-Zhe Song, Tao Xiang, and Timothy M Hospedales. Generalizable person re-identification by domain-invariant mapping network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 719–728, 2019.

[138] Qingyu Song, Changan Wang, Zhengkai Jiang, Yabiao Wang, Ying Tai, Chengjie Wang, Jilin Li, Feiyue Huang, and Yang Wu. Rethinking counting and localization in crowds: A purely point-based framework. In *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[139] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *Advances in Neural Information Processing Systems (NeuIPS)*, 2019.

[140] 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.

[141] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1199–1208, 2018.

[142] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1701–1708, 2014.

[143] Gido M. van de Ven and Andreas S Tolias. Three scenarios for continual learning. In *NeurIPS CL Workshop*, 2019.

[144] Ricardo Vilalta and Youssef Drissi. A perspective view and survey of meta-learning. *Artificial intelligence review*, 18(2):77–95, 2002.

[145] Cédric Villani. *Optimal transport: old and new*, volume 338. Springer Science & Business Media, 2008.

[146] Jia Wan, Ziquan Liu, and Antoni B Chan. A generalized loss function for crowd counting and localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[147] Boyu Wang, Huidong Liu, Dimitris Samaras, and Minh Hoai. Distribution matching for crowd counting. In *NeurIPS*, 2020.

[148] Feng Wang, Jian Cheng, Weiyang Liu, and Haijun Liu. Additive margin softmax for face verification. *IEEE Signal Processing Letters*, 25(7):926–930, 2018.

[149] Feng Wang, Xiang Xiang, Jian Cheng, and Alan Loddon Yuille. Normface: L2 hypersphere embedding for face verification. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 1041–1049, 2017.

[150] Kai Wang, Xialei Liu, Andrew D. Bagdanov, Luis Herranz, Shangling Jui, and Joost van de Weijer. Incremental meta-learning via episodic replay distillation for few-shot image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 3729–3739, June 2022.

[151] Kai Wang, Chenshen Wu, Xialei Liu, Shiqi Yang, Shangling Jui, and Joost van de Weijer. Positive pair distillation considered harmful: Continual meta metric learning for lifelong object re-identification. In *BMVA British Machine Vision Conference (BMVC)*, 2022.

[152] Qi Wang, Junyu Gao, Wei Lin, and Yuan Yuan. Learning from synthetic data for crowd counting in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[153] 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.

[154] Yaxing Wang, Chenshen Wu, Luis Herranz, Joost van de Weijer, Abel Gonzalez-Garcia, and Bogdan Raducanu. Transferring gans: generating images from limited data. In *European Conference on Computer Vision(ECCV)*, 2018.

[155] Longhui Wei, Shiliang Zhang, Wen Gao, and Qi Tian. Person transfer gan to bridge domain gap for person re-identification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 79–88, 2018.

[156] Max Welling. Herding dynamical weights to learn. In *International Conference on Machine Learning (ICML)*, pages 1121–1128, 2009.

[157] Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Learning structured sparsity in deep neural networks. In *Advances in Neural Information Processing Systems (NeuIPS)*, 2016.

[158] Mitchell Wortsman, Vivek Ramanujan, Rosanne Liu, Aniruddha Kembhavi, Mohammad Rastegari, Jason Yosinski, and Ali Farhadi. Supermasks in superposition. In *Advances in Neural Information Processing Systems (NeuIPS)*, 2020.

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

[160] Chenshen Wu, Luis Herranz, Xialei Liu, Yaxing Wang, Joost Van De Weijer, and Bogdan Raducanu. Memory replay gans: learning to generate images from new categories without forgetting. In *Advances in Neural Information Processing Systems (NeuIPS)*, 2018.

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

[162] Gui-Song Xia, Xiang Bai, Jian Ding, Zhen Zhu, Serge Belongie, Jiebo Luo, Mihai Datcu, Marcello Pelillo, and Liangpei Zhang. Dota: A large-scale dataset for object detection in aerial images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[163] Tong Xiao, Shuang Li, Bochao Wang, Liang Lin, and Xiaogang Wang. End-to-end deep learning for person search. *arXiv preprint arXiv:1604.01850*, 2(2):4, 2016.

[164] Shipeng Yan, Jiangwei Xie, and Xuming He. Der: Dynamically expandable representation for class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3014–3023, 2021.

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

[166] Mang Ye, Jianbing Shen, Gaojie Lin, Tao Xiang, Ling Shao, and Steven CH Hoi. Deep learning for person re-identification: A survey and outlook. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

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

[168] Fisher Yu, Yinda Zhang, Shuran Song, Ari Seff, and Jianxiong Xiao. Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015.

[169] Lu Yu, Bartłomiej Twardowski, Xialei Liu, Luis Herranz, Kai Wang, Yongmei Cheng, Shangling Jui, and Joost van de Weijer. Semantic drift compensation for class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6980–6989, 2020.

[170] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. *arXiv preprint arXiv:2103.03230*, 2021.

[171] Guanxiong Zeng, Yang Chen, Bo Cui, and Shan Yu. Continual learning of context-dependent processing in neural networks. *Nature Machine Intelligence*, 2019.

[172] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International Conference on Machine Learning (ICML)*, pages 3987–3995. JMLR. org, 2017.

[173] 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 *IEEE International Conference on Computer Vision (ICCV)*, pages 5907–5915, 2017.

[174] Xiao Zhang, Yixiao Ge, Yu Qiao, and Hongsheng Li. Refining pseudo labels with clustering consensus over generations for unsupervised object re-identification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3436–3445, 2021.

[175] Yingying Zhang, Desen Zhou, Siqin Chen, Shenghua Gao, and Yi Ma. Single-image crowd counting via multi-column convolutional neural network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2016-Decem, pages 589–597, 2016.

[176] Bo Zhao, Shixiang Tang, Dapeng Chen, Hakan Bilen, and Rui Zhao. Continual representation learning for biometric identification. In *Proceedings of the IEEE Workshop on Applications of Computer Vision*, pages 1198–1208, 2021.

[177] Borui Zhao, Quan Cui, Renjie Song, Yiyu Qiu, and Jiajun Liang. Decoupled knowledge distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11953–11962, 2022.

[178] Haiyu Zhao, Maoqing Tian, Shuyang Sun, Jing Shao, Junjie Yan, Shuai Yi, Xiaogang Wang, and Xiaoou Tang. Spindle net: Person re-identification with human body region guided feature decomposition and fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1077–1085, 2017.

[179] Jianan Zhao, Fengliang Qi, Guangyu Ren, and Lin Xu. Phd learning: Learning with pompeiu-hausdorff distances for video-based vehicle re-identification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2225–2235, 2021.

[180] Liang Zheng, Liyue Shen, Lu Tian, Shengjin Wang, Jingdong Wang, and Qi Tian. Scalable person re-identification: A benchmark. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1116–1124, 2015.

[181] Wei-Shi Zheng, Shaogang Gong, and Tao Xiang. Associating groups of people. In *BMVA British Machine Vision Conference (BMVC)*, volume 2, pages 1–11, 2009.

[182] Fei Zhu, Xu-Yao Zhang, Chuang Wang, Fei Yin, and Cheng-Lin Liu. Prototype augmentation and self-supervision for incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5871–5880, 2021.

[183] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.