Universitat Autònoma
de Barcelona

Visual recognition in the wild: learning from rankings in small domains and continual learning in new domains

A dissertation submitted by **Xialei Liu** 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, 2019

Director	**Dr. Joost van de Weijer** Centre de Visió per Computador Universitat Autònoma de Barcelona **Dr. Andrew D. Bagdanov** Media Integration and Communication Center University of Florence
Thesis committee	**Dr. Tinne Tuytelaars** Department of Electrical Engineering Katholieke Universiteit Leuven **Dr. Marta R. Costa-jussà** Signal Theory and Communications Department Universitat Politècnica de Catalunya **Dr. Dimosthenis Karatzas** Centre de Visió per Computador Universitat Autònoma de Barcelona **Dr. David Berga Garreta** Centre de Visió per Computador Universitat Autònoma de Barcelona **Dr. Petia Radeva** Department of Mathematics and Computer Science Universitat de Barcelona

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

ISBN: 978-84-121011-4-0

Printed by Ediciones Gráficas Rey, S.L.

Acknowledgements

It has been a long journey with ups and downs, and it would not have been possible without all distinct individuals supporting me. I appreciate all of you sincerely from the bottom of my heart.

I still remember the first day I came to Barcelona, sunny and mild. I met my supervisors Prof. Joost van de Weijer and Prof. Andrew D. Bagdanov, who have been the most important people for me to finish my PhD. Prof. Andrew D. Bagdanov is an expert on computer science and mathematics. At the very beginning, you spared no effort to teach me how to become a better programmer, how to read papers and how to improve my English skills from the countless meetings and emails. Prof. Joost van de Weijer is an excellent researcher and a great leader. You scheduled the basic courses for us thoughtfully and discussed together in the group meetings. You helped me understand important concepts very quickly. You taught me in all aspects of research from ideas, experiments to writing. It is hard to imagine how to publish my first paper without your support, guidance and patience. More importantly, you have treated me as a friend and set a good example for me to behave in the future. Furthermore, you both have shared a positive attitude towards life and work, which will be priceless and valuable in the future to come.

I am happy that I could witness some of important moments in your life as well. Prof. van de Weijer is now a father of an adorable baby and Prof. Andrew D. Bagdanov is now a husband of an elegant lady. I am happy to see that our group have grown and matured a lot during this four years. I wish a prosperous future for everyone.

I would also like to thank the great LAMPers for their kind support. My sincere thanks to Marc, Yaxing, Lichao, Laura, Aymen, Oguz, Aitor, Chenshen, Kai, Mikel, Olaia, Carola, Javad. Especially Marc helped me a lot to settle down everything and make me feel very welcome in this foreign country. I also appreciate the guidance and help from Luis, Bogdan, Abel and Mikhail. Thanks to CVC for providing me with an open-minded and multi-cultural environment. Many thanks to all CVC people with your kind interactions, especially to Dena, Yi, Corina, Zhijie, Lei, Arash, Akhil, Pau, Montse and Gigi. I also want to extend my appreciation to my dear friends in Barcelona, Fangchang, Tingting, Xiaoqing, Rong, Yi, Jianqiang, Yu, Junpeng. I

enjoyed every moment during my stay in Barcelona.

I would also like to thank my colleagues in Amazon. I appreciate the support and useful discussions from my manager Avinash and my mentor Hao Yang. Many thanks to Davide, Wei, Hao Li, Hao Chen, Hao Wu, and all the interns in the Video and Rekognition group.

Last but not least, I would like to thank my family for your endless support. You always give me the best you can. You always believe that knowledge can change people's fate and you have always believed in me. I apologize for not being able to be there all the time by your side, and I have missed your company. Distance can be hard, but feelings are always strong. I know that no matter what happens, you always have my back. Moreover, thanks to my country for supporting me.

Finally, to my love Lu, time flies, I am so lucky that we have been together for seven amazing years. I am so grateful that at every moment I know you always stand beside me and support me unconditionally. You help make all distractions go away to allow me to be more concentrated. And you relieve my stress to let me calm down. Above all, you make my life colorful and wonderful. You are my love forever.

Abstract

Deep convolutional neural networks (CNNs) have achieved superior performance in many visual recognition application, such as image classification, detection and segmentation. In this thesis we address two limitations of CNNs. Training deep CNNs requires huge amounts of labeled data, which is expensive and labor intensive to collect. Another limitation is that training CNNs in a continual learning setting is still an open research question. Catastrophic forgetting is very likely when adapting trained models to new environments or new tasks. Therefore, in this thesis, we aim to improve CNNs for applications with limited data and to adapt CNNs continually to new tasks.

Self-supervised learning leverages unlabelled data by introducing an auxiliary task for which data is abundantly available. In the first part of the thesis, we show how rankings can be used as a proxy self-supervised task for regression problems. Then we propose an efficient backpropagation technique for Siamese networks which prevents the redundant computation introduced by the multi-branch network architecture. In addition, we show that measuring network uncertainty on the self-supervised proxy task is a good measure of informativeness of unlabeled data. This can be used to drive an algorithm for active learning. We then apply our framework on two regression problems: Image Quality Assessment (IQA) and Crowd Counting. For both, we show how to automatically generate ranked image sets from unlabeled data. Our results show that networks trained to regress to the ground truth targets for labeled data and to simultaneously learn to rank unlabeled data obtain significantly better, state-of-the-art results. We further show that active learning using rankings can reduce labeling effort by up to 50% for both IQA and crowd counting.

In the second part of the thesis, we propose two approaches to avoiding catastrophic forgetting in sequential task learning scenarios. The first approach is derived from Elastic Weight Consolidation, which uses a diagonal Fisher Information Matrix (FIM) to measure the importance of the parameters of the network. However the diagonal assumption is unrealistic. Therefore, we approximately diagonalize the FIM using a set of factorized rotation parameters. This leads to significantly better performance on continual learning of sequential tasks. For the second approach, we show that forgetting manifests differently at different layers in the network and

propose a hybrid approach where distillation is used in the feature extractor and replay in the classifier via feature generation. Our method addresses the limitations of generative image replay and probability distillation (i.e. *learning without forgetting*) and can naturally aggregate new tasks in a single, well-calibrated classifier. Experiments confirm that our proposed approach outperforms the baselines and some start-of-the-art methods.

Key words: *visual recognition, self-supervised learning, learning from rankings, image quality assessment, crowd counting, continual learning*

Resumen

Las redes neuronales convolucionales profundas (CNNS) han alcanzado resultados muy positivos en diferentes aplicaciones de reconocimiento visual, tales como clasificación, detección o segmentación de imágenes. En esta tesis, abordamos dos limitaciones de las CNNs. La primera, entrenar CNNs profundas requiere grandes cantidades de datos etiquetados, los cuales son muy costosos y arduos de conseguir. La segunda es que entrenar en sistemas de aprendizaje continuo es un problema abierto para la investigación. El olvido catastrófico en redes es muy común cuando se adapta un modelo entrenado a nuevos entornos o nuevas tareas. Por lo tanto, en esta tesis, tenemos como objetivo mejorar las CNNs para aplicaciones con datos limitados y adaptarlas de forma continua a nuevas tareas.

El aprendizaje auto-supervisado compensa la falta de datos etiquetados con la introducción de tareas auxiliares en las cuales los datos están fácilmente disponibles. En la primera parte de la tesis, mostramos cómo los ránquings se pueden utilizar de forma parecida a una tarea auto-supervisada para los problemas de regresión. Después, proponemos una técnica de propagación hacia atrás eficiente para redes siamesas que previene el computo redundante introducido por las arquitecturas de red multi-rama. Además, demostramos que medir la incertidumbre de las redes en las tareas parecidas a las auto-supervisadas, es una buena medida de la cantidad de información que contienen los datos no etiquetados. Dicha medida puede ser entonces usada para la ejecución de algoritmos de aprendizaje activo. Estos marcos que proponemos los aplicamos entonces a dos problemas de regresión: Evaluación de Calidad de Imagen (IQA) y el contador de personas. En los dos casos, mostramos cómo generar de forma automática grupos de imágenes ranqueadas para los datos no etiquetados. Nuestros resultados muestran que las redes entrenadas para la regresión de las anotaciones de los datos etiquetados, a la vez que para aprender a ordenar los ránquings de los datos no etiquetados, obtienen resultados significativamente mejores al estado del arte. También demostramos que el aprendizaje activo utilizando ránquings puede reducir la cantidad de etiquetado en un 50 % para ambas tareas de IQA y contador de personas.

En la segunda parte de la tesis, proponemos dos métodos para evitar el olvido catastrófico en escenarios de aprendizaje secuencial de tareas. El primer método deriva del de Consolidación Elástica de Pesos, el cuál utiliza la diagonal de la Matriz de Información de Fisher (FIM) para medir la importancia de los pesos de la red. No obstante, la aproximación asumida no es realista. Por lo tanto, diagonalizamos la aproximación de la FIM utilizando un grupo de parámetros de rotación factorizada proporcionando una mejora significativa en el rendimiento de tareas secuenciales para el caso del aprendizaje continuo. Para el segundo método, demostramos que el olvido se manifiesta de forma diferente en cada capa de la red y proponemos un método híbrido donde la destilación se utiliza para el extractor de características y la rememoración en el clasificador mediante generación de características. Nuestro método soluciona la limitación de la rememoración mediante generación de imágenes y la destilación de probabilidades (como la utilizada en el método *Aprendizaje Sin Olvido*), y puede añadir de forma natural nuevas tareas en un único clasificador bien calibrado. Los experimentos confirman que el método propuesto sobrepasa las métricas de referencia y parte del estado del arte.

Palabras clave: *reconocimiento visual, aprendizaje auto-supervisado, aprendizaje mediante ránquings, evaluación de calidad de imagen, contador de personas, aprendizaje continuo*

Resum

Les xarxes neuronals convolucionals profundes (CNNs) han assolit resultats molt positius en diverses aplicacions de reconeixement visual, tals com classificació, detecció o segmentació d'imatges. En aquesta tesis, abordem dues limitacions de les CNNs. La primera, entrenar CNNs profundes requereix grans quantitats de dades etiquetades, les quals són molt costoses i àrdues d'aconseguir. La segona és que entrenar CNNs en sistemes d'aprenentatge continuu és un problema obert per a la recerca. L'oblit catastròfic en xarxes és molt comú quan s'adapta un model entrenat a nous entorns o noves tasques. Per tant, en aquesta tesis, tenim com a objectiu millorar les CNNs per a les aplicacions amb dades limitades i adaptar-les de forma contínua en noves tasques.

L'aprenentatge auto-supervisat compensa la falta de dades etiquetades amb la introducció de tasques auxiliars en les quals les dades estan fàcilment disponibles. En la primera part de la tesis, mostrem com els rànquings es poden utilitzar de forma semblant a una tasca auto-supervisada per a problemes de regressió. Després, proposem una tècnica de propagació cap endarrera eficient per a xarxes siameses que prevenen el còmput redundant introduït per les arquitectures de xarxa multi-branca. A més a més, demostrem que mesurar la incertesa de les xarxes en les tasques semblants a les auto-supervisades és una bona mesura de la quantitat d'informació que contenen les dades no etiquetades. Aquesta mesura pot ser, aleshores, utilitzada per a l'execució de algoritmes d'aprenentatge actiu. Aquests marcs que proposem els apliquem doncs a dos problemes de regressió: Avaluació de la Qualitat d'Imatge (IQA) i el comptador de persones. En els dos casos, mostrem com generar de forma automàtica grups d'imatges ranquejades per a les dades no etiquetades. Els nostres resultats mostren que les xarxes entrenades per a la regressió de les anotacions de les dades etiquetades a la vegada que per aprendre a ordenar els rànquings de les dades no etiquetades, obtenen significativament millors resultats que superen l'estat de l'art. També demostrem que l'aprenentatge actiu utilitzant rànquings pot reduir la quantitat d'etiquetatge en un 50% per ambdues tasques de IQA i comptador de persones.

A la segona part de la tesis, proposem dos mètodes per a evitar l'oblit catastròfic en escenaris d'aprenentatge seqüencial de tasques. El primer mètode deriva del de Consolidació Elàstica de Pesos, el qual utilitza la diagonal de la Matriu

d'Informació de Fisher (FIM) per a mesurar la importància dels paràmetres de la xarxa. No obstant, l'aproximació assumida no és realista. Per tant, diagonalitzem aproximadament la FIM utilitzant un grup de paràmetres de rotació factoritzada proporcionant una millora significativa del rendiment de tasques seqüencials en el cas de l'aprenentatge continu. Per al segon mètode, demostrem que l'oblit es manifesta de forma diferent en cada capa de la xarxa i proposem un mètode híbrid on la destil·lació s'utilitza per a l'extractor de característiques i la rememoració en el classificador mitjançant generació de característiques. El nostre mètode soluciona la limitació de la rememoració mitjançant la generació d'imatges i la destil·lació de probabilitats (com l'utilitzat en el mètode *Aprenentatge Sense Oblit*), i pot afegir de forma natural noves tasques en un únic classificador ben calibrat. Els experiments confirmen que el mètode proposat sobrepassa les mètriques de referència i part de l'estat de l'art.

Paraules clau: *reconeixement visual, aprenentatge auto-supervisat, aprenentatge per rànquings, avaluació de qualitat d'imatge, contador de persones, aprenentatge continu*

Contents

List of Figures

List of Tables

1 Introduction

Visual recognition is one of the basic skills which humans acquire early in life. We are capable of perceiving the physical properties of objects such as color, shape and texture. Furthermore, we can also associate semantic meanings to different visual objects like apples, an elephant, or a building. In this process, we understand how to distinguish different objects, how different objects relate to others, and what is their usage [27]. Regardless of location, illumination, occlusion, or perspective of an object, humans are able to recognize them with very high accuracy.

Computer vision aims at understanding visual objects in digital images or videos like human beings understand them [44]. Especially in this Era of Internet, images and videos are everywhere in our daily lives. Equipped with computer vision systems, intelligent machines are becoming smarter and smarter. For examples, cars are capable of driving themselves like cars driven by humans. Amazon Go is a new kind of store with no checkout required. It automatically detects when products are taken from or returned to the shelves and keeps track of them in a virtual cart. When you're done shopping, you can just leave the store, which makes our life easier.

Although impressive advances in computer vision have been made in recent years, machines are still far behind human beings in many visual recognition tasks. It is remarkable that we can recognize unknown objects after only seeing them a few times, while computer vision systems require a massive amount of images of the same object to acquire them. Humans learn and update knowledge throughout their entire life, but machines suffer from catastrophic forgetting of old knowledge during learning. Moreover, there is a myriad of complex tasks which are impossible for machines to perform. Computer vision systems still have a long way to go before closing the gap between themselves and human vision.

1.1 Self-supervised learning

The performance of deep convolutional neural networks [63] (CNNs) has been improved significantly in recent years due to the development of Graphics Processing Units (GPUs) and massive amounts of labelled training data. A variety of networks including AlexNet [58], VGG [135], GoogLeNet [141], ResNet [39] and DenseNet [43], as well as large scale datsets such as ImageNet [22] and OpenImage [59] have been

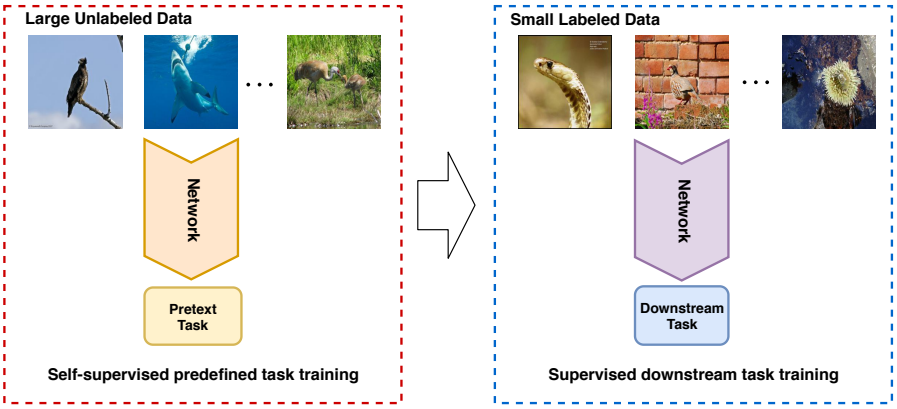

Figure 1.1 – The general pipeline of self-supervised learning. The network is first trained on unlabeled data using some predefined pretext task. No human annotations are required to generate the ground truth for this predefined pretext task. Then, the learned model can be transferred to other downstream tasks with limited amounts of annotated data. The weights of the network are used to initialize the network for downstream supervised tasks.

proposed to advance many computer vision tasks. However, collecting and annotating large scale datasets is time consuming and labor intensive. This is one of the main reasons that *pre-trained ImageNet models* are often used as a starting point for learning other computer vision tasks with relatively small amounts of annotated data. Self-supervised learning is an alternative way to learn meaningful representations *without* any labels.

1.1.1 General self-supervised learning

Self-supervised learning aims to learn useful representations from images or videos without using any human annotations. It can be beneficial for other computer vision tasks even without models pre-trained on massive labeled datasets like ImageNet. To learn from unlabeled data, the basic idea of self-supervised learning is to solve some *predefined pretext task*, where the labels of these pretext tasks are obtainable for free (as shown in Figure 1.1). Some examples of pretext tasks for self-supervision include: image colorization [164], image inpainting [107], image jigsaw puzzle [98] solving, context prediction [26], image rotation [33], and learning to count [99]. Once representations are learned through the self-supervised learning for the pretext task, the learned knowledge can be transferred to downstream tasks

Figure 1.2 – Example input grayscale images and output colorizations from [164]. In order to accurately colorize grayscale images, the network must implicitly learn image semantics useful for many downstream tasks.

(for more details see [21, 49]).

Image colorization [164] is a generation-based method for self-supervision – which are the most straightforward of self-supervised techniques. Given any RGB image, grayscale image is trivial to compute. Thus, the inverse problem of converting from grayscale to color image requires *no human annotation*. This is clearly also an under-constrained problem (as illustrated in Figure 1.2). Learning to colorize with a million color images forces networks to implicitly understand image semantics. Following this line of work, super resolution [65] and image inpainting [107] can also be used to define generation-based pretext tasks for self-supervised learning. Another line of work is based on context understanding: given a large, unlabeled image collection, random pairs of patches from images are extracted and networks are trained to predict the position of the second patch relative to the first [26]. As we can see from Figure 1.3, to predict relative position, the network must understand how objects or images are arranged. Moreover, learning from self-constrained sources is another way of self-supervised learning. These include videos [142], multi-modal sources like RGB and depth [86], images and texts [34, 106] or visual and audio correspondence [4].

Typical Downstream tasks for self-supervised learning include object classification, detection, and semantic segmentation on the PASCAL VOC dataset [28], and object classification on ImageNet [22] and Places [166]. For the ImageNet and Places datasets, after training on self-supervised pretext tasks, a linear classifier is trained on top of frozen convolutional layers using the training split of the respective

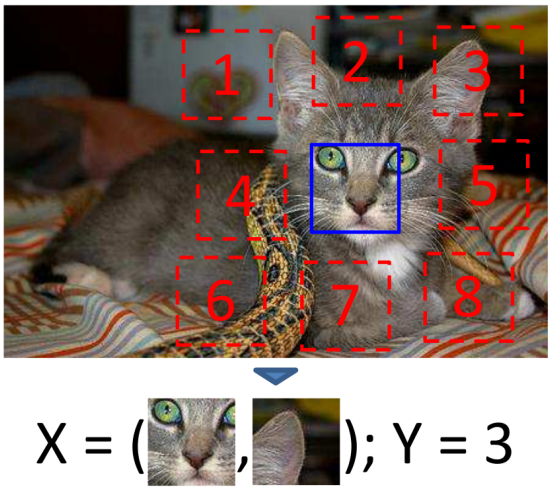

Figure 1.3 – The algorithm receives two patches in one of these eight possible spatial arrangements, without any context, and must then classify which configuration was sampled [26].

datasets. The classification performance on the two datasets are used to evaluate the quality of the learned features. Learning with supervised labels end-to-end is seen as upper bound of self-supervised learning. For detection and segmentation on PASCAL VOC, the pre-trained models from self-supervised learning methods are finetuned on the training set of PASCAL VOC and tested on the validation set.

Because of the effectiveness of self-supervised learning, it has been applied to other tasks as well. For instance, self-supervision can encourage the discriminator of Generative Adversarial Networks (GANs) to learn meaningful feature representations, which in turn improves the quality of the generated images [16]. BERT [24] is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. As a result, the pre-trained BERT model can be finetuned with just one additional output layer to create state-of-the-art models for a wide range of tasks, such as question answering and language inference, without substantial task-specific architecture modifications. Moreover, self-supervised learning has been successfully applied to problems in robotics like navigation [37], grasping [108], and imitation learning [122]. *In this thesis, instead of learning general image representations, we focus on some specific applications, where relatively few annotations are available.*

1.1.2 Learning from rankings

Rankings are generally used in information retrieval [74]. For instance, image retrieval is a computer vision system for browsing, searching, and retrieving images from a large database of digital images. As the results of image retrieval, a list of items are returned in order based on some similarity criterion, which could be content, color, shape or semantic meaning. However, rankings are actually everywhere in our daily life. For instance, without even thinking about rankings, it is not uncommon to compare who is taller in the class when we were young, which food is better when we eat, and what kind of shoes is more suitable for hiking.

Rankings have not been explored for self-supervised learning. It can be a good signal for self-supervision since there is no need for absolute measures, only relative measures between pairs of inputs is enough. Considering the examples mentioned before, when we compare the heights of different children, it is easy to arrive at the correct answer without asking for exact values – we can just ask them to stand next to each other on level ground. When we compare which food is better, we even have no idea how to measure what specific flavors make us preferable to one of them. *In our work, we use learning from rankings as self-supervision task. We show several applications for which ranked sets can be generated without any human labelling.*

1.2 Continual learning

In recent years, deep CNNs have achieved significant improvement on various computer vision tasks. The performance of some tasks (like classification on ImageNet and playing Go) even surpasses that of humans. However, most models are trained on a specific task and tested on the same one. The generalization of these models is often unsatisfying when applying them in new environments. For humans, the ability to learn new knowledge throughout our entire lifetime is taken for granted, while for deep models learning and integrating new knowledge without catastrophic forgetting are difficult to achieve.

Continual learning considers a series of tasks, where artificial neural networks are learned in sequence [18] (illustrated in in Figure 1.4). Normally, catastrophic forgetting of old knowledge happens when adapting to a new task [29]. Most researchers focus on catastrophic forgetting in classification problems, as we do in this thesis, however there are works on other tasks as well such as object detection [132] and few-shot learning [32].

Continual learning methods can be roughly divided into three categories: regularization based approaches, rehearsal-based approaches, and dynamic architecture approaches [103]. Regularization approaches alleviate catastrophic forgetting

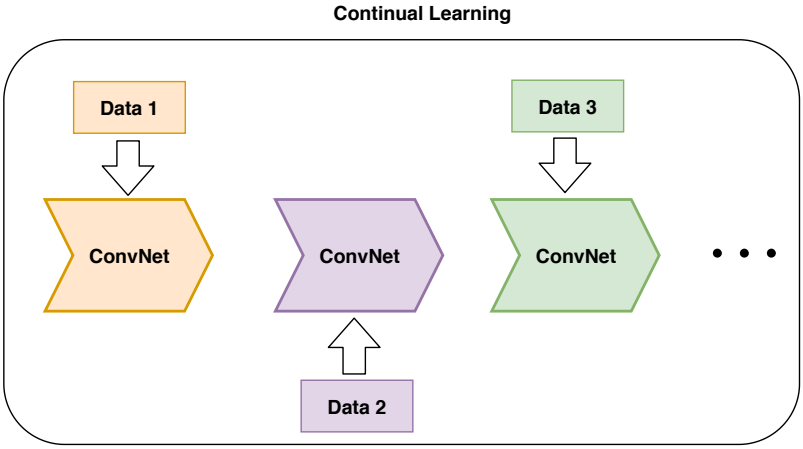

Figure 1.4 – Continual learning. Data comes in sequence and the ConvNet is trained starting from a previous model and is updated with new data. Without any constraints, it suffers from catastrophic forgetting of previous knowledge when updating to new tasks.

by imposing constraints on updating the neural weights. Rehearsal approaches mitigate catastrophic forgetting by keeping samples from previous tasks (called exemplars) or replaying synthetic samples using generative models. Dynamic architecture approaches avoid catastrophic forgetting by dynamically changing the architectural properties when learning a new task, which results in less interference between tasks. *In this thesis, we have explored a regularization-based approach and a rehearsal-based approach to mitigate catastrophic forgetting in continual learning of object classification tasks.*

1.2.1 Regularization-based continual learning

Regularization-based continual learning proposes to include an extra regularization term in the loss function. Knowledge distillation [41] was used to compress networks for fast inference by distilling knowledge from a large network to a small one. The Learning withoug Forgetting (LwF) [71] approach first applied knowledge distillation to continual learning. The idea is mimic the outputs of previous models given the data of new task. Following this line of work, Rannen et al. [113] proposed to train an encoder to capture the crucial features of each task, and then to prevent the reconstructions of the features with these autoencoders from changing. The

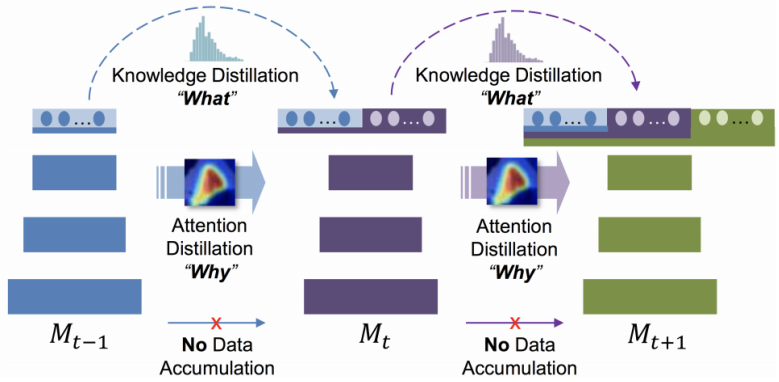

Figure 1.5 – Attention distillation applied to continual learning [25]. Conventional knowledge distillation is about *what* is distilled, while attention distillation is more about *why* it is distilled.

Learning without Memorizing (LwM) [25] approach instead includes an information preserving penalty with attention distillation to overcome forgetting (shown in Figure 1.5).

An additional regularization term could function on the parameters of networks instead of network activations. To avoid forgetting, a straightforward method is to prevent the current network moving away from previous models, normally L_2 distance is used (shown in Figure 1.6). However, this treats each parameter the same. Elastic Weight Consolidation (EWC) [55] was the first approach to penalize parameters with different weights by considering the importance of each one. Following the sequential Bayesian framework for neural networks [85], the posterior probability must contain information about which parameters are important. However, computing the true posterior probability is intractable. Following the work on the Laplace approximation [85], EWC approximates the posterior as a Gaussian distribution with mean given by the parameters of the network and a diagonal precision given by the diagonal of the Fisher Information Matrix (FIM). The FIM is equivalent to the second order derivative of the loss and is assumed to be diagonal in practice in the Laplace approximation – which might be an unrealistic assumption.

Synaptic intelligence (SI) [159] estimates the importance of weights in an online manner and takes into account all previous tasks by accumulating the parameter specific contribution to changes in the total loss. RWalk [14] combines SI and an online version of EWC with a theoretically grounded KL-divergence based perspective.

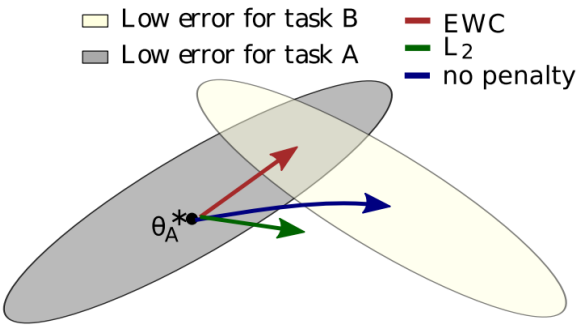

Figure 1.6 – EWC [55] ensures task A is remembered whilst training on task B. Using L_2 norm or without any penalty would result in catastrophic forgetting.

Incremental moment matching (IMM) [67] estimates Gaussian posteriors for each task. Mean-IMM simply averages the parameters of two neural networks, whereas mode-IMM tries to find a maximum of the mixture of Gaussian posteriors. In stead of calculating the gradients of the loss function, MAS [1] obtains the gradients of the squared L_2 norm of the network outputs. Therefore importance of parameters can be measured in unsupervised setting. *In this thesis, we propose to find a reparameterization of the network parameter space for which the FIM is approximately diagonal.*

1.2.2 Rehearsal-based continual learning

Rehearsal-based continual learning can be briefly divided into three categories in terms of the source of samples: real exemplars from previous tasks, synthetically generated images from previous tasks, and synthetically generate features from previous tasks. iCaRL [114] was the first rehearsal-based method. It keeps exemplars from previous tasks such that the mean of features for each class is approximately close to the mean of the entire dataset. In addition, knowledge distillation is used to maintain similar outputs of the previous and current models. Gradient Episodic Memory (GEM) [80] keeps exemplars to focus on minimizing negative backward transfer (catastrophic forgetting) by solving a constrained optimization problem. RWalk [14] compares different ways of keeping exemplars. One drawback of keeping exemplars is the limitation of scalability with increasing number of tasks. VCL [97] fuses online variational inference and recent advances in Monte Carlo variational inference for neural networks with coresets from previous tasks. VCL can success-

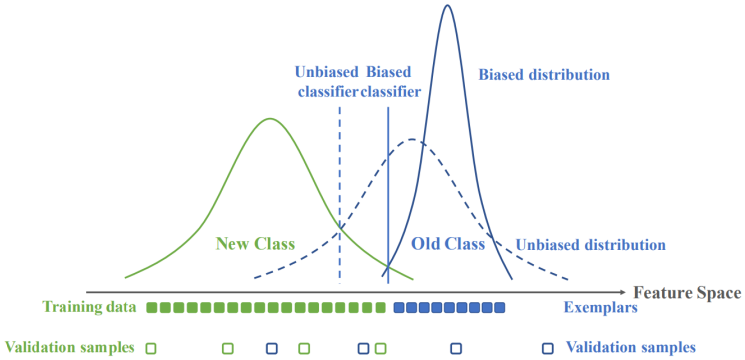

Figure 1.7 – Bias correction in continual learning [151]. Since the number of exemplars from previous tasks is small, they have narrow distributions in the feature space. This causes the learned classifier to prefer new classes over old ones.

fully train both deep discriminative models and deep generative models. Another drawback is the imbalance of training set in continual learning with a fixed memory size. End-to-End Incremental Learning (EEIL) [12] keeps all classifiers at each incremental step and uses them to perform knowledge distillation. Balanced fine-tuning is used to further alleviate the imbalance problem. The authors of [151] found that the last fully connected layer has a strong bias towards the new classes due to the unbalanced training set (shown in Figure 1.7). Therefore, they proposed correcting the bias using a linear model. Similarly, the authors of [42] learned a unified classifier for previous and current tasks combined.

Due to the privacy issues, storing exemplars might not be always allowed. Generative models have achieved huge progress in generating realistic and diverse images. Deep Generative Replay (DGR) [131] was the first method to apply Generative Adversarial Networks (GANs) to continual learning by incorporating synthetic images from previous tasks. DGR trains an extra classifier to assign ground truth of each synthetic sample. Instead, the Memory Replay GAN (MeRGAN) [150] uses conditional GANs to explicitly generate samples for each class given one-hot conditioning vector. The authors of [101] evaluated two variants of neural masking applied to layer activations and to connection weights directly. Furthermore, a dynamic network expansion mechanism was proposed to ensure sufficient model capacity to accommodate for continually incoming tasks. LifelongGAN [160] extended conditioning from one-hot vectors to images. The drawback of these methods applied to continual learning is that the generative models can be very large and complex to

learn, and the quality of generative samples are limited by the ability of GANs.

For feature generation, FearNet [54] learns an encoder and a decoder for features extracted from a pre-trained model. The mean and covariance are computed from low dimensional features and used to replay during continual learning. The drawback of FearNet is that features have to be fixed from pre-trained model. It prevents the model from learning new and better representations for new tasks. Otherwise, the features could drift away from previous tasks, which makes feature replay meaningless and results in catastrophic forgetting as well. *In this thesis, we propose to combine the power of GANs and knowledge distillation to eliminate the drawbacks of generative replay methods.*

1.3 Objectives and approach

In this thesis, in Part 1 we aim to improve networks trained on small domains with self-supervised learning from rankings. Then, in Part 2 we aim to update networks for new domains without catastrophic forgetting.

1.3.1 Self-supervised learning from rankings

The above discussion we motivates our approach to self-supervised learning from rankings. We first expound the theory of self-supervision from rankings, and then apply this theory to two specific applications.

Theory of self-supervised learning from rankings. Learning-to-rank has been used in many information retrieval systems. However, in this work our main objective is not learning to rank, but rather to use rankings as self-supervision to reduce overfitting on tasks with limited amounts of labeled data.

In Chapter 2, we show that rankings can be used for a self-supervised task to address the shortage of labeled data for small domains. Instead of using a traditional pair-wise loss function, we propose a fast Siamese backpropagation algorithm that avoids redundant computations. Furthermore, we show that the ranking task can be exploited as a selection function for active learning that chooses which images should be annotated.

Applications of self-supervised learning from rankings. As we discuss above, most self-supervised learning methods are evaluated on classification, detection and segmentation problems. Improved representations should provide better understanding of classes and objects. In this thesis, we use rankings as a *proxy* task designed for a specific downstream objective instead of learning general representations. The final objective is a super-task of the proxy ranking tasks.

In Chapter 3, we apply self-supervised learning from rankings to two different

regression problems: Image Quality Assessment (IQA) and dense crowd counting. They are both regression problems and available datasets are relatively small – largely due to the high annotation cost associated with these tasks. We have investigated two different ways of integrating the ranking proxy task and the main task. The first approach is to learn first on rankings and then to do finetuning on the main task. The second approach is to jointly learn both ranking and main tasks with a multi-task loss. Our approaches obtain state-of-the-art results on both applications.

1.3.2 Continual learning

Based on the analysis of related work on continual learning, we investigate two methods:

Rotated Elastic Weight Consolidation for Less Catastrophic Forgetting. Several methods use the Fisher Information Matrix (FIM) to approximate the importance of weights in the network for consolidation. However, they assume the covariance matrix of the posterior is diagonal and there are no correlations between the neurons. In Chapter 4, we propose a technique for network reparameterization that approximately diagonalizes the Fisher Information Matrix of the network parameters. This reparameterization takes the form of a factorized rotation of parameter space which, when used in conjunction with Elastic Weight Consolidation (which assumes a diagonal Fisher Information Matrix), leads to significantly better performance on lifelong learning of sequential tasks.

Generative Feature Replay For Task-agnostic Continual Learning. As we discussed in the previous section, keeping exemplars is not scalable to a large number of tasks and results in imbalance between new and old classes. Generating synthetic images with GANs is possible, but complex, and it is difficult to generate high-resolution and realistic images. As an alternative, generating features could be more feasible for continual learning. However, this suffers from fixed representations if we keep the feature extractor fixed or representation drift with no access to low level features if we update feature extractor.

Therefore, in Chapter 5, we propose a hybrid approach where distillation is used in the feature extractor and replay in the classifier via feature generation. Our method addresses the limitations of image generative replay and probability distillation (e.g. *learning without forgetting*), and can naturally aggregate new tasks in a single and well calibrated classifier.

Learning from Rankings Part I

How do we use self-supervision of rankings to benefit for tasks in small domains?

2 Self-supervised Learning to Rank*

2.1 Introduction

Training large deep neural networks requires massive amounts of labeled training data. This fact hampers their application to domains where training data is scarce and the process of collecting new datasets is laborious and/or expensive. Recently, self-supervised learning has received attention because it offers an alternative to collecting labeled datasets. Self-supervised learning is based on the idea of using an auxiliary task (different, but related to the original supervised task) for which data is freely available and no annotation is required. As a consequence, self-supervised learning can be much more scalable. In [26] the self-supervised task is estimating the relative location of patches in images. Training on this task allows the network to learn features discriminative for semantic concepts. Other self-supervised tasks include generating color images from gray scale images and vice versa [62, 164], recovering a whole patch from the surrounding pixels by inpainting [107], and learning from equivalence relations [99].

In this chapter, we investigate the use of ranking as a self-supervised auxiliary task. In particular we consider regression problems in computer vision for which it is easy to obtain ranked data automatically from unlabeled data. By *ranked data* we mean that we know that for some samples the regression output is known to be larger (or smaller) than for some others. In these cases the unlabeled data, converted into ranked subsets of data, can be added in a multi-task sense during training by minimizing an additional ranking loss. The main advantage of our approach is that it allows adding large amounts of unlabeled data to the training dataset, and as a results train better deep neural networks. In addition, we show that the ranked subsets of images can be exploited to performing active learning by identifying which images, when labeled, will result in the largest improvement of performance of the learning algorithm (here a neural network).

In this chapter we study the use of ranking as a self-supervised proxy task to leverage unlabeled data and improve the training of deep networks for regression problems. The contributions of this chapter are:

*This chapter is based on a publication in the Journal of Transactions on Pattern Analysis and Machine Intelligence (TPAMI, 2019) [79].

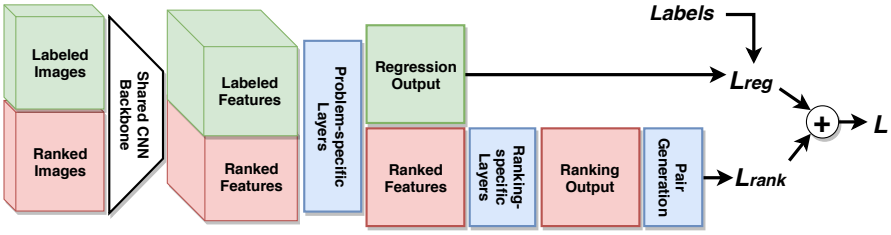

Figure 2.1 – Self-supervised learning to rank. Our architecture is based on a shared CNN backbone (in white) to which we add problem-specific layers (in blue) that end in an output that solves the primary regression task, and layers (also in blue) that end in an output that solves a self-supervised ranking task (see Section 2.3). Self supervision is provided by a pair generation module that is able to generate pairs with known relative ranks.

- We show that ranking tasks can be used as self-supervised proxy tasks, and how this can be exploited to leverage unlabeled data for applications suffering from a shortage of labeled data.

- We propose a method for fast Siamese backpropagation which avoids the redundant computation common to training multi-branch Siamese network architectures. Similar observations have been made others [138, 139] concurrently with our original work [75].

- We show that the ranking task on unlabeled data can be exploited as an active learning strategy to determine which images should be labeled to improve the performance of the network.

This chapter is organized as follows. In the next section we review work from the literature related to our work. In section 2.3 we describe our general learning to rank framework for self-supervised learning.

2.2 Related work

In this section we review work from the literature on learning from rankings and active learning.

2.2.1 Learning from rankings

Several works have studied how to learn *to rank*, and they focus on learning a ranking function from ground-truth rankings [17, 120]. They learn a ranking function from ground-truth rankings by minimizing a ranking loss [17]. This function can then be applied to rank test objects. The authors of [120] adapt the Stochastic Gradient Descent method to perform pairwise learning to rank. This has been successfully applied to large datasets. However, these approaches are very different from ours in which we aim to learn *from* rankings.

In a recent paper [99] a method is proposed where the self-supervised task is to learn to count. The authors propose two proxy tasks – scaling and tiling – which guide self-supervised training. The network learn to count visual primitives in image regions. It is self-supervised by the fact that the number of visual primitives is not expected to change under scaling, and that the sum of all visual primitives in individual tiles should equal the total number of visual primitives in the whole image. Unlike our approach, they do not consider rankings of regions and their counts are typically very low (several image primitives).

2.2.2 Active learning

Active learning [124] is a machine learning procedure that reduces the cost of annotation by actively selecting the best samples to label among the abundantly available unlabeled data. Active learning is well-motivated in many modern machine learning problems where data may be abundant, but labels are scarce or expensive to obtain. Since massive amounts of data are required to train deep neural networks, informative samples are more valuable to annotate instead of annotating randomly picked samples. Active learning has been explored in many applications such as image classification [167], object detection [102] and image segmentation [156].

There are several ways to approach the active learning problem. Uncertainty sampling [68] is the simplest and most commonly used that samples the instance whose prediction is least confident. Margin sampling [118] aims to correct for the shortcomings of uncertainty sampling by examining the difference between second and first most likely labels for candidate samples. A more general strategy considers all prediction probabilities using entropy. Expected Model Change [126] is a metric that measures change in gradient by calculating the length as an expectation over the possible labelings. However, these approaches only consider the uncertainty of instances and ignore the representatives of the underlying distribution. The method proposed in [125] addresses this issue by computing the similarity between the candidate instance and all other samples in the training set. Note that these active learning approaches are more about classification problems, while we are

primarily interested in regression.

Instead of using the above approaches, we measure the informativeness of unlabeled instances via mistakes made by the network on a self-supervised ranking proxy task. Performance on this proxy task is guaranteed to be consistent with the main task, and our hypothesis is that the instances with more mistakes on the proxy task vary more from the training set. Training on such instances should allows us to increase the generalizing capacity of neural networks.

2.3 Learning from rankings

In this section we lay out a general framework for our approach, then describe how we use a Siamese network architecture to learn from rankings. In section 2.3.3 we show how backpropagation for training Siamese networks from ranked samples can be made significantly more efficient. Finally, in section 2.3.4 we show how the ranking proxy task can be used as an active learning algorithm to identify which are the most important images to label first.

2.3.1 Ranking as a self-supervised proxy task

Regression problems consist of finding a mapping function between input variables and a continuous output variable. It is a vital area of research in machine learning, and many important problems in computer vision are regression problems. The mapping function is typically learned from a training dataset of labelled data which consists of pairs of input and output variables. The complexity of the mapping function that can be learned is limited by the number of labeled examples in the training set. In this chapter, we are interested in using deep convolutional neural networks (CNNs) as mapping functions, and images as input data.

In the following sections we will show that ranked data can be used to train networks for regression problems. We are especially interested in domains where the ranked data can be automatically generated from images of the problem domain without requiring any additional human labeling. This allows us to create large dataset of ranked data. We consider regression to be the *principal task* of the network, and we refer to the ranking task as a *self-supervised proxy task*. It is self-supervised since the ranking task is an additional task for which data is freely available and no annotation is required.

2.3.2 Multi-task regression and ranking

In this section, we formalize the problem of training from both labeled and ranked data for regression problems. We consider a regression problem where we have a dataset of observed data in pairs:

$$D = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2) ..., (\mathbf{x}_n, y_n)\}, \tag{2.1}$$

where $\mathbf{x}_i$ are images and $y_i \in \mathbb{R}$. The input images $\mathbf{x}_i$ and target variables y_i are assumed to be related by some unknown function $f(\mathbf{x}_i) = y_i$. The aim is to find a function $\hat{f}(\mathbf{x};\theta)$ with parameters θ that captures (and generalizes) the relationship between input $\mathbf{x}$ and output y. The parameters θ of the regression function $\hat{f}$ are usually fit by minimizing an empirical risk over training examples D, for example the squared Euclidean loss:

$$L_{reg} = \sum_{i=1}^{n} (\hat{f}(\mathbf{x}_i;\theta) - y_i)^2. \tag{2.2}$$

In our formulation, $\hat{f}$ is a deep Convolutional Neural Network (CNN), and minimizing L_{reg} is done with stochastic gradient descent (SGD). We refer to the regression task as the *principal task*, since the final objective is to accurately estimate such a regression.

We also assume that we have a function $h(\cdot, \varphi)$ which we can apply to input images. These functions are special in that the parameter space is ordered and that for any parameters φ_i, φ_j and any image $\mathbf{x}$:

$$\varphi_i \leq \varphi_j \Rightarrow f(h(\mathbf{x};\varphi_i)) \leq f(h(\mathbf{x};\varphi_j)). \tag{2.3}$$

What this means is that we have a *partial order* in the parameter space that induces an ordering in the proxy task space.

We can now apply this function h to generate an auxiliary dataset E consisting of *ranked images*. Importantly, the ordering of the parameter space and the application of h is independent of any labeling of training data. The dataset E could contain images $\mathbf{x}_i$ which are present in dataset D but in addition it could also contain images not in D and for which we have no annotations. We can use any input data $\mathbf{x}_i$ relevant to the domain in order to generate the dataset E of ranked images. Since f respects this ranking condition under application of functions h, we can use the h functions to generate training data for learning $\hat{f}$.

From dataset E we can train a network by minimizing the ranking hinge loss

according to:

$$L_{rank} = \sum_{\substack{\mathbf{z} \in E \\ \varphi_i \leq \varphi_j}} \max(0, \hat{f}(h(\mathbf{z}; \varphi_i); \theta) - \hat{f}(h(\mathbf{z}; \varphi_j); \theta) + \varepsilon) \tag{2.4}$$

where ε is a margin, and $\mathbf{z} \in E$ are (potentially unlabeled) images transformed into *ranked* images after applying $h(\cdot, \varphi_i)$ and $h(\cdot, \varphi_j)$ for $\varphi_i \leq \varphi_j$. Training a network with Eq. (2.4) yields a network which can *rank* images, and we will refer to the task of ranking as the *self-supervised proxy task*.

The most common approach to minimizing losses like L_{rank} uses a Siamese network [19], which is a network with two identical branches connected to a loss module. The two branches share weights during training. Pairs of images and labels are the input of the network, yielding two outputs which are passed to the loss. The gradients of the loss function with respect to all model parameters are computed by backpropagation and updated by the stochastic gradient method (e.g. SGD). For problems where both labeled data (like in dataset D) and ranked data (like in dataset E) are present, we can optimize the network using *both* sources of data using a *multi-task loss*:

$$L = L_{reg} + \lambda L_{rank} \tag{2.5}$$

where λ is a tradeoff parameter that balances the relative weight of the losses. It is important to note here that we consider the same function $\hat{f}(\cdot; \theta)$ for the regression and the ranking loss. In practice this means that the three networks, one for regression, and the two networks used in the Siamese network, have the same architecture and share their parameters.

2.3.3 Efficient Siamese backpropagation

One drawback of Siamese networks is redundant computation. Consider all possible image pairs constructed from three images. In a standard implementation all three images are passed twice through the network, because they each appear in two pairs. Since both branches of the Siamese network are identical, we are essentially doing twice the work necessary since any image need only be passed *once* through the network. It is exactly this idea that we exploit to render backpropagation more efficient for Siamese network training. In fact, nothing prevents us from considering *all* possible pairs in a mini-batch, with hardly any additional computation. We add a new layer to the network that generates all possible pairs in a mini-batch at the end of the network right before computing the loss. This eliminates the problem of pair selection and boosts efficiency. At the end of this section we discuss how

our approach compares to works [138, 139] published concurrently with ours and observed similar efficiency gains.

To appreciate the speed-up of efficient Siamese backpropagation consider the following. If we have one reference image distorted from which we have generated n ranked images using h, then for a traditional implementation of the Siamese network we would have to pass a total of $n^2 - n$ images through the network – which is twice the number of pairs you can generate with n images. Instead we propose to pass all images only *once* and consider all possible pairs only in the loss computation layer. This reduces computation to just n images passed through the network. Therefore, in this case the speed-up is equal to: $\frac{n^2-n}{n} = n-1$. In the best scenario n is equal to the number of images in the mini-batch, and hence the speed-up of this method would be in the order of the mini-batch size. Due to the high correlation among the set of all pairs in a mini-batch, we expect the final speedup in convergence to be lower.

To simplify notation in the following, assume we have an image $\mathbf{z}$ and two transformation parameters $\varphi_i \leq \varphi_j$. Letting $\hat{y}_i = \hat{f}(h(\mathbf{z}, \varphi_i); \theta)$, the contribution to the ranking loss L_{rank} of these two images can be written as:

$$g(\hat{y}_i, \hat{y}_j) = \max(0, \hat{y}_i - \hat{y}_j + \varepsilon), \tag{2.6}$$

The gradient of this term from L_{rank} with respect to the model parameters θ is:

$$\nabla_\theta g = \frac{\partial g(\hat{y}_i, \hat{y}_j)}{\partial \hat{y}_i} \nabla_\theta \hat{y}_i + \frac{\partial g(\hat{y}_i, \hat{y}_j)}{\partial \hat{y}_j} \nabla_\theta \hat{y}_j. \tag{2.7}$$

This gradient of g above is a sum since the model parameters are shared between both branches of the Siamese network and $\hat{y}_i$ and $\hat{y}_j$ are computed using exactly the same parameters.

Considering all pairs in a mini-batch of size M, the loss L_{rank} from Eq. (2.4) can then be written as:

$$L_{rank} = \sum_{i=1}^{M} \sum_{j>i}^{M} g(\hat{y}_i, \hat{y}_j). \tag{2.8}$$

The gradient of the mini-batch loss with respect to parameter θ can then be written as:

$$\nabla_\theta L_{rank} = \sum_{i=1}^{M} \sum_{j>i}^{M} \frac{\partial g(\hat{y}_i, \hat{y}_j)}{\partial \hat{y}_i} \nabla_\theta \hat{y}_i + \frac{\partial g(\hat{y}_i, \hat{y}_j)}{\partial \hat{y}_j} \nabla_\theta \hat{y}_j. \tag{2.9}$$

We can now express the gradient of the loss function of the mini-batch in matrix

form as:

$$\nabla_\theta L_{rank} = \begin{bmatrix} \nabla_\theta \hat{y}_1 & \nabla_\theta \hat{y}_2 & \dots & \nabla_\theta \hat{y}_M \end{bmatrix} P \mathbf{1}_M, \tag{2.10}$$

where $\mathbf{1}_M$ is the vector of all ones of length M. For a standard single-branch network, we would average the gradients for all batch samples to obtain the gradient of the mini-batch. This is equivalent to setting P to the identity matrix in Eq. (2.10) above. For Siamese networks where we consider all pairs in the mini-batch we obtain Eq. (2.9) by setting P to:

$$P = \begin{bmatrix} 0 & \frac{\partial g(\hat{y}_1,\hat{y}_2)}{\partial \hat{y}_1} & \dots & \frac{\partial g(\hat{y}_1,\hat{y}_M)}{\partial \hat{y}_1} \\ \frac{\partial g(\hat{y}_1,\hat{y}_2)}{\partial \hat{y}_2} & 0 & \dots & \frac{\partial g(\hat{y}_2,\hat{y}_M)}{\partial \hat{y}_2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial g(\hat{y}_1,\hat{y}_M)}{\partial \hat{y}_M} & \dots & \dots & 0 \end{bmatrix}. \tag{2.11}$$

For the ranking hinge loss we can write:

$$P = \begin{bmatrix} 0 & a_{12} & \cdots & a_{1M} \\ a_{21} & 0 & \cdots & a_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ a_{M1} & \cdots & a_{M(M-1)} & 0 \end{bmatrix}, \tag{2.12}$$

where

$$a_{ij} = \begin{cases} 0 & \text{if } l_{ij}\left(\hat{y}_i - \hat{y}_j\right) + \varepsilon \leq 0 \\ l_{ij} & \text{otherwise} \end{cases} \tag{2.13}$$

and l_{ij} is used to indicate the ordering of the φ parameters used to generate the M images to which $\hat{f}$ was applied to derive outputs $\hat{y}_i$ and $\hat{y}_j$:

$$l_{ij} = \begin{cases} 1 & \text{if } \varphi_i \leq \varphi_j \\ -1 & \text{if } \varphi_i > \varphi_j \\ 0 & \text{if } \varphi_i \text{ and } \varphi_j \text{ are not comparable.} \end{cases} \tag{2.14}$$

The above analysis works for different parameter settings φ on the same source image. When considering multiple source images in a mini-batch, only different parameter settings φ on the same image are considered comparable when defining l_{ij}.

Generally, the complexity of training Siamese networks is ameliorated via different pair sampling techniques. In [134], the authors propose a hard positive and hard negative mining strategy to forward propagate a set of pairs and sample the highest-loss pairs for backpropagation. However, hard mining comes with a high computational cost (they report an increase of up to 80% of total computation cost). In [119] the authors propose semi-hard pair selection, arguing that selecting hardest pairs can lead to bad local minima. In [146] the authors take a batch of pairs as input and choose the four hardest negative samples within the minibatch. In parallel with our work on fast backpropagation for Siamese networks [75] several similar methods have been developed [138, 139]. To solve for bad local optima, [139] optimize a smooth upper bound loss function. This is implemented by considering all possible pairs in a mini-batch after forwarding the images through the network. In [138], the N-pair Loss is proposed to compute pairwise similarity within the batch to construct $N-1$ negative examples instead of one in triplet loss. Hard negative class mining is applied to improve convergence speed. Both these works, like ours, prevent the redundant computation which is introduced by the multiple branches in the Siamese network.

2.3.4 Active learning from rankings

The objective of active learning is to reduce the cost of labeling by prioritizing the most informative samples for labeling first. Rather than labeling randomly selected examples from a pool of unlabeled data, active learning methods analyze unlabeled data with the goal of identifying images considered difficult and that therefore are more valuable if labeled. Especially for deep networks, which require many examples, as well as for applications for which labeling is very costly, active learning is an important and active area of research.

We show here how the self-supervised proxy task can be leveraged for active learning. For this purpose we define a function $C(\mathbf{x}_i)$ which estimates the certainty of the current network on a specific image $\mathbf{x}_i$. This estimate allows us to order the available unlabeled dataset according to certainty. Labeling the images for which the algorithm is *uncertain* is then expected to yield larger improvement in performance of the network than just randomly adding images.

The certainty function C is defined as:

$$C(\mathbf{z};\theta) = \frac{1}{K}\sum_{\varphi_i \leq \varphi_j} T(\hat{f}(h(\mathbf{z};\varphi_i);\theta) < \hat{f}(h(\mathbf{z};\varphi_j);\theta)) \tag{2.15}$$

where $T(s) = 1$ if predicate s is true or false otherwise, and K is the number of sampled parameter pairs (φ_i, φ_j) applied to each image $\mathbf{z}$. As described in the

Algorithm 1 : Active learning loop.

Input:

$D = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$: labeled samples
$E = \mathbf{z}_1, \ldots, \mathbf{z}_m$: unlabeled samples

Require:

θ_0 : initial network parameters
T : number of active learning cycles
S : number of images added in each cycle

for $t = 1 : T$

$\theta_t \leftarrow$ **train**(D, θ_{t-1}) // Train network on D.
$D^S \leftarrow$ **label**(E, S, θ_t) // Evaluate Eq. (2.15) for all samples in E,
// and Label S least confident samples.
$D \leftarrow D \bigcup D^S$ // Update labeled set.
$E \leftarrow E \setminus D^S$ // Update unlabeled set.

end for

previous section, the pairs of parameters $\varphi_i \leq \varphi_j$ can be used to generate ranked images via the function h. Again, these pairs can be automatically computed and no annotation is required.

By design, we know that $0 \leq C(\mathbf{x}_i) \leq 1$. Given an unlabeled dataset and a current state of the trained network $\hat{f}(\cdot, \hat{\theta})$, we perform the proxy task for a total of K times on each image and then compute C. We then label images starting from low confidence to high confidence. The process is detailed in Algorithm 1.

2.4 Conclusions

In this chapter we explored ranking as a self-supervised proxy task for regression problems. For many regression problems the collection of supervised data is an expensive and laborious process. We showed, however, that there exist some problems for which it is easy to obtain ranked image sets, and that these ranked image sets can be exploited to improve the training of the network. In addition, we proposed a method for fast backpropagation for the ranking loss. This method removes the redundant computation which is introduced by the multiple branches of Siamese networks, and instead uses a single branch after which all possible pairs of the minibatch are combined (rather than just a selection of pairs).

3 Applications to Image Quality Assessment and Crowd Counting*

3.1 Introduction

We consider two specific computer vision regression problems to demonstrate the advantages of learning from ranked data: Image Quality Assessment (IQA) and crowd counting.

The first regression problem we consider is No-Reference Image Quality Assesment (NR-IQA), where the task is to predict the perceptual quality of images without using the undistorted image (also called the reference image). This research field has also seen large improvements in recent years due to the advent of Convolutional Neural Networks (CNNs) [51, 52, 72]. The main problems these papers had to address is the lack of large datasets for IQA. However, the annotation process for IQA image datasets requires multiple human annotations for every image, and thus the collection process is extremely labor-intensive and costly. As a result, most available IQA datasets are too small to be effective for training CNNs. We show how to automatically generate rankings for the task of IQA, and how these rankings can be used to improve the training of CNNs for NR-IQA.

The second regression problem we consider is crowd counting. Crowd counting is a daunting problem because of perspective distortion, clutter, occlusion, non-uniform distribution of people, complex illumination, scale variation, and a host of other scene-incidental imaging conditions. Techniques for crowd counting have also seen improvement recently due to the use of CNNs. These recent approaches include scale-aware regression models [100], multi-column CNNs [165], and switching networks [6]. As with most CNN architectures, however, these person counting and crowd density estimation techniques are highly data-driven. Even modestly deep architectures for visual recognition require massive amounts of labeled training data for learning. For person counting, the labeling burden is even more onerous than usual. Training data for person counting requires that each individual person be meticulously labeled in training images. It is for this reason that person counting and crowd density estimation datasets tend to have only a few hundred images available for training. As a consequence, the ability to train

*This chapter is based on publications in the international Conference of Computer Vision (ICCV, 2017) [77] and Computer Vision and Pattern Recognition (CVPR, 2018) [78].

these sophisticated CNN-based models suffers. We show how ranked sets of images can be generated for the task of crowd counting, and how these can be exploited to train deep networks.

In this chapter we demonstrate the advantages of the above contributes on two applications: Image Quality Assessment (IQA) and crowd counting. On both tasks we show how to effectively leverage unlabeled data and that this significantly improves performance over the state-of-the-art.

This chapter is organized as follows. In the next section we review work from the literature related to our work. In section 3.3 we describe how to automatically generate rankings for image quality assessment. In section 3.4 we show how the proposed framework can be applied to the problem of crowd counting. In sections 3.5.1 and 3.5.2 we give extensive experimental evaluations for IQA and crowd counting, respectively. We conclude in section 3.6 with a discussion of our contributions and some indications of potential future research lines.

3.2 Related work

In this section we review work from the literature on image quality assessment and crowd counting.

3.2.1 Image quality assessment

We briefly review the IQA literature related to our approach. We focus on recent deep learning based methods for distortion-generic, No-reference IQA since it is more generally applicable than the other IQA research lines.

In recent years several works have used deep learning for NR-IQA [7, 51, 52]. One of the main drawbacks of deep networks is the need for large labeled datasets, which are currently not available for NR-IQA research. To address this problem Kang et al. [51] consider small 32×32 patches rather than images, thereby greatly augmenting the number of training examples. The authors of [9, 52] follow the same pipeline. In [52] the authors design a multi-task CNN to learn the type of distortions and image quality simultaneously. Bianco at al. [7] propose to use a pre-trained network to mitigate the lack of training data. They extract features from a pre-trained model fine-tuned on an IQA dataset. These features are then used to train an SVR model to map features to IQA scores.

There are other works which, like us, apply rankings in the context of NR-IQA. Gao et al. [31] generate pairs in the dataset itself by using the ground truth scores with a threshold and combine different hand-crafted features to represent image pairs from the IQA dataset. Xu et al. [155] propose training a specific model for

each distortion type in a multi-task learning model. Instead of using the ground truth in the dataset, they learn a ranking function. The most relevant work is from Ma et al. [83], which was published concurrently with our work on NR-IQA [75]. Like us, they use learning-to-rank to deal with the extremely limited ground truth data for training. However, they generate pairing data by using other Full-reference IQA methods with a threshold like in [31], while our approach does not require other methods and operates in a self-supervised manner. Another difference is that we use a knowledge transfer technique for further fine-tuning on the target dataset with the same network, which can be also learned in a multi-task, end-to-end way, while they train an independent regression model based on the feature representations to perform the prediction. In addition to these differences, we show results on a larger number of distortion types instead of only working on the four main distortions [83].

In our work, we propose a radically different approach to address the lack of training data: we use a large number of automatically generated rankings of image quality to train a deep network. This allows us to train much deeper and wider networks than other methods in NR-IQA which train directly on absolute IQA data.

3.2.2 Crowd counting

We focus on deep learning methods for crowd counting in still images. For a more complete review of the literature on crowd counting, including classical approaches, we refer the reader to [137].

As introduced in the review of [137], CNN-based approaches can be classified into different categories based on the properties of the CNN. Basic CNNs incorporate only basic CNN layers in their networks. The approaches in [30, 145] use the AlexNet network [58] to map from crowd scene patches to global number of people by changing the output of AlexNet from 1000 to 1. The resulting network can be trained end-to-end. Due to the large variations of density in different images, recent methods have focused on scale-awareness. The method proposed in [165] trains a multi-column based architecture (MCNN) to capture the different densities by using different sizes of kernels in the network. Similarly, the authors of [100] propose the Hydra-CNN architecture that takes different resolutions of patches as inputs and has multiple output layers (heads) which are combined in the end. Most recently, in [6] the authors propose a switching CNN that can select an optimal head instead of combining the information from all network heads. Finally, context-aware models are networks that can learn from the context of images. In [30, 136] the authors propose to classify images or patches into one of five classes: very high density, high density, medium density, low density and very low density. However, the definition of these five classes varies across datasets and must be carefully

chosen using knowledge of the statistics of each dataset.

Although CNN-based methods have achieved great success in crowd counting, due to lack of labeled data it is still challenging to train deep CNNs without overfitting. The authors of [161] propose to learn density map and global counting in an alternating sequence to obtain better local optima. The method in [50] uses side information like ground-truth camera angle and height to help the network to learn. However, this side information is expensive to obtain and is not available in most existing crowd counting datasets.

There is some interesting recent work on CNNs for crowd counting. CSRNet [70] consists of two components: a convolutional neural network as 2D feature extractor and a dilated CNN for estimating a density map to yield larger receptive fields and to replace pooling operations. DecideNet [73] starts by estimating the crowd density using detection and regression separately. It then assesses the reliability of these two estimates with an attention module. Shen et al. [129] propose using a U-net structure to generate a high quality density map with an adversarial loss. Shi et al. [130] formulate a single ConvNet as ensemble learning. Marsden et al. [89] adapt object counting models to new visual domains like cell counting and penguins counting. Both works [11, 112] propose generating a high resolution density map. Ierees et al. [47] propose solving the problems of counting, density map estimation and localization simultaneously. Laradji et al. [61] propose a detection-based method that does not need to estimate the size and shape of the objects.

In this chapter, we show how a large number of unlabeled crowd data can improve the training of crowd counting networks. We automatically generate rankings from the unlabeled images, which are used during the training process in the self-supervised proxy task.

For some regression problems it can be easy to obtain a *ranked dataset*. Such a dataset contains relative information between pairs of input examples, describing which of the two is larger. For image quality assessment (IQA) it is easy to generate ranked images by applying different levels of distortions to an image. As an example, given a reference image we can apply various levels of Gaussian blur. The set of images which is thus generated can be easily *ranked* because we do know that adding Gaussian blur (or any other distortion) *always* deteriorates the quality score. Note that in such a set of ranked images we do not have any absolute IQA scores for any images – but we do know for any pair of images *which is of higher quality*. See Fig. 3.1 (bottom) for an illustration of this.

For the crowd counting problem, we can obtain ranked sets of images by comparing parts of the same image which are contained within each other: an image which is contained by another image will contain the same number or fewer persons than the larger image. This fact can be used to generate a large dataset of ranked images from unlabeled crowd images. See Fig. 3.3 (bottom) for an illustration of

this process for crowd counting.

In the following sections we will show that ranked data can be used to train networks for regression problems. We are especially interested in domains where the ranked data can be automatically generated from images of the problem domain without requiring any additional human labeling. This allows us to create large dataset of ranked data. We consider regression to be the *principal task* of the network, and we refer to the ranking task as a *self-supervised proxy task*. It is self-supervised since the ranking task is an additional task for which data is freely available and no annotation is required.

3.3 Image quality assessment by learning to rank

In this section we apply the framework proposed in the previous section to the problem of Image Quality Assessment (IQA) [147]. IQA aims to automatically predict the perceptual quality of images. IQA estimates should be highly correlated with quality assessments made by a range of very many human evaluators (commonly referred to as the Mean Opinion Score (MOS) [109, 128]). We focus here on no-reference IQA (NR-IQA), which refers to the case where the undistorted image (called reference image) is not available during the quality assessment.

The application of CNNs to IQA has resulted in significant improvements compared to previous hand-crafted approaches [51, 52, 72]. These methods had to train their networks on the small available datasets for IQA. Further improvements would be expected if larger datasets were made available. However, the annotation of IQA images is a labor intensive task, which requires multiple human annotators for every image. It is therefore an interesting application field to evaluate our framework, since by adding ranking as a proxy task, we are able to add unlabeled data during the training process.

We first discuss existing datasets for IQA and how to automatically generate IQA rankings. Then we introduce the network which we train for the IQA task, together with some application-specific training choices.

3.3.1 IQA datasets

We perform experiments on two standard IQA datasets:

- **LIVE** [127]: Consists of 808 images generated from 29 original images by distorting them with five types of distortion: Gaussian blur (GB), Gaussian noise (GN), JPEG compression (JPEG), JPEG2000 compression (JP2K) and fast fading (FF). The ground-truth Mean Opinion Score for each image is in the range [0, 100] and is estimated using annotations by 161 human annotators.

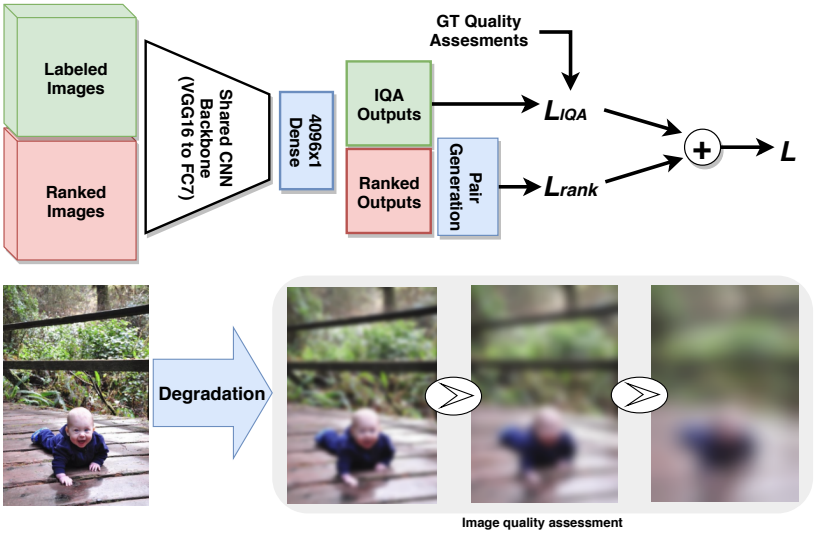

Figure 3.1 – Network architecture and ranked pair generation for IQA. **Top**: our network for no-reference IQA uses a VGG16 network pretrained on ImageNet. We decapitate the network and replace the original head with a new fully-connected layer generating a single output. **Bottom**: pairs with known ranking are generated by distorting images with standard, parametric distortions. Increasing the distortion level guarantees that the images are of progressively worse quality.

- **TID2013** [109]: Consists of 25 reference images with 3000 distorted images from 24 different distortion types at 5 degradation levels. Mean Opinion Scores are in the range [0, 9]. Distortion types include a range of noise, compression, and transmission artifacts. See the original publication for the list of specific distortion types.

3.3.2 Generating ranked image sets for IQA

The IQA datasets LIVE and TID2013 are derived from only 29 and 25 original images, respectively. Deep networks trained on so few original images will almost certainly have difficulty generalizing to other images. Here we show how to automatically generate rankings from arbitrary images which can be added during the training process (i.e. we show what function h in Eq. (2.4) we use to generate IQA rankings).

Using an arbitrary set of images, we can synthetically generate deformations of these over a range of distortion intensities. As an example, consider Fig. 3.1 (bottom) where we have distorted an image with increasing levels of Gaussian noise. Although we do not know the absolute IQA score, we do know that images which are *more* distorted should have a *lower* score than images which are less distorted. This fact can be used to generate huge datasets of ranked images. We can use a large variety of images and distortions to construct the datasets of ranked images.

We use the Waterloo [84] dataset, which consists of 4,744 high quality natural images carefully chosen from the Internet, to generate a large ranking dataset. To test on the LIVE database, we generate four types of distortions which are widely used and common: Gaussian Blur (GB), Gaussian Noise (GN), JPEG, and JP2K. We apply these distortions at five levels, resulting in a total of 20 distortions for all images in the Waterloo dataset. To test on TID2013, we generate 17 out of a total of 24 distortions at five levels, yielding a total of 85 distortions per image (see the Appendix for more details). There were seven distortions which we could not generate, however we found that adding the other distortions also resulted into improvements for the distortions for which we could not generate rankings.

3.3.3 IQA network

For the IQA problem we have a training dataset with images $\mathbf{x}_i$ and ground truth image quality y_i. After generating the ranked image dataset we can train an IQA network. As a CNN backbone we use the VGG-16 network, only changing the last layer to output a single IQA score. With respect to the basic architecture given in Fig. 2.1, here we specify the architecture we use for IQA estimation as shown in Fig. 3.1 (top). For the labeled data we use the Euclidean distance between the

prediction of the network $\hat{y}_i$ and the ground truth as the regression loss:

$$L_{IQA}(y_i, \hat{y}_i) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2, \tag{3.1}$$

and will optimize this loss jointly with the ranking loss L_{rank} as proposed in Eq. (2.5).

We randomly sample sub-images from the original high resolution images. We do this instead of scaling to avoid introducing distortions caused by interpolation or filtering. The size of sampled images is determined by each network. However, the large size of the input images is important since input sub-images should be at least 1/3 of the original images in order to capture context information. This is a serious limitation of the patch sampling approach [51, 52] that samples very small 32×32 patches from the original images. In our experiments, we sample 224×224 pixel images from original images varying from 300 to 700 pixels.

3.4 Crowd counting by learning to rank

In this section, we adapt the proposed framework to the task of crowd counting. Perspective distortion, clutter, occlusion, non-uniform distribution of people, complex illumination, scale variation, and a host of other scene-incidental imaging conditions render person counting and crowd density estimation in unconstrained images an daunting problem.

Techniques for crowd counting have been recently improved using Convolutional Neural Networks (CNNs). As with most CNN architectures, however, these person counting and crowd density estimation techniques are highly data-driven. For person counting, the labeling burden is even more onerous than usual. Training data for person counting requires that each individual person be meticulously labeled in training images. It is for this reason that person counting and crowd density estimation datasets tend to have only a few hundred images available for training. As a consequence, crowd counting networks are expected to benefit from additional unlabeled data which is used to train a ranking proxy task.

3.4.1 Crowd counting datasets

We use two standard benchmark crowd counting datasets:

- **UCF_CC_50** [46]: This dataset contains 50 annotated images of different resolutions, illuminations and scenes. The variation of densities is very large among images from 94 to 4543 persons with an average of 1280 persons per image.
- **ShanghaiTech** [165]: Consists of 1198 images with 330,165 annotated heads. This

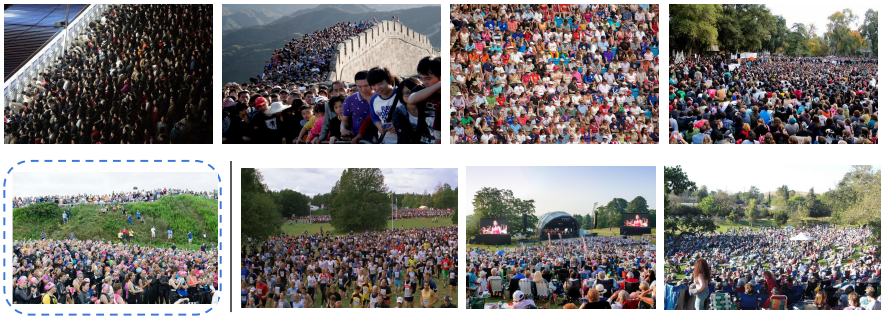

Figure 3.2 – Example images from the retrieved crowd scene dataset. (top) Representative images using key words as query. (bottom) Representative images using training image as query image (the query image is depicted on the left).

dataset includes two parts: 482 images in Part_A which are randomly crawled from the Internet, and 716 images in Part_B which are taken from busy streets. Both parts are further divided into training and evaluation sets. The training and test of Part_A has 300 and 182 images, respectively, whereas that of Part_B has 400 and 316 images, respectively.

Ground truth annotations for crowd counting typically consist of a set of coordinates which indicate the 'center' (typically head center of a person). To convert this data to crowd density maps we place a Gaussian with standard deviation of 15 pixels and sum these for all persons in the scene to obtain y_i. This is a standard procedure and is also used in [100, 165].

3.4.2 Generating ranked image sets for counting

Here we show how to automatically generate the rankings from unlabeled crowd counting images. The main idea is based on the observation that all patches contained within a larger patch must have a fewer or equal number of persons than the larger one (see Fig. 3.3). This observation allows us to collect large datasets of crowd images with known relative ranks. Rather than having to painstakingly annotate each person we are only required to verify if the image contains a crowd. Given a crowd image we extract ranked patches according to Algorithm 2.

To collect a large dataset of crowd images from the Internet we use two different approaches:

- **Keyword query:** We collect a crowd scene dataset from Google Images by us-

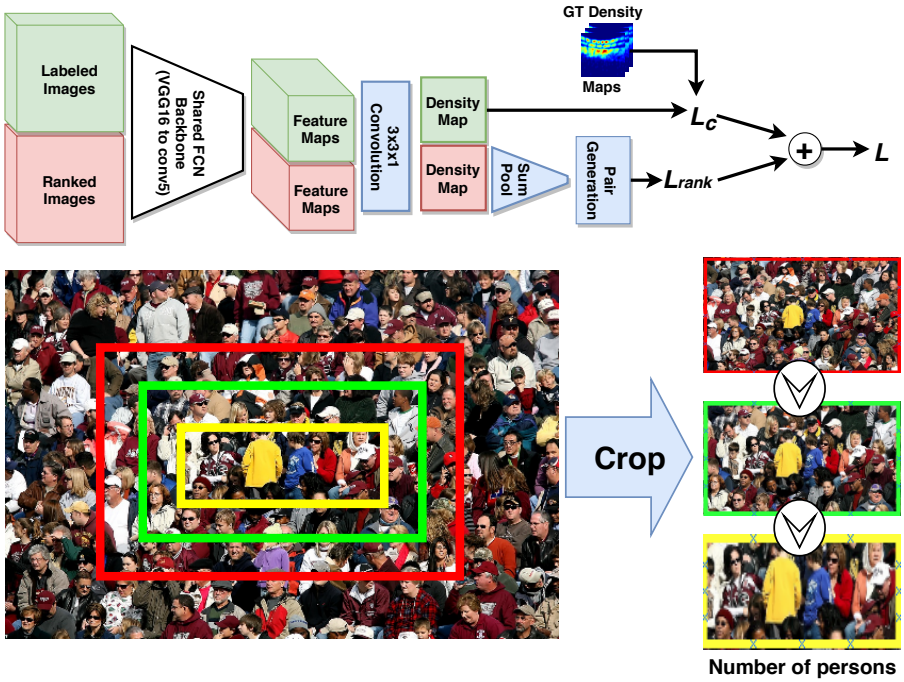

Figure 3.3 – Network architecture and ranked pair generation for crowd counting. **Top**: our counting network uses a VGG16 network truncated at the fifth convolutional layer (before maxpooling). To this network we add a $3 \times 3 \times 1$ convolutional layer with stride 1 which should estimate local crowd density. A sum pooling layer is added to the ranking channel of the network to arrive at a scalar value whose relative rank is known. **Bottom**: image pairs with known relative ranks are generated by selectively cropping unlabeled crowd images so that successive crops are entirely contained in previous ones.

Algorithm 2 : Algorithm to generate ranked datasets.

Input:	A crowd scene image, number of patches k and scale factor s.
Step 1:	Choose an anchor point randomly from the anchor region. The anchor region is defined to be $1/r$ the size of the original image, centered at the original image center, and with the same aspect ratio as the original image.
Step 2:	Find the largest square patch centered at the anchor point and contained within the image boundaries.
Step 3:	Crop $k-1$ additional square patches, reducing size iteratively by a scale factor s. Keep all patches centered at anchor point.
Step 4:	Resize all k patches to input size of network.
Output:	A list of patches ordered according to the number of persons in the patch.

ing different key words: *Crowded, Demonstration, Train station, Mall, Studio, Beach*, all of which have high likelihood of containing a crowd scene. Then we delete images not relevant to our problem by simple visual inspection. In the end, we collected 1,180 high resolution crowd scene images, which is about 24x the size of the UCF_CC_50 dataset, 2.5x the size of ShanghaiTech Part_A, and 2x the size of ShanghaiTech Part_B. Note that *no other annotation of images is performed.* Example images from this dataset are given in Fig. 3.2 (top row).

- **Query-by-example image retrieval:** For each annotated benchmark dataset, we collect an unlabeled dataset using the training images as queries with the *Google Images* visual image search engine. We choose the first ten similar images and remove irrelevant ones. For UCF_CC_50 we collected 256 images, for ShanghaiTech Part_A 2229 images, and for ShanghaiTech Part_B 3819 images. An example of images returned for a specific query image is given in Fig. 3.2 (bottom row).

3.4.3 Crowd density estimation network

We first explain the network architecture which is trained on available crowd counting datasets with ground truth annotations (see Fig. 3.3) . This network regresses to a crowd density image which indicates the number of persons per pixel (examples of such maps are given in Fig. 3.5). A summation of all values in such a crowd density image gives an estimate of the number of people in the scene. In the experimental section we consider this network as the baseline method to which we compare.

Our baseline network is derived from VGG-16 [135] pre-trained on ImageNet. VGG-16 consists of 13 convolutional layers followed by three fully connected layers. We adapt the network to regress to person density maps by removing its three fully connected layers and the last max-pooling layer (pool5) to prevent further reduction of spatial resolution. In their place we add a single convolutional layer (a $3 \times 3 \times 512$ filter with stride 1 and zero padding to maintain same size) which directly regresses to the crowd density map. As the counting loss L_c we use the Euclidean distance between the estimated and ground truth density maps, as given by:

$$L_c = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 \tag{3.2}$$

where N is the number of images in a training batch, y_i is ground truth person density map of the i-th image in the batch, and the prediction from the network as $\hat{y}_i$.

To further improve the performance of our baseline network, we introduce multi-scale sampling from the available labeled datasets during training. Instead of using the whole image as an input, we randomly sample square patches of varying size (from 56 to 448 pixels). In the experimental section we verify that this multi-scale sampling is important for good performance. Since we are processing patches rather than images we use $\hat{y}_i$ to refer to the estimate of patch i from now on. The importance of multi-scale processing of crowd data was also noted in [8].

Finally, we add a summation layer to the network. This summation layer takes as an input the estimated density map and sums it to a single number (the estimate of the number of persons in the image). This output is used to compute the ranking loss (see Eq. (2.4)) for the unlabeled images in the ranked dataset. With respect to the basic architecture in Fig. 2.1, we use a sum polling layer as a ranking specific layer as shown in Fig. 3.3 (top).

3.5 Experimental results

In this section we report on an extensive set of experiments performed to evaluate the effectiveness of learning from rankings for Image Quality Assessment in section 3.5.1 and crowd counting in section 3.5.2. We use the Caffe [48] deep learning framework in all the experiments.

3.5.1 Image quality assessment (IQA)

We performed a range of experiments designed to evaluate the performance of our approach with respect to baselines and the state-of-the-art in IQA. These experiments make use of standard datasets (for benchmark evaluation) and an additional dataset used for generating ranked distorted image pairs as explained in section 3.3.

We use Stochastic Gradient Descent (SGD) with an initial learning rate of 1e-4 for efficient Siamese network training and 1e-6 for fine-tuning. Training rates are decreased by a factor of 0.1 every 10,000 iterations for a total of 50,000 iterations. For both training phases we use ℓ_2 weight decay (weight 5e-4). For multi-task training, we use $\lambda = 1$ in Eq. (2.5) with a learning rate of 1e-6 on the LIVE dataset and 1e-5 on TID2013. We found $\lambda = 1$ to work well in initial experiments, but cross validating λ on held-out data is expected to improve results for specific datasets. During training we sample a single subimage from each training image per epoch. When testing, we randomly sample 30 sub-images from the original images, as suggested in [7], and pass all the trained models. The average of all outputs of the sub-regions is the final score for each distorted image.

Two evaluation metrics are traditionally used to evaluate the performance of IQA algorithms: the Linear Correlation Coefficient (LCC) and the Spearman Rank Order Correlation Coefficient (SROCC). LCC is a measure of the linear correlation between the ground truth and the predicted quality scores. Given N distorted images, the ground truth of i-th image is denoted by y_i, and the predicted score from the network is $\hat{y}_i$. The LCC is computed as:

$$LCC = \frac{\sum_{i=1}^{N}(y_i - \overline{y})(\hat{y}_i - \overline{\hat{y}})}{\sqrt{\sum_{i}^{N}(y_i - \overline{y})^2}\sqrt{\sum_{i}^{N}(\hat{y}_i - \overline{\hat{y}})^2}} \tag{3.3}$$

where $\overline{y}$ and $\overline{\hat{y}}$ are the means of the ground truth and predicted quality scores, respectively.

Given N distorted images, the SROCC is computed as:

$$SROCC = 1 - \frac{6\sum_{i=1}^{N}\left(v_i - p_i\right)^2}{N\left(N^2 - 1\right)}, \tag{3.4}$$

where v_i is the *rank* of the ground-truth IQA score y_i in the ground-truth scores, and p_i is the *rank* of $\hat{y}_i$ in the output scores for all N images. The SROCC measures the monotonic relationship between ground-truth and estimated IQA.

Table 3.1 – Ablation study on the entire TID2013 database.

Method	LCC	SROCC
Baseline	0.663	0.612
RankIQA	0.566	0.623
RankIQA+FT (Random)	0.775	0.738
RankIQA+FT (Hard)	0.782	0.748
RankIQA+FT (Ours)	0.799	0.780
MT-RankIQA (Random)	0.802	0.770
MT-RankIQA (Hard)	0.810	0.779
MT-RankIQA (Ours)	**0.827**	**0.806**

Table 3.2 – Performance evaluation (SROCC) on the entire TID2013 database.

Method	#01	#02	#03	#04	#05	#06	#07	#08	#09	#10	#11	#12	#13
BLIINDS-II [117]	0.714	0.728	0.825	0.358	0.852	0.664	0.780	0.852	0.754	0.808	0.862	0.251	0.755
BRISQUE [93]	0.630	0.424	0.727	0.321	0.775	0.669	0.592	0.845	0.553	0.742	0.799	0.301	0.672
CORNIA-10K [158]	0.341	-0.196	0.689	0.184	0.607	-0.014	0.673	**0.896**	0.787	0.875	0.911	0.310	0.625
HOSA [154]	0.853	0.625	0.782	0.368	**0.905**	0.775	0.810	0.892	0.870	0.893	**0.932**	**0.747**	0.701
RankIQA	**0.891**	**0.799**	**0.911**	**0.644**	0.873	**0.869**	**0.910**	0.835	**0.894**	**0.902**	0.923	0.579	0.431
RankIQA+FT	0.667	0.620	0.821	0.365	0.760	0.736	0.783	0.809	0.767	0.866	0.878	0.704	0.810
MT-RankIQA	0.780	0.658	0.882	0.424	0.839	0.762	0.852	0.861	0.799	0.879	0.909	0.744	**0.824**

Method	#14	#15	#16	#17	#18	#19	#20	#21	#22	#23	#24	ALL
BLIINDS-II [117]	0.081	0.371	0.159	-0.082	0.109	0.699	0.222	0.451	0.815	0.568	0.856	0.550
BRISQUE [93]	0.175	0.184	0.155	0.125	0.032	0.560	0.282	0.680	0.804	0.715	0.800	0.562
CORNIA-10K [158]	0.161	0.096	0.008	0.423	-0.055	0.259	0.606	0.555	0.592	0.759	0.903	0.651
HOSA [154]	0.199	0.327	0.233	0.294	0.119	0.782	0.532	0.835	**0.855**	**0.801**	**0.905**	0.728
RankIQA	0.463	**0.693**	**0.321**	**0.657**	0.622	**0.845**	0.609	**0.891**	0.788	0.727	0.768	0.623
RankIQA+FT	**0.512**	0.622	0.268	0.613	0.662	0.619	0.644	0.800	0.779	0.629	0.859	0.780
MT-RankIQA	0.458	0.658	0.198	0.554	**0.669**	0.689	**0.760**	0.882	0.742	0.645	0.900	**0.806**

Ablation study

In this experiment, we evaluate the effectiveness of using rankings to estimate image quality. We compare our multi-task approach with different baselines: fine-tuning the VGG-16 network initialized from ImageNet to obtain the mapping from images to their predicted scores (which we call Baseline in our experiments), and two other baselines: VGG-16 (initialized pre-trained ImageNet weights) trained on ranking data (called RankIQA), and RankIQA approach fine-tuned on TID2013 after training using ranked pairs of images (called RankIQA+FT). In order to evaluate the effectiveness of our sampling method in the multi-task setting, we also report the accuracy for multi-task training (called MT-RankIQA) with different sampling methods: standard random pair sampling, and a hard-negative mining method similar to [134]. For standard random pair sampling we randomly choose 36 pairs

for each mini-batch from the training sets. For the hard negative mining strategy we start from 36 pairs in a mini-batch, and gradually increase the number of hard pairs every 5000 iterations. For our method we pass 72 images in each mini-batch. With these settings the computational costs for all three methods are equal, since at each iteration 72 images are passed through the network.

We follow the experimental protocol used in HOSA [154]. The entire TID2013 database including all types of distortions is divided into 80% training images and 20% testing images according to the reference images and their distorted versions. The results are shown in Table 3.1, where ALL means testing all distortions together. All the experiments are performed 10 times and the average SROCC is reported

From Table 3.1, we can draw several conclusions. First, it is hard to obtain good results by training a deep network directly on IQA data. This is seen in the Baseline results and is due to the scarcity of training data. Second, competitive results are obtained using RankIQA without access to the ground truth of IQA dataset during training the ranking network, which strongly demonstrates the effectiveness of training on ranking data. The RankIQA-trained network alone does not provide accurate IQA scores (since it has never seen any) but does yield high correlation with the IQA scores as measured by SROCC. After fine-tuning on the TID2013 database, we considerably improve performance for all sampling methods: with random sampling we improve on the baseline by 11%, while our efficient sampling method further outperforms random sampling by 2.4% in terms of LCC (similar conclusions can be drawn from the SROCC results.

Finally, in Table 3.1 we also compare the three optimization methods for multi-task training: random pair sampling, hard negative mining, and our proposed efficient Siamese backpropagation. We see that our efficient back-propagation strategy with multi-task setting obtains the best results, further improving the accuracy by 2.8% on LCC and 2.6% on SROCC with respect to RankIQA+FT. This clearly shows the benefits of the proposed backpropagation scheme. In the next section, we use MT-RankIQA to refer to our method trained with the multi-task loss using our efficient Siamese backpropagation method.

To demonstrate the ability of our approach to generalize to unseen distortions, we trained our multi-task approach with different numbers of synthetic distortions as auxiliary data combined with all labeled data in the TID2013 dataset. All results are the average of three runs. As shown in Table 3.3, adding more distortions results in increased overall performance on the test set. Note also that adding specific distortions not only consistently improves accuracy on seen distortions consistently, but also on unseen ones.

Table 3.3 – Generalization to unseen distortions. Results of MT-RankIQA with different numbers of synthetic distortions as auxiliary data combined with all labeled data in the TID2013 dataset. Results show that also on the unseen distortions a significant gain in performance is obtained.

Overall LCC accuracy		Average gain	
		Seen distortions	Unseen distortions
Baseline	0.687	–	–
7 distortions	0.803	+0.102	+0.016
11 distortions	0.817	+0.102	+0.032
15 distortions	0.823	+0.104	+0.067
All	0.829	+0.109	+0.075

Comparison with the state-of-the-art

We compare the performance of our method with state-of-the-art Full-reference IQA (FR-IQA) and NR-IQA methods on both TID2013 and LIVE dataset.

Evaluation on TID2013. Table 3.2 includes results of state-of-the-art methods. We see that for several very challenging distortions (14 to 18), where all other methods fail, we obtain satisfactory results. For individual distortions, there is a huge gap between RankIQA and other methods on most distortions. The state-of-the-art method HOSA performs slightly better than our methods on 6 out of 24 distortions. For all distortions, RankIQA+FT achieves about 5% higher than HOSA, and about 3% more is gained by using multi-task training. Our methods also perform well on distortions for which were unable to generate rankings. This indicates that different distortions share some common representation and training the network jointly on all distortions also improves results for the distortions for which we did not generate rankings.

Evaluation on LIVE. As done in [51, 163], we randomly split the reference images and their distorted version from LIVE into 80% training and 20% testing sample and compute the average LCC and SROCC scores on the testing set after training to convergence. This process is repeated ten times and the results are averaged. These results are shown in Table 3.4. For fair comparison with the state-of-the-art, we train our ranking model on four distortions (all but FF), but we fine-tune our model on all five distortions in the LIVE dataset to compute ALL. As shown in Table 3.4 our approach improves by 0.4% and 1% in LCC and SROCC, respectively, the best NR-IQA results reported on ALL distortions. This indicates that our method outperforms existing work including the current state-of-the-art NR-IQA method

Table 3.4 – LCC (above) and SROCC (below) evaluation on LIVE dataset. We divide approaches into full-reference (FR-) and no-reference (NR-) IQA.

	LCC	JP2K	JPEG	GN	GB	FF	ALL
FR-IQA	PSNR	0.873	0.876	0.926	0.779	0.87	0.856
	SSIM [148]	0.921	0.955	0.982	0.893	0.939	0.906
	FSIM [162]	0.91	0.985	0.976	0.978	0.912	0.96
	DCNN [72]	–	–	–	–	–	0.977
NR-IQA	DIVINE [96]	0.922	0.921	0.988	0.923	0.888	0.917
	BLIINDS-II [117]	0.935	0.968	0.98	0.938	0.896	0.93
	BRISQUE [93]	0.923	0.973	0.985	0.951	0.903	0.942
	CORNIA [158]	0.951	0.965	0.987	0.968	0.917	0.935
	CNN [51]	0.953	0.981	0.984	0.953	0.933	0.953
	SOM [163]	0.952	0.961	0.991	0.974	0.954	0.962
	DNN [9]	–	–	–	–	–	0.972
	MT-RankIQA	**0.972**	**0.978**	**0.988**	**0.982**	**0.971**	**0.976**

	SROCC	JP2K	JPEG	GN	BLUR	FF	ALL
FR-IQA	PSNR	0.87	0.885	0.942	0.763	0.874	0.866
	SSIM [148]	0.939	0.946	0.964	0.907	0.941	0.913
	FSIM [72]	0.97	0.981	0.967	0.972	0.949	0.964
	DCNN [72]	–	–	–	–	–	0.975
NR-IQA	DIVINE [96]	0.913	0.91	0.984	0.921	0.863	0.916
	BLIINDS-II [117]	0.929	0.942	0.969	0.923	0.889	0.931
	BRISQUE [93]	0.914	0.965	0.979	0.951	0.887	0.94
	CORNIA [158]	0.943	0.955	0.976	0.969	0.906	0.942
	CNN [51]	0.952	0.977	0.978	0.962	0.908	0.956
	SOM [163]	0.947	0.952	0.984	0.976	0.937	0.964
	DNN [9]	–	–	–	–	–	0.960
	MT-RankIQA	**0.971**	**0.978**	**0.985**	**0.979**	**0.969**	**0.973**

SOM [163] and DNN [9], and even achieves competitive results as state-of-the-art FR-IQA method DCNN [72] which, being a full-reference approach, has the benefit of having access to the high-quality original reference image.

Active learning for IQA

We demonstrate the effectiveness of active learning on the IQA problem. As a dataset we consider all 24 distortions for each of 19 reference images from TID2013, yielding a dataset of 456 image-distortion pairs. Each image is distorted for each distortion at five distortion levels. During active learning we aim to select which images with what particular distortion are expected to most improve performance. We add all five distortion levels of the selected image with the particular distortion to the labeled pool. The active learning loop starts with 10% of this data labeled; hence the labeled samples D is 10% and the examples E consist of the remaining 90% of the data without labels. We then perform $T = 9$ active learning cycles. In each cycle $S = 10\%$ images with a particular distortion are added incrementally, using the full training set from the previous iteration to estimate informativeness for the remaining unlabeled data. We use $K = 100$ in the experiment, and compare results to a baseline of randomly adding 10% additional labeled samples at each step.

Results for active learning are given in Fig. 3.4. It is clear that our active learning algorithm obtains results superior to random selection. When adding an additional 10% of data (using a total of 20% labeled data) we achieve similar accuracy as adding an additional 40% data (using a total of 50% labeled data) with random selection, thereby reducing the labeling cost by 75% compared to the baseline.

3.5.2 Crowd counting

Here we report on a range of experiments evaluating our approach with respect to baselines and the state-of-the-art methods for crowd counting.

We use SGD with a batch size of 25 for both ranking and counting, and thus a batch size of 50 for multi-task training. For the ranking plus fine-tuning method, the learning rate is 1e-6 for both ranking and fine-tuning. For multi-task training, we found that $\lambda = 100$ yielded good results on all datasets. Cross validating λ on held-out data is expected to improve results for specific datasets. Learning rates are decreased by a factor of 0.1 every 5,000 iterations for a total of 10K iterations. For both training phases we use ℓ_2 weight decay with a weight of 5e-4. During training we sample one sub-image from each training image per epoch. We perform down-sampling of three scales and up-sampling of one scale on the UCF_CC_50 dataset and only up-sampling of one scale on the ShanghaiTech dataset. The number

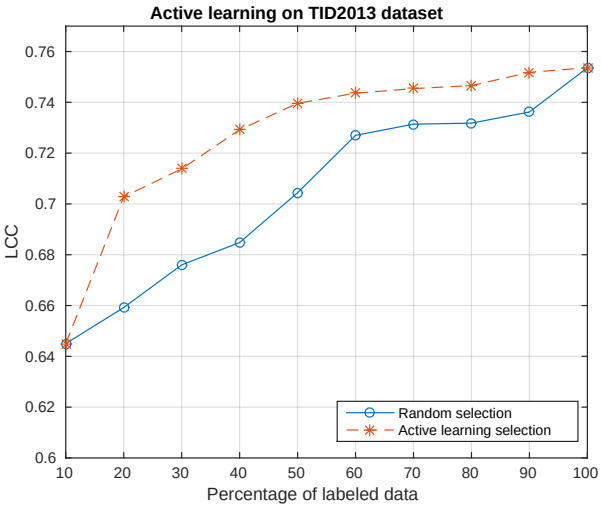

Figure 3.4 – Active learning results on TID 2013. We plot LCC as a function of the percentage of labeled data used from the training set.

of ranked crops $k = 5$, the scale factor $s = 0.75$, and the anchor region $r = 8$ (see Algorithm 2).

Following existing work, we use the mean absolute error (MAE) and the mean squared error (MSE) to evaluate different methods. These are defined as follows:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| \tag{3.5}$$

$$MSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \tag{3.6}$$

where N is the number of test images, y_i is the ground truth number of persons in the ith image, and $\hat{y}_i$ is number of persons predicted by the network in the ith image.

Ablation study

We begin with an ablation study on the UCF_CC_50 dataset. The aim is to evaluate the relative gain of the proposed improvements and to evaluate the use of a ranking

Table 3.5 – Ablation study on UCF_CC_50 with five-fold cross validation.

Method	Split 1	Split 2	Split 3	Split 4	Split 5	Ave MAE
Basic CNN	701.41	394.52	497.57	263.56	415.23	454.45
+ Pre-trained model	570.01	350.63	334.89	184.79	202.41	328.54
+ multi-scale	532.85	307.43	266.75	216.96	216.35	308.06
Ranking+FT	552.68	375.38	241.28	211.66	247.70	325.73
Multi-task (Random)	462.71	345.31	218.71	226.44	210.19	292.67
Multi-task (Hard)	460.35	343.91	208.23	221.75	205.57	287.96
Multi-task (Ours)	443.68	340.31	196.76	218.48	199.54	**279.60**

loss against the baseline. The ranked images in this experiment are generated from the Keyword dataset. The results are summarized in Table 3.5. We can immediately observe the benefit of using a pre-trained ImageNet model in crowd counting, with a significant drop in MAE of around 28% compared to the model trained from scratch. By using both multi-scale data augmentation and starting from a pre-trained model, another improvement of around 6% is obtained.

The Ranking+FT method performs worse than directly fine-tuning from a pre-trained ImageNet model. This is probably caused by the poorly-defined nature of the self-supervised task. To optimize this task the network could decide to count anything, e.g. 'hats', 'trees', or 'people with red shirts', or even just 'edges' – all of which would satisfy the ranking constraints that are imposed.

Next, we compare the three sampling strategies for combining the ranking and counting losses for multi-task training. When using multi-task training with random pair sampling, the average MAE is reduced by about 15 points. Hard mining obtains about 5 points average MAE less than random sampling. However, our efficient back-propagation approach reduces the MAE further to 279.6. This shows that by *jointly* learning both the self-supervised and crowd counting tasks, the self-supervised task is forced to focus on counting persons. Given its superior results, we consider only the "Multi-task" with efficient back-propagation approach for the remainder of the experiments.

Comparison with the state-of-the-art

Evaluation on the UCF_CC_50 dataset. A five-fold cross-validation was performed for evaluating the methods. Results are shown in Table 3.6. Our multi-task training method using the unlabeled Keyword Dataset reduces the MAE from 291.0 to 279.6, which is comparable to ACSCP [129] which was published at the

Table 3.6 – MAE and MSE error on the UCF_CC_50 dataset.

Method	MAE	MSE
Idrees et al. [46]	419.5	541.6
Cross-scene [161]	467.0	498.5
MCNN [165]	377.6	509.1
Onoro et al. [100]	333.7	425.2
Walach et al. [144]	364.4	341.4
Switching-CNN [6]	318.1	439.2
CP-CNN [136]	295.8	**320.9**
ACSCP [129]	291.0	404.6
CSRNet [70]	266.1	397.5
ic-CNN [112]	**260.9**	365.5
Multi-task (Query-by-example)	291.5	397.6
Multi-task (Keyword)	279.6	408.1

same time as our original work. Our approach performs slightly worse than the state-of-the-art methods CSRNet [70] and ic-CNN [112] (also published around the same time as our original work), but in general our model has fewer parameters than CSRNet and a simpler inference procedure compared to ic-CNN. However, the MSE of our method on UCF_CC_50 dataset is worse than the state-of-the-art methods [112, 136, 144], but achieves competitive results compared to [70, 129]. This indicates that our method and also [129] work better in general but have more extreme outliers. Compared to training on the Keyword dataset, learning from the Query-by-example dataset is slightly worse, which might be because most images from UCF_CC_50 are black and white with low resolution, which often does not lead to satisfactory query results. An example of prediction in UCF_CC_50 using our network is shown in Fig. 3.5.

Evaluation on the ShanghaiTech dataset. Looking at Table 3.7, we can draw conclusions similar to those on UCF_CC_50. We see here that using the Query-by-example Dataset further improves by about 2% on ShanghaiTech – especially for Part_A, where our approach surpasses the state-of-the-art method [136], but is still slightly worse than CSRNet [70] and ic-CNN [112]. An example of prediction by our network on ShanghaiTech is given in Fig. 3.5. For comparison, we also provide the results of our baseline method (including fine-tuning from a pre-trained model and multi-scale data augmentation) on this dataset: $MAE = 77.7$ and $MSE = 115.9$ on Part A, and $MAE = 14.7$ and $MSE = 24.7$ on Part B.

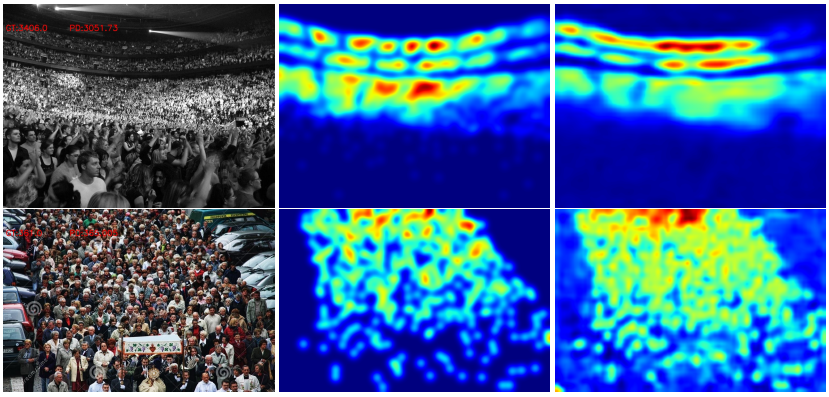

Figure 3.5 – Examples of predicted density maps for the UCF_CC_50 (Top row, true count: 3406 prediction: 3052) and ShanghaiTech datasets (Bottom row, true count: 361 prediction: 365). Left column: crowd image. Middle column: ground truth. Right column: prediction.

Evaluation on the WorldExpo'10 dataset The WorldExpo'10 dataset [161] consists of 3980 frames of size 576× 720 from 1132 video sequences captured by 108 surveillance cameras. The dataset is split into training set with 103 scenes and test set consisting of 5 different scenes. Regions of interest (ROIs) are provided for the whole dataset, which are used as a mask during testing. We consider training directly on the only ground truth labels as a baseline, and for our multi-task approach, when we test on one specific scene, the rest of scenes in the test set are used to generate the ranked image set. We compare our multi-task training to the baseline and other state-of-the-art methods in Table 3.8. It is clear that our multi-task training approach outperforms the baseline in all five cases and achieves comparable results compared to other methods in terms of MAE.

Evaluation on the UCSD dataset The UCSD dataset [13] has 2000 frames with a region of interest (ROI) varying from 11 to 46 persons per image. The resolution of each frame is fixed and small (238 × 158), so we change the input of network to 112 × 112 by removing all layers after pool4 in VGG-16. Thus the output retains the same 1/16 of input size. We follow the same settings as [6], using frames between 600 and 1400 as training set, and the rest as test set. In order to train our multi-task approach and compare fairly to other methods, we generate ranked sets using the frames from 1 to 600 and test on the frames from 1401 to 2000 (and vice versa). We do not collect additional unlabeled data since the training and test set are from the same camera. Results are shown in Table 3.9. Our baseline performs similarly

Table 3.7 – MAE and MSE error on ShanghaiTech.

Method	Part A		Part B	
	MAE	MSE	MAE	MSE
Cross-scene [161]	181.8	277.7	32.0	49.8
MCNN [165]	110.2	173.2	26.4	41.3
Switching-CNN [6]	90.4	135.0	21.6	33.4
CP-CNN [136]	73.6	106.4	20.1	30.1
ACSCP [129]	75.7	**102.7**	17.2	27.4
CSRNet [70]	**68.2**	115.0	**10.6**	**16.0**
ic-CNN [112]	68.5	116.2	10.7	**16.0**
Ours: Multi-task (Query-by-example)	72.0	106.6	14.4	23.8
Ours: Multi-task (Keyword)	73.6	112.0	13.7	21.4

Table 3.8 – MAE results on the WorldExpo'10 dataset.

Method	Scene 1	Scene 2	Scene 3	Scene 4	Scene 5	Ave MAE
MCNN [165]	3.4	20.6	12.9	13.0	8.1	11.6
Switching-CNN [6]	4.4	15.7	10.0	11.0	5.9	9.4
CP-CNN [136]	2.9	14.7	10.5	10.4	5.8	8.9
ACSCP [129]	2.8	14.05	9.6	8.1	2.9	**7.5**
CSRNet [70]	2.9	11.5	8.6	16.6	3.4	8.86
ic-CNN [112]	17.0	12.3	9.2	8.1	4.7	10.3
Baseline	5.0	22.0	14.3	15.7	5.3	12.5
Ours	3.8	17.5	13.8	12.7	5.2	10.5

to the state-of-the-art methods [6, 70], but slightly worse than ACSCP [129] and MCNN [165]. Our multi-task approach reduces the baseline MAE from 1.60 to 1.17. Our multi-task approach works on less dense datasets like UCSD because, though the baseline might make incorrect predictions on patches with the almost same number of headcounts, our learning-to-rank branch can constrain it and ensure correct ranking predictions.

Evaluation on the UCF-QNRF dataset. UCF-QNRF [47] is a very challenging dataset consisting of 1,535 images with average of 815 people per image. The average resolution of images is much larger compared to other datasets, with images up to 6,000×9,000 pixels. We resize all images to have a maximum dimension of 1024 pixels without changing the aspect ratio. The Keyword Dataset is used as unlabeled data to for the multi-task learning. Results are shown in Table 3.10, which indicate

Table 3.9 – MAE and MSE error on the UCSD dataset.

Method	MAE	MSE
Cross-scene [161]	1.60	3.31
MCNN [165]	1.07	**1.35**
Switching-CNN [6]	1.62	2.10
ACSCP [129]	**1.04**	**1.35**
CSRNet [70]	1.16	1.47
Baseline	1.60	2.13
Ours	1.17	1.55

Table 3.10 – MAE and MSE error on the UCF-QNRF dataset.

Method	MAE	MSE
MCNN [165]	277	426
Switching-CNN [6]	228	445
CompositionLoss [47]	132	**191**
Baseline	137	228
Ours	**124**	196

that our technique consistently improves counting performance – even on datasets like UCF-QNRF with significantly more labeled training samples. Compared to our baseline, the MAE is reduced from 137 to 124 and MSE is down to 196. Our method outperforms all other methods including CompositionLoss [47] in MAE and performs slightly worse in MSE.

Evaluation on transfer learning. As proposed in [165], to demonstrate the generalization of the learned model, we test our method in the transfer learning setting by using Part_A of the ShanghaiTech dataset as the source domain and using UCF_CC_50 dataset as the target domain. The model trained on Part_A of ShanghaiTech is used to predict the crowd scene images from UCF_CC_50 dataset, and the results can be seen in Table 3.11. Using only counting information improves the MAE by 12% compared to reported results in [165]. By combining both ranking and counting datasets, the MAE decreases from 349.5 to 337.6, and MSE decreases from 475.7 to 434.3. In conclusion, these results show that our method significantly outperforms the only other work reporting results on the task of cross-dataset crowd counting.

Table 3.11 – Transfer learning across datasets. Models were trained on Part_A of ShanghaiTech and tested on UCF_CC_50.

Method	MAE	MSE
MCNN [165]	397.6	624.1
Counting only	349.5	475.7
Multi-task	337.6	434.3

Active learning for crowd counting

We use the Shanghai Part_A dataset to evaluate our active learning approach on Crowd counting. We use 10% of the training set as the initially labeled training samples D (and E the remaining 90%), and set S to add 10% of the training images in each of the $T = 9$ active learning cycles (see Algorithm 1). We use $K = 100$ in this experiment to evaluate ranking certainty. Again we compare to the baseline of randomly selecting images from E. The result is shown in Fig. 3.6. Our approach performs consistently better than random selection in terms of MAE. We obtain similar results with 20% of the training data as random approach does with 40% – reducing the labeling effort by 50%. These results also show that performance saturates after 60% and no further improvement is obtained by adding the last 40% images considered least useful by the active learning algorithm. The results clearly show that our active learning method correctly identifies the images, which when labeled, contribute most to an improved crowd counting network.

3.6 Conclusions

We applied the learning-to-rank framework to two regression problems: Image Quality Assessment and crowd counting. In the case of Image Quality Assessment, the ranked data sets are formed by adding increasing levels of distortions to images. For crowd counting, ranked sets are formed by comparing image crops which are contained within each other: a smaller image contained in another larger one will contain the same number or fewer persons than the larger image. Experimental results show that for both applications results improve significantly when adding unlabeled data for the ranking task. In addition, we have shown that the best results are obtained when using our efficient backpropagation method in a multi-task setting.

We have also shown that the proxy task can be used as an informativeness measure for unlabeled images. The number of errors made on the proxy task can

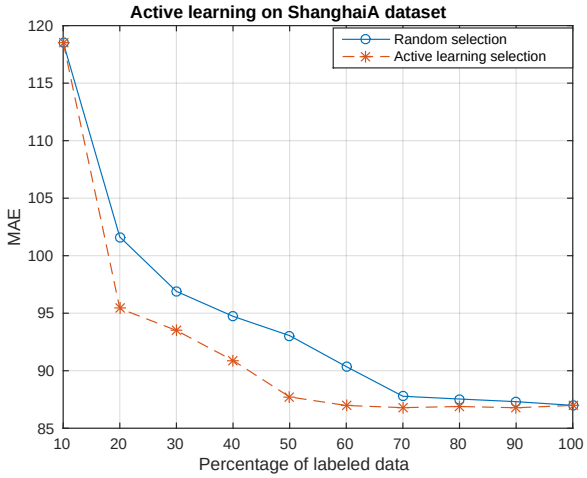

Figure 3.6 – Active learning results on Shanghai A. MAE is plotted as a function of the percentage of labeled data from the training set.

be used to drive an active learning algorithm to select the best images to label from a pool of unlabeled ones. These images, once added to the training set, will most improve the performance of the network. Experimental results show that, for both IQA and crowd counting, this method can reduce the labeling effort by a large margin.

Continual Learning Part II

How do we learn new domains without forgetting old domains?

4 Rotated Elastic Weight Consolidation for Less Catastrophic Forgetting*

4.1 Introduction

Neural networks are very effective models for a variety of computer vision tasks. In general, during training these networks are presented with examples from all tasks they are expected to perform. In a lifelong learning setting, however, learning is considered as a sequence of tasks to be learned [133], which is more similar to how biological systems learn in the real world. In this case networks are presented with *groups* of tasks, and at any given moment the network has access to training data from only *one* group. The main problem which systems face in such settings is *catastrophic forgetting*: while adapting network weights to new tasks, the network forgets the previously learned ones [91].

There are roughly two main approaches to lifelong learning (which has seen increased interest in recent years). The first group of methods stores a small subset of training data from previously learned tasks. These stored exemplars are then used during training of new tasks to avoid forgetting the previous ones [80, 114]. The second type of algorithm instead avoids storing any training data from previously learned tasks. A number of algorithms in this class are based on Elastic Weight Consolidation (EWC) [55, 67, 159], which includes a regularization term that forces parameters of the network to remain close to the parameters of the network trained for the previous tasks. In a similar vein, Learning Without Forgetting (LWF) [71] regularizes *predictions* rather than weights. Aljundi et al. [2] learns a representation for each task and use a set of gating autoencoders to decide which expert to use at testing time.

EWC is an elegant approach to selective regularization of network parameters when switching tasks. It uses the Fisher Information Matrix (FIM) to identify directions in feature space critical to performing already learned tasks (and as a consequence also those directions in which the parameters may move freely without forgetting learned tasks). However, EWC has the drawback that it assumes the Fisher Information Matrix to be diagonal – a condition that is almost never true.

In this chapter we specifically address this diagonal assumption made by the

*This chapter is based on a publication in the International Conference of Pattern Recognition (ICPR, 2018) [76]

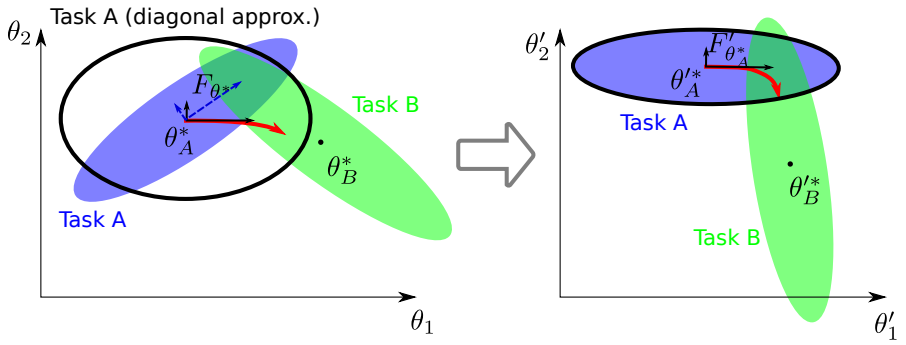

Figure 4.1 – Sequential learning in parameter space, illustrating the optimal model parameters for tasks A and B and regions with low forgetting. The red line indicates the learning path for task B from the previously learned solution for task A. **Left**: the diagonal approximation (black ellipse) of the Fisher Information Matrix can be poor and steer EWC in directions that will forget task A. **Right**: after a suitable reparameterization (i.e., a rotation) the diagonal approximation is better and EWC can avoid forgetting task A.

EWC algorithm. If the FIM is not diagonal, EWC can fail to prevent the network from straying away from "good parameter space" (see Fig. 4.1, left). Our method is based on *rotating* the parameter space of the neural network in such a way that the output of the forward pass is unchanged, but the FIM computed from gradients during the backward pass is approximately diagonal (see Fig. 4.1, right). The result is that EWC in this rotated parameter space is significantly more effective at preventing catastrophic forgetting in sequential task learning problems. An extensive experimental evaluation on a variety of sequential learning tasks shows that our approach significantly outperforms standard elastic weight consolidation.

The rest of this chapter is organized as follows. In the next section we review the EWC learning algorithm and in Section 4.4 we describe the approach to rotating parameter space in order to better satisfy the diagonal requirement of EWC. We give an extensive experimental evaluation of the proposed approach in Section 4.5 and conclude with a discussion of our contribution in Section 4.6.

4.2 Related work

An intuitive approach to avoiding forgetting for sequential task learning is to retain a portion of training data for each task. The iCaRL method of Rebuffi et al. [114]

is based on retaining exemplars which are rehearsed during the training of new tasks. Their approach also includes a method for exemplar herding which ensures that the class mean remains close even when the number of exemplars per class is dynamically varied. A recent paper proposed the Gradient Episodic Memory model [80]. Its main feature is an episodic memory storing a subset of the observed examples. However, these examples are not directly rehearsed but instead are used to define inequality constraints on the loss that ensure that it does not increase with respect to previous tasks. A cooperative dual model architecture consisting of a deep generative model (from which training data can be sampled) and a task solving model is proposed in [131].

Learning without Forgetting in [71] works without retaining training data from previous tasks by regularizing the network output on new task data to not stray far from its original output. Similarly, the lifelong learning approach described in [113] identifies informative features using per-task autoencoders and then regularizes the network to preserve these features in its internal, shared-task feature representation when training on a new task. Elastic Weight Consolidation (EWC) [55, 67, 159] includes a regularization term that forces important parameters of the network to remain close to the parameters of the network trained for the previous tasks. The authors of [123] propose a task-based hard attention mechanism that preserves information about previous tasks without affecting the current task's learning. Aljundi et al. [2] learn a representation for each task and use a set of gating autoencoders to decide which expert to use at testing time.

Another class of learning methods related to our approach are those based on the *Natural Gradient Descent (NGD)* [3, 104]. The NGD uses the Fisher Information Matrix (FIM) as the natural metric in parameter space. Candidate directions are projected using the inverse FIM and the best step (in terms of decreasing loss) is taken. The Projected NGD algorithm estimates a network reparamaterization online that whitens the FIM so that vanilla Stochastic Gradient Descent (SGD) is equivalent to NGD [23]. The authors of [60] show how the natural gradient can be used to explain and classify a range of adaptive stepsize strategies for training networks. In [35] the authors propose an approximation to the FIM for convolutional networks and show that the resulting NGD training procedure is many times more efficient than SGD.

4.3 Elastic weight consolidation

Elastic Weight Consolidation (EWC) addresses the problem of catastrophic forgetting in continual and sequential task learning in neural networks [55, 91]. In this section we give a brief overview of EWC and discuss some of its limitations.

4.3.1 Overview

The problem addressed by EWC is that of learning the K-th task in a network that already has learned $K-1$ tasks. The main challenge is to learn the new task in a way that prevents *catastrophic forgetting*. Catastrophic forgetting occurs when new learning interferes catastrophically with prior learning during sequential neural network training, which causes the network to partially or completely forget previous tasks [91].

The final objective is to learn the optimal set of parameters $\theta^*_{1:K}$ of a network given the previous ones $\theta^*_{1:K-1}$ learned from the previous $K-1$ tasks.[2] Each of the K tasks consists of a training dataset $\mathscr{D}_k = (\mathscr{X}_k, \mathscr{Y}_k)$ with samples $x \in \mathscr{X}_k$ and labels $y \in \mathscr{Y}_k$ (and for previous tasks $\mathscr{D}_{1:K-1} = (\mathscr{X}_1, \ldots, \mathscr{X}_{K-1}, \mathscr{Y}_1, \ldots, \mathscr{Y}_{K-1})$). We are interested in the configuration θ which maximizes the posterior $p_{1:K} \equiv p(\theta|\mathscr{D}_{1:K})$.

EWC [55] uses sequential Bayesian estimation [67] to factorize $p_{1:K}$ as:

$$\log p_{1:K} = \log p(\mathscr{Y}_K|\mathscr{X}_K;\theta) + \log p(\theta|\mathscr{D}_{1:K-1}) - \log p(\mathscr{D}_K|\mathscr{D}_{1:K-1}) + C \tag{4.1}$$

where $p_{1:K-1}$ is the prior model from previous tasks, $p_K \equiv p(\theta|\mathscr{X}_K, \mathscr{Y}_K)$ is the posterior probability corresponding to the new task, and C is a constant factor that can be ignored for training.

Since calculating the true posterior is intractable, EWC uses the Laplace approximation to approximate the posterior with a Gaussian:

$$\log p_{1:K} \approx \log p(\mathscr{Y}_K|\mathscr{X}_K;\theta) - \frac{\lambda}{2}\left(\theta - \theta^*_{1:K-1}\right)^\intercal \tilde{F}_{1:K-1}\left(\theta - \theta^*_{1:K-1}\right) + C' \tag{4.2}$$

$$\log p_{1:K} \approx \log p(\mathscr{Y}_K|\mathscr{X}_K;\theta) - \frac{\lambda}{2}\sum_i \tilde{F}_{1:K-1,ii}\left(\theta_i - \theta^*_{1:K-1,i}\right)^2 + C' \tag{4.3}$$

where $\tilde{F}_{1:K-1}$ is the Fisher Information Matrix (FIM) that approximates the inverse of the covariance matrix at $\theta^*_{1:K-1}$ used in the Laplace approximation of $\log p_{1:l}$ in (4.2), and the second approximation in (4.3) assumes $\tilde{F}$ is diagonal – a common assumption in practice – and thus that the quadratic form in (4.2) can be replaced with scaling by the diagonal entries $\tilde{F}_{1:K-1,ii}$ of the FIM at $\theta^*_{1:K-1}$.

The FIM of the true distribution $p\left(y|x;\theta\right)$ is estimated as:

$$F_\theta = \mathbb{E}_{x\sim\pi}\left\{\mathbb{E}_{y\sim p(y|x;\theta)}\left[\left(\frac{\partial \log p}{\partial \theta}\right)\left(\frac{\partial \log p}{\partial \theta}\right)^\intercal\right]\right\} \tag{4.4}$$

$$= \mathbb{E}_{x\sim\pi}\left\{\mathbb{E}_{y\sim p(y|x;\theta)}\left[\frac{\partial^2 \log p}{\partial \theta^2}\right]\right\}, \tag{4.5}$$

[2] We adapt the notation from [67] and [23].

where π is the empirical distribution of a training set $\mathcal{X}$.

This definition of the FIM as the second derivatives of the log-probability is key to understanding its role in preventing forgetting. Once a network is trained to a configuration θ^*, the FIM F_{θ^*} indicates how prone each dimension in the parameter space is to causing forgetting when gradient descent updates the model to learn a new task: it is preferable to move along directions with low Fisher information, since $\log p$ decreases slowly (the red line in Fig. 4.1). EWC uses F_{θ^*} in the regularization term of (4.2) and (4.3) to penalize moving in directions with higher Fisher information and which are thus likely to result in forgetting of already-learned tasks. EWC uses different regularization terms per task [55]. Instead of using (4.3), we only use the FIM from the previous task because we found it works better, requires storage of only one FIM, and better fits the sequential Bayesian perspective (as shown in [45]).

4.3.2 Limitations of EWC

Assuming the FIM to be diagonal is a common practice in the Laplace approximation for two reasons. First, the number of parameters is reduced from $O(N^2)$ to $O(N)$, where N is the number of elements in parameter space θ, so a diagonal matrix is much more efficient to compute and store. Additionally, in many cases the required matrix is the inverse of the FIM (for example in NGD methods [3]), which is significantly simpler and faster to compute for diagonal matrices.

However, in the case of sequential learning of new tasks with EWC, the diagonal FIM assumption might be unrealistic – at least in the original parameter space. On the left of Fig. 4.1 is illustrated a situation where a simple Gaussian distribution (solid blue ellipse) is approximated using a diagonal covariance matrix (black ellipse). By rotating the parameter space so that $\frac{\partial \log p}{\partial \theta}$ are aligned (on average) with the coordinate axes (Fig. 4.1, right), the diagonal assumption is more reasonable (or even true in the case of a Gaussian distribution). In this rotated parameter space, EWC is better able to optimize the new task while not forgetting the old one.

The example in Fig. 4.2 (Left) shows the FIM obtained for the weights in the second layer of a multilayer perceptron trained on MNIST [140] (specifically, four dense layers with 784, 10, 10, and 10 neurons). The matrix is clearly non-diagonal, so the diagonal approximation misses significant correlations between weights and this may lead to forgetting when using the diagonal approximation. The diagonal only retains 40.8% of the energy of the full matrix in this example.

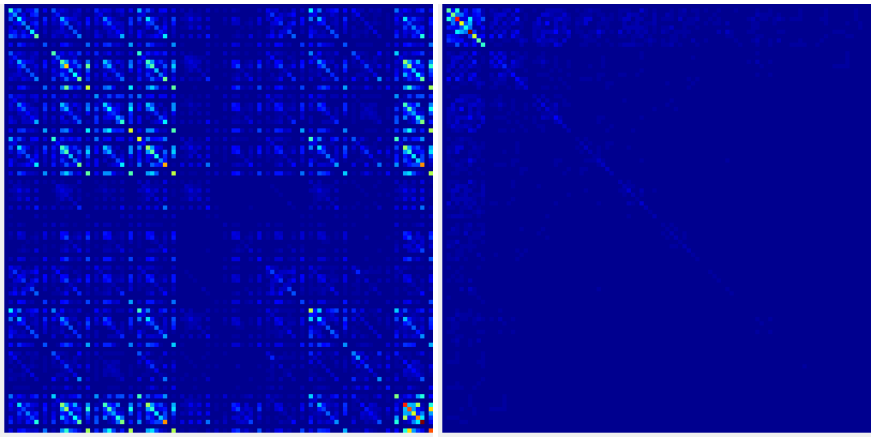

Figure 4.2 – Fisher Information Matrix: (Left) original and (Right) rotated using the proposed technique. The range is color coded and normalized for better visualization.

4.4 Rotated elastic weight consolidation

Motivated by the previous observation, we aim to find a reparameterization of the parameter space θ. Specifically, we desire a reparameterization which does not change the feed-forward response of the network, but that better satisfies the assumption of diagonal FIM. After reparameterization, we can assume a diagonal FIM which is efficiently estimated in that new parameter space. Finally, minimization by gradient descent on the new task is also performed in this new space.

One possible way to obtain this reparameterization is by computing a rotation matrix using the Singular Value Decomposition (SVD) of (4.4). Note that this decomposition is performed in the parameter space. Unfortunately, this approach has three problems. First, the SVD is extremely expensive to compute on very large matrices. Second, this rotation ignores the sequential structure of the neural network and would likely catastrophically change the feed-forward behavior of the network. Finally, we do not have the FIM in the first place.

4.4.1 Indirect rotation

In this section we show how rotation of fully connected and convolutional layer parameters can be applied to obtain a network for which the assumption of diagonal FIM is more valid. These rotations are chosen so as to not alter the feed-forward

response of the network.

For simplicity, we first consider the case of a single fully-connected layer given by the linear model $\mathbf{y} = W\mathbf{x}$, with input $\mathbf{x} \in \mathbb{R}^{d_1}$, output $\mathbf{y} \in \mathbb{R}^{d_2}$ and weight matrix $W \in \mathbb{R}^{d_2 \times d_1}$. In this case $\theta = W$, and to simplify the notation we use $L = \log p\left(\mathbf{y}|\mathbf{x}; W\right)$. Using (4.4), the FIM in this simple linear case is (after applying the chain rule):

$$\begin{aligned} F_W &= \mathbb{E}_{\substack{x\sim\pi \\ y\sim p}}\left[\left(\frac{\partial L}{\partial \mathbf{y}}\frac{\partial \mathbf{y}}{\partial W}\right)\left(\frac{\partial L}{\partial \mathbf{y}}\frac{\partial \mathbf{y}}{\partial W}\right)^{\mathsf{T}}\right] \\ &= \mathbb{E}_{p\sim\pi}\left[\left(\frac{\partial L}{\partial \mathbf{y}}\right)\mathbf{x}\mathbf{x}^{\mathsf{T}}\left(\frac{\partial L}{\partial \mathbf{y}}\right)^{\mathsf{T}}\right]. \end{aligned} \tag{4.6}$$

If we assume that $\frac{\partial L}{\partial \mathbf{y}}$ and $\mathbf{x}$ are independent random variables we can factorize (4.6) as done in [23]:

$$F_W = \mathbb{E}_{\substack{x\sim\pi \\ y\sim p}}\left[\left(\frac{\partial L}{\partial \mathbf{y}}\right)\left(\frac{\partial L}{\partial \mathbf{y}}\right)^{\mathsf{T}}\right]\mathbb{E}_{x\sim\pi}\left[\mathbf{x}\mathbf{x}^{\mathsf{T}}\right], \tag{4.7}$$

which indicates that we can approximate the FIM using two independent factors that only depend on the backpropagated gradient at the output $\frac{\partial L}{\partial \mathbf{y}}$ and on the input $\mathbf{x}$, respectively. This result also suggests that there may exist a pair of rotations of the input and the output, respectively, that lead to a rotation of F_W in the parameter space W.

In fact, these rotation matrices can be obtained as $U_1 \in \mathbb{R}^{d_1 \times d_1}$ and $U_2 \in \mathbb{R}^{d_2 \times d_2}$ from the following SVD decompositions:

$$\mathbb{E}_{x\sim\pi}\left[\mathbf{x}\mathbf{x}^{\mathsf{T}}\right] = U_1 S_1 V_1^{\mathsf{T}} \tag{4.8}$$

$$\mathbb{E}_{\substack{x\sim\pi \\ y\sim p}}\left[\left(\frac{\partial L}{\partial \mathbf{y}}\right)\left(\frac{\partial L}{\partial \mathbf{y}}\right)^{\mathsf{T}}\right] = U_2 S_2 V_2^{\mathsf{T}} \tag{4.9}$$

Since both rotations are local, i.e. they are applied to a single layer, they can be integrated in the network architecture as two additional (fixed) linear layers $\mathbf{x}' = U_1\mathbf{x}$ and $\mathbf{y} = U_2\mathbf{y}'$ (see Fig. 4.3).

The new, rotated weight matrix is then:

$$W' = U_2^{\mathsf{T}} W U_1^{\mathsf{T}}. \tag{4.10}$$

Thus, that the forward passes of both networks in Fig. 4.3 is equivalent since $U_2 W' U_1 = W$. In this way, the sequential structure of the network is not broken, and the learning problem is equivalent to learning the parameters of W' for the new problem $\mathbf{y}' = W'\mathbf{x}'$. The use of layer decomposition with SVD was also investigated

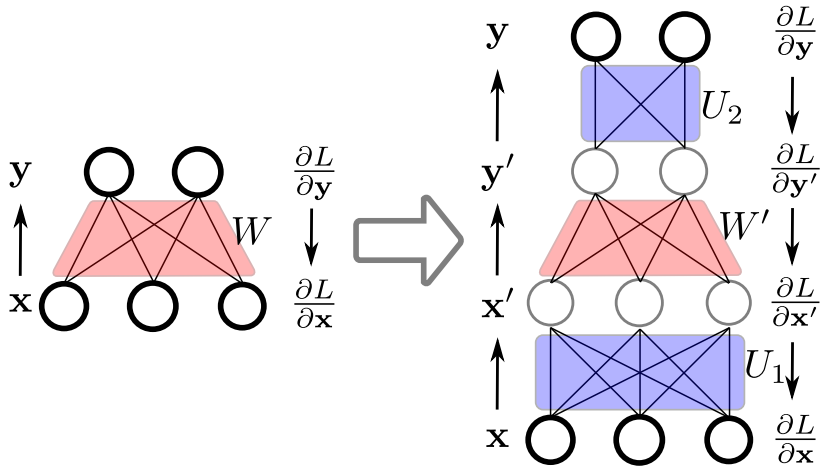

Figure 4.3 – Reparameterization of a linear layer W as W' using two additional linear layers U_1 and U_2.

in the context of network compression [69, 90]. However this SVD analysis was based on layer weight matrices and the original network layer was only decomposed into two new layers.

The training procedure is exactly the same as in (4.1), but using $\mathbf{x}'$, $\mathbf{y}'$ and W' instead of $\mathbf{x}$, $\mathbf{y}$ and W to estimate the FIM and to learn the weights. The main difference is that the approximate diagonalization of W in W' will be more effective in preventing forgetting using EWC. Fig. 4.2 (Right) shows the resulting matrix after applying the proposed rotations in the previous example. Note that most of the energy concentrates in fewer weights and it is also better conditioned for a diagonal approximation (in this case the diagonal retains 74.4% of the energy).

Assuming a block-diagonal FIM, the extension to multiple layers is straightforward by applying the same procedure layer-wise using the corresponding inputs instead of $\mathbf{x}$ and backpropagated gradients to the output of the layer as $\frac{\partial L}{\partial \mathbf{y}}$ (that is, estimating the FIM for each layer, and computing layer-specific U_1, W' and U_2). In Algorithm 3 we describe the reparameterization used in the training process.

4.4.2 Extension to convolutional layers

The method proposed in the previous section for fully connected layers can be applied to convolutional layers with only slight modifications. The rotations are performed by adding two additional 1×1 convolutional layers (see Fig. 4.4).

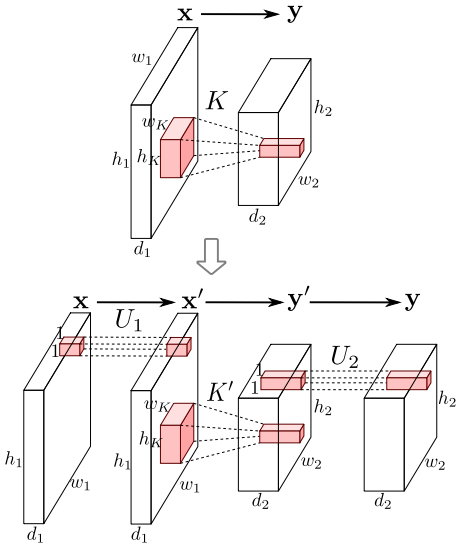

Figure 4.4 – Reparameterization of a convolutional layer K as K' using two additional 1×1 convolutional layers U_1 and U_2 as rotations.

Assume we have an input tensor $\mathbf{x} \in \mathbb{R}^{w_1 \times h_1 \times d_1}$, a kernel tensor $K \in \mathbb{R}^{w_k \times h_k \times d_1 \times d_2}$, and that the corresponding output tensor is $\mathbf{y} \in \mathbb{R}^{w_2 \times h_2 \times d_2}$. For convenience let the mode-3 fiber[3] of $\mathbf{x}$ be $\mathbf{x}_{l,m} = \left(x_{l,m,i} | i = 1, \ldots, d_1\right)^\intercal$ and of the output gradient tensor $\frac{\partial L}{\partial \mathbf{y}}$ as $\mathbf{z}_{l,m} = \left(\left(\frac{\partial L}{\partial \mathbf{y}}\right)_{l,m,i} | i = 1, \ldots, d_2\right)^\intercal$. Note that each $\mathbf{x}_{l,m}$ and $\mathbf{z}_{l,m}$ are d_1-dimensional and d_2-dimensional vectors, respectively. Now we can compute the self-correlation matrices averaged over all spatial coordinates as:

$$X = \frac{1}{w_1 h_1} \sum_{l=1}^{w_1} \sum_{m=1}^{h_1} \mathbf{x}_{l,m} \mathbf{x}_{l,m}^\intercal \tag{4.11}$$

$$Z = \frac{1}{w_2 h_2} \sum_{l=1}^{w_2} \sum_{m=1}^{h_2} \mathbf{z}_{l,m} \mathbf{z}_{l,m}^\intercal, \tag{4.12}$$

[3] A mode-i fiber of a tensor is defined as the vector obtained by fixing all its indices but i. A slice of a tensor is a matrix obtained by fixing all its indices but two. See [56] for more details.

Algorithm 3 : Incremental task learning.

Input: $\mathscr{D}_k = (\mathscr{X}_k, \mathscr{Y}_k)$, training samples in per-class sets.
Require: Initial parameters Θ_0, total number of tasks K
for $k = 1, \ldots, K$
  if k=1
    $\Theta \leftarrow$ UPDATE($(\mathscr{X}_k, \mathscr{Y}_k)$, Θ_0, 0)
  else
    $F_\Theta \leftarrow$ COMPUTE_FIM($(\mathscr{X}_{k-1}, \mathscr{Y}_{k-1})$, Θ^R)
    $\Theta^R \leftarrow$ UPDATE($(\mathscr{X}_k, \mathscr{Y}_k)$, Θ^R, F_Θ)
    $\Theta \leftarrow$ COMBINE(Θ^R)
  $\Theta^R \leftarrow$ ROTATE(Θ)
end for

UPDATE($(\mathscr{X}_k, \mathscr{Y}_k)$, Θ^R, F) above fits the model for task k using EWC, with rotated parameters Θ^R, and FIM F (0 is the zero matrix). COMBINE(Θ^R) fuses current parameters before computation of new rotated parameters for the coming tasks.

and compute the decompositions of (4.8) and (4.9) as

$$\mathbb{E}_{x\sim\pi}[X] = U_1 S_1 V_1^\intercal \tag{4.13}$$

$$\mathop{\mathbb{E}_{x\sim\pi}}_{y\sim p}[Z] = U_2 S_2 V_2^\intercal. \tag{4.14}$$

We define $K_{l,m} = \left(k_{l,m,i,j} | i = 1, \ldots, d_1, j = 1, \ldots, d_2\right)$, which is a slice[3] of the kernel tensor K. The rotated slices are then obtained as:

$$K'_{l,m} = U_2^\intercal K_{l,m} U_1^\intercal. \tag{4.15}$$

and the final rotated kernel tensor $K' \in \mathbb{R}^{w_k \times h_k \times d_1 \times d_2}$ is obtained simply by tiling all slices $K'_{l,m}$ computed for every l and m.

4.5 Experimental results

In this section we report on a number of experiments comparing our approach to EWC [55] and other baselines.[4]

[4] Code available at: https://github.com/xialeiliu/RotateNetworks

Table 4.1 – Ablation study on Disjoint MNIST when $T=2$. Numbers in **bold** indicate the best performing configuration of each method. All versions of R-EWC that rotate fully-connected layers significantly outperform EWC.

	$\lambda=1$		$\lambda=10$		$\lambda=100$		$\lambda=1000$		$\lambda=10000$	
	T1	T2	T1	T2	T1	T2	T1	T2	T1	T2
FT	6.1	97.6	6.1	97.6	6.1	97.6	6.1	97.6	6.1	97.6
EWC [55]	66.8	90.9	75.3	95.6	**85.8**	**92.8**	78.4	93.7	81.0	88.8
R-EWC - conv only	62.7	89.2	67.5	96.1	80.4	91.4	**84.7**	**93.1**	75.5	93.7
R-EWC - fc only	78.9	95.3	79.0	95.8	87.4	93.5	93.0	82.3	**94.3**	**88.0**
R-EWC - all	77.2	96.7	**91.7**	**91.2**	86.9	95.9	96.3	81.1	92.1	86.0
R-EWC - all no last	71.5	91.8	84.9	97.0	**91.6**	**94.5**	94.6	88.4	97.9	79.4

4.5.1 Experimental settings

Datasets. We evaluate our method on two small datasets and three fine-grained classification datasets: MNIST [140], CIFAR-100 [57], CUB-200 Birds [143] and Stanford-40 Actions [157]. Each dataset is equally divided into T groups of classes, which are seen as the sequential tasks to learn. In the case of the CUB-200 dataset, we crop the bounding boxes from the original images and resize them to 224×224. For Actions, we resize the input images to 256×256, then take random crops of size 224×224. We perform no data augmentation for the MNIST and CIFAR-100 datasets.

Training details. We chose the LeNet [64] and VGG-16 [135] network architectures, which are slightly modified due to the requirements of different datasets. For MNIST, we add 2×2 padding to the original 28×28 images to obtain 32×32 input images for LeNet. LeNet is trained from scratch, while for CIFAR-100 the images are passed through the VGG-16 [135] network pre-trained on ImageNet, which has been shown to perform well when changing to other domains. The input images are 32×32×3 and this provides a feature vector of 1×1×512 at the end of the *pool5* layer. We use those feature vectors as an input to a classification network consisting of 3 fully-connected layers of output dimensions of 256, 256 and 100, respectively. For the two fine-grained datasets, we fine-tune from the pre-trained model on ImageNet. To save memory and limit computational complexity, we add a global pooling layer after the final convolutional layer of VGG-16. The fully-connected layers used for our experiments are of size 512, 512 and the size of the output layer corresponding to the number of classes in each dataset. The Adam optimizer is used with a learning rate of 0.001 for all experiments. We train for 5 epochs on MNIST and for 50 epochs

Table 4.2 – Comparison EWC / R-EWC for $T=2$.

	EWC [55] (T1 / T2)	R-EWC (T1 / T2)
MNIST	89.3 (85.8 / 92.8)	**93.1** (91.6 / 94.5)
CIFAR-100	37.5 (23.5 / 51.5)	**42.5** (30.2 / 54.7)
CUB-200 Birds	45.3 (42.3 / 48.6)	**48.4** (53.3 / 45.2)
Stanford-40 Actions	50.4 (44.3 / 58.4)	**52.5** (52.3 / 52.6)

on the other datasets.

Evaluation protocols. In our experiments, we share all layers in the networks across all tasks during training, which allows us to perform inference without explicitly knowing the task. We report the classification accuracy of each previous task $T-1$ and the current task T after training on the T-th task. When the number of tasks is large, we only report the average classification accuracy over all trained tasks.

Lifelong learning is evaluated in two settings depending on knowledge of the task label at inference time. Here we consider the more difficult scenario where task labels are unknown, and results cannot be directly compared to methods which consider labels [80, 114, 131]. As a consequence our method is implemented with as the last layer in a single network head, which is also used in [14]. During training we increase the number of output neurons as new tasks are added.

4.5.2 Disjoint MNIST comparison and ablation study

We use the disjoint MNIST dataset [140], which assigns half of the numbers as task 1 and the rest as task 2. We compare our method (R-EWC) with fine-tuning (FT) and Elastic Weight Consolidation (EWC) [55]. First, task 1 is learned from scratch on the LeNet architecture. For FT, task 2 is learned starting from task 1 as initialization. For EWC and R-EWC, task 2 is learned according to the corresponding method. In addition, we also train task 2 with only applying R-EWC to the convolutional layers (conv only), to fully connected layers (fc only), and to all layers except for the last fully-connected (all no last).

Table 4.1 compares the performance of the proposed methods for different values for the trade-off parameter λ between classification loss and FIM regularization. Results were obtained using 200 randomly selected samples (40 per class) from the validation set for computing the FIM. Each experiment was executed 3 times and the results in Table 4.1 are the average.

Results show that R-EWC clearly outperforms FT and EWC for all λ, while the best trade-off value might vary depending on the layers involved. As expected, lower values of the trade-off tend towards a more FT strategy where task 1 is forgotten more quickly. On the other hand, larger values of the trade-off give more importance to keeping the weights useful for task 1, avoiding catastrophic forgetting but also making task 2 a bit more difficult to learn. In conclusion, the improvement of our proposed method over EWC is that while maintaining similar task 2 performance, it allows for much less catastrophic forgetting on task 1. We have observed that during training the regularized part of the FIM is usually between 10^{-2} to 10^{-4}, which could explain why values around $\lambda = 100$ give a more balanced trade-off.

4.5.3 Comparison with EWC on two tasks

We further compare EWC and R-EWC on several larger datasets divided into two tasks – that is, in which all datasets are divided into 2 groups with an equal number of classes. The network is trained on task 1 as usual, and then both methods are applied for task 2. After the learning process is done, we evaluate the two tasks again. The accuracy for each task and average accuracy of two tasks are reported in Table 4.2. Results show that our method clearly outperforms EWC for all datasets with an absolute gain on accuracy of R-EWC over EWC that varies from 2.1% to 5%. When comparing the accuracy on the first task only, R-EWC forgets significantly less in all cases while attaining similar accuracy on the second task.

4.5.4 Comparison with EWC on more tasks

We compare both EWC and R-EWC when having more tasks for datasets with larger images. We divide both CUB-200 Birds and Stanford-40 Actions datasets into four groups of an equal number of classes. We train a network on task 1 for both methods and proceed to iteratively learn the other tasks one at a time, while testing the performance at each stage. Results are shown in Figure 4.5, where we observe that the accuracy decreases with increasing number of tasks for both methods as expected. However, R-EWC outperforms EWC consistently, by a margin that grows larger as more tasks are learned on Stanford-40 Actions dataset. Note that in lifelong learning settings it becomes more difficult to balance performance of new tasks as the number of previous learned tasks increases. Results for all previous tasks after training the T-th task on Stanford-40 Actions are given in Table 4.3. For each single previous task, our method manages to avoid forgetting better than EWC.

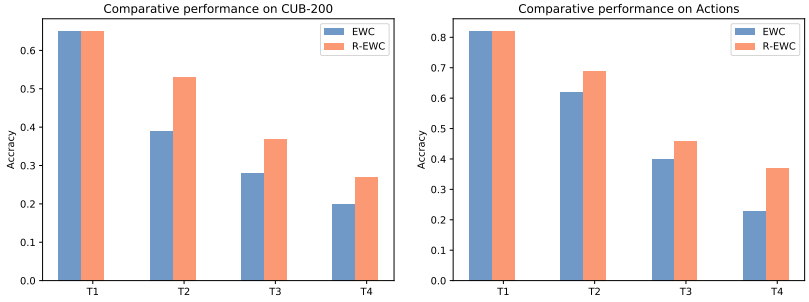

Figure 4.5 – Comparison with EWC when $T=4$ on CUB-200 Birds and Stanford-40 Actions datasets.

Table 4.3 – Comparison EWC / R-EWC for $T=4$ on Stanford Actions.

Current Task	Accuracy T1	T2	T3	T4	Average
T1	81.5 / 81.5	-	-	-	81.5 / 81.5
T2	49.5 / **55.4**	75.8 / **81.5**	-	-	62.0 / **69.0**
T3	6.1 / **18.8**	45.1 / **48.7**	69.9 / **72.1**	-	40.3 / **47.2**
T4	0.0 / **12.0**	7.0 / **31.3**	44.5 / **56.9**	46.8 / **50.9**	23.0 / **37.2**

4.5.5 Comparison with the state-of-the-art

We compare our method (R-EWC) with fine-tuning (FT), Elastic Weight Consolidation (EWC) [55], Learning without Forgetting (LwF) [71] and Expert Gate (EG) [2]. We exclude methods which use samples of previous tasks during training of new tasks. We split the CIFAR-100 dataset [57] into 4 groups of classes, where each group corresponds to a task. For EG, the base network is trained on the 4 tasks independently, and for each of them we learn an auto-encoder with dimensions 4096, 1024 and 100, respectively. For FT, each task is initialized with the weights of the previous one. In addition, an UpperBound is shown by learning the newer tasks with all the images for all previous tasks available.

Results are shown in Figure 4.6, where we clearly see our method outperforms all others. FT usually performs worse compared to other baselines, since it tends to forget the previous tasks completely and is optimal only for the last task. EG usually has higher accuracy when the tasks are easy to distinguish (on CUB-200 and Stanford Actions, for example), however it is better than FT but worse than other

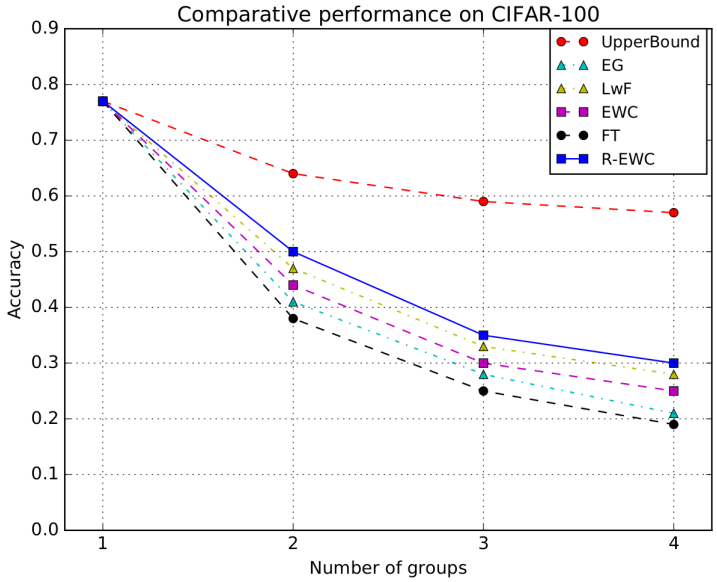

Figure 4.6 – Comparison with the state-of-the-art on CIFAR-100.

baselines in this setting since the groups are randomly sampled from the CIFAR-100 dataset. EWC performs worse than LwF, however our method gains about 5% over EWC and achieves better performance than LwF. While our method still performs worse than the UpperBound, this baseline requires all data at all training times and can not be updated for new tasks.

4.6 Conclusions

EWC helps to prevent forgetting but is very sensitive to the diagonal approximation of the FIM used in practice (due to the large size of the full FIM). We show that this approximation discards important information for preventing forgetting and propose a reparametrization of the layers that results in more compact and more diagonal FIM. This reparametrization is based on rotating the FIM in the parameter space to align it with directions that are less prone to forgetting. Since direct rotation is not possible due to the feedforward structure of the network, we devise an indirect method that approximates this rotation by rotating intermediate features, and that can be easily implemented as additional convolutional and fully connected layers.

However, the weights in these layers are fixed, so they do not increase the number of parameters. Our experiments with several tasks and settings show that EWC in this rotated space (R-EWC) consistently improves the performance compared to EWC in the original space, obtaining results that are comparable or better than other state-of-the-art algorithms using weight consolidation without exemplars.

5 Generative Feature Replay for Task-agnostic Continual Learning[1]

5.1 Introduction

Humans and animals are capable of continually acquiring and updating knowledge throughout their lifespan. The ability to accommodate new knowledge while retaining previously one is referred to as *continual learning*, which is essential to building artificially intelligent systems. Current deep neural networks have achieved impressive performance in many benchmarks, comparable and even better than humans (e.g. image classification [38]). However, when trained for another task, the networks forget almost completely the previous ones, due to the problem of *catastrophic interference* [91] between the new and previous tasks.

Motivated by this limitation, continual learning has become a very active research topic in recent years. Several approaches, inspired in part by biological systems, have been proposed for overcoming this *catastrophic forgetting*. The first category of approaches propose regularizers that can limit the plasticity of the network while training on new tasks so the network remains stable for previous tasks [1, 55, 71, 159]. Another type of approach is to dynamically increase the capacity of the network to accommodate new tasks [66, 116, 123], often combined with task-dependent masks on the weights [87, 88] or activations [123] to reduce the chance of catastrophic interference. A third approach relies on using a fraction of data from the previous tasks while learning new tasks. These data can be real samples (rehearsal) [14, 80, 114] or synthetic samples, replayed by generative models learned during previous tasks (also known as pseudorehearsal) [115, 131, 150].

Another fundamental problem that arises in continual learning, although less studied than catastrophic forgetting, is how to aggregate the current to the previously learned tasks in a *task-agnostic* network. Most works in continual learning tend to assume that the task is known during inference, which greatly simplifies training and inference since each task-specific classifier is trained independently.

Generative replay approaches can address both forgetting and task-agnostic classification. However, previous works have focused on replaying full images [97, 101, 131, 150], which is inefficient. Current state-of-the-art image generation mod-

[1] This chapter is under submission at the Conference of Computer Vision and Pattern Recognition (CVPR, 2020)

els [10, 53] struggle to generate high-quality and realistic images from limited data.

Motivated by recent works on zero- and few-shot learning [152, 153], in this chapter we study feature generation applied to two specific problems of continual learning: catastrophic forgetting and task aggregation for task-agnostic classification. We visualize and analyze how and where the network is forgetting, and, based on this analysis, we design a hybrid model which combines generative feature replay at the classifier level and distillation in the feature extractor. We show that this effectively mitigates forgetting and is computationally efficient and scalable. To our knowledge, we are the first to investigate the potential of generative feature replay for continual learning.

5.2 Related work

In addition to distillation, catastrophic forgetting can be partially mitigated by weight regularization. Elastic Weight Consolidation (EWC) [55] regularizes network parameters using a diagonal approximation the Fisher Information Matrix (FIM). R-EWC (introduced in Chapter 4) proposes a better approximation of FIM using a re-parameterization of the layers. Scalable versions of the EWC have been proposed concurrently in [14] and [116]. The regularization strength can be also computed online by accumulating gradients [159]. and in an unsupervised manner using sensitivity analysis [1]. Rehearsal methods save and exploit data from previous tasks, such as real samples [114] or gradients [15, 80].

Memory replay for continual learning has been explored at the image level with different generative models, including autoencoders [54], GANs [131] and conditional GANs [101, 121, 150]. Training generative image models is difficult and most works focus on the design of new models [10, 53, 82, 110], losses [5, 36], regularization [94] and conditioning mechanisms [92, 95]. Recently, Xian *et al.* [152, 153] proposed feature generation to overcome the lack of visual training data in zero- and few-shot learning, where maps to unseen/few-shot classes are learned by synthesizing CNN features. We use feature generation as memory replay mechanism to prevent forgetting in task-agnostic continual learning.

5.3 Analyzing continual learning in feature extractor and classifier

In this section we take a closer look at how forgetting affects the feature extractor and the classifier.

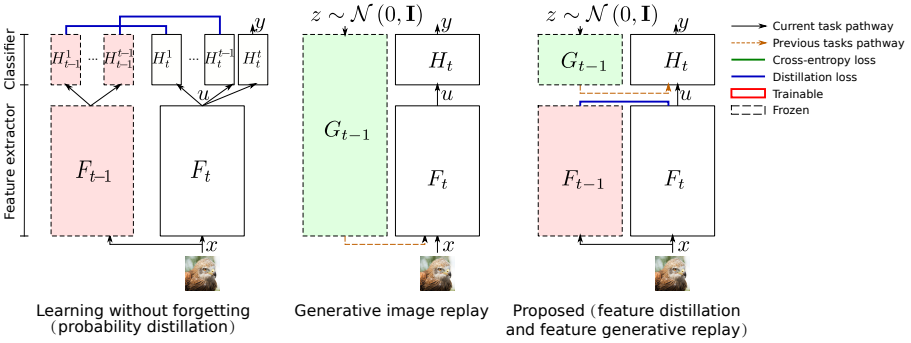

Figure 5.1 – Preventing forgetting in feature extractor and classifier. The proposed method combines generative replay in the classifier layer and distillation in the early layers (i.e. feature extractor).

5.3.1 Continual learning in classification networks

Classification model and task. We consider classification tasks learned from a dataset $\mathscr{D} = \{(\mathbf{x}_i, l_i)\}_{i=1}^N$, where $\mathbf{x}_i \in \mathscr{X}$ is the ith image, $l_i \in \mathscr{C}$ is the corresponding label (from a vocabulary of K classes) and N is the size of the dataset. The classifier network has the form $\mathbf{y} = C(\mathbf{x};\theta,V) = H(F(\mathbf{x};\theta);V)$, where we explicitly distinguish between *feature extractor* $F(\mathbf{x};\theta)$, parametrized by θ, and *classifier* $H(\mathbf{u};V) = \mathscr{A}(V\mathbf{u})$, where V is a matrix projecting the output of the feature extractor u to the class scores (in the following we omit parameters θ and V), and $\mathscr{A}$ is the softmax function that normalizes the scores to class probabilities. During training we minimize the cross-entropy loss between true labels and predictions $\mathscr{L}_{\text{CE}}(\mathscr{D}) = -\Sigma_{i=1}^N \mathbf{l}_i \cdot \log \mathbf{y}_i$, where $\mathbf{l}_i$ is the one-hot representation of class label $l_i \in \mathscr{C}$.

Continual learning. We consider the continual learning setting where T classification tasks are learned independently and in sequence from the corresponding datasets $\mathscr{D}_1, \ldots, \mathscr{D}_t, \ldots, \mathscr{D}_T$. The resulting model C_t after learning task t has feature extractor F_t and classifier H_t . We assume that the classes in each task are disjoint, i.e. $\mathscr{C}_t \bigcap \mathscr{C}_{t'} = \emptyset$ for all $t' \neq t$. Ideally, after learning task t, the model can perform inference on all tasks $t' \leq t$ (i.e. it remembers current and previous tasks).

Task awareness. There are two evaluation scenarios: *task-aware* classification and *task-agnostic* classification, depending on whether the task index t is known or unknown at inference time. The former is a simpler problem where the potential predictions are limited to $\mathscr{C}_t$, and it is often addressed using a multi-head architecture where classifiers $H_1, \ldots, H_t$ are indepedent. In the case of multi-head networks

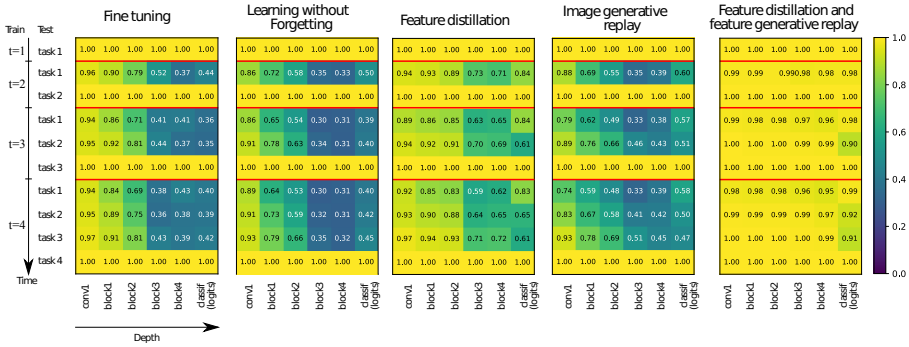

Figure 5.2 – CCA similarity between features at different layers for different continual learning methods (CIFAR-100, four tasks with 25 classes each).

we denote the class probability estimated at head j by model C_t as $\mathbf{y}^{t,j} = C_t^j(\mathbf{x})$. Task-agnostic classification is more complex, since it requires predictions over the aggregated class space, i.e. $\mathscr{C}'_t = \bigcup_{j=1}^{t} \mathscr{C}_j$.

Catastrophic forgetting. The challenge in continual learning lies in dealing with the interference between learning the new task (observing new data from $\mathscr{D}_t$) and not forgetting the previous ones $1,\ldots,t-1$. This occurs because parameters are shared across tasks. If the underlying distributions of data for the new and previous tasks are not the same (e.g. domain or task shift), this interference will cause performance to drop, which is known as *(catastrophic) forgetting*.

Figure 5.2 (far left) illustrates the effect of continual learning (via simply fine-tuning the network on new tasks) over features extracted at different layers of the network. Forgetting is measured using Canonical Correlation Analysis (CCA) similarity[2] between the features extracted for task $t' \leq t$ by model C_t and the optimal model $C_{t'}$ (i.e. trained at time t' with $\mathscr{D}_{t'}$). Earlier features remain fairly correlated, while the correlation decreases progressively with increasing layer depth. This suggests that forgetting in higher-level features is more pronounced, since they become progressively more task-specific, while lower features are more generic and closer to the visual domain of the task.

[2] CCA similarity computes the similarity between distributed representations even when they are not aligned. This is important, since learning new tasks may change how different patterns are distributed in the representation. We use SVCCA [111] which first removes noise using singular value decomposition (SVD).

5.3.2 Learning without forgetting

A popular method to prevent forgetting in task-aware classification is *Learning without Forgetting* (LwF) [71], which keeps a copy of the model C_{t-1} before learning the new task and distills its predicted probabilities into the new model C_t (which may otherwise suffer interference from the current task t). In particular, LwF uses a modified cross-entropy loss over each head of previous tasks: $\mathscr{L}_{\text{LwF}}(\mathscr{X}_t) = -\mathbb{E}_{\mathbf{x}\sim\mathscr{X}_t}\Sigma_{j=1}^{t-1}\tilde{\mathbf{y}}^{t-1,j}\cdot\log\tilde{\mathbf{y}}^{t,j}$ where probabilities $\tilde{\mathbf{y}}$ are further renormalized from the output probabilities $\mathbf{y}$ as $\tilde{y}^{(c)} = \left(y^{(c)}\right)^{1/\mathscr{T}} / \Sigma_{k=1}^{K}\left(y^{(k)}\right)^{1/\mathscr{T}}$ (with temperature $\mathscr{T}=2$).

Note that the probabilities $\mathbf{y}^{t-1,j}$ and $\mathbf{y}^{t,j}$ are always estimated with current input samples $\mathbf{x}\in\mathscr{X}_t$, since data from previous tasks is not available. Since tasks are different, there is a distribution shift in the visual domain (i.e. $\mathbf{y}^{t-1,j}$ if extracted from $\mathbf{x}\in\mathscr{X}_{t-1}$ instead of $\mathbf{x}\in\mathscr{X}_t$), which can reduce the effectiveness of distillation when the domain shift from $\mathscr{X}_{t-1}$ to $\mathscr{X}_t$ is large. Figure 5.2 shows how LwF helps to increase the CCA similarity for previous tasks at the classifier and higher-level layers, effectively alleviating forgetting and maintaining higher accuracy for previous tasks than fine tuning. However, the correlation at middle and lower-level layers in the feature extractor remains similar or lower to the case of fine tuning, which may be a problem since representations at those level seem to experience forgetting. This may be caused by the shift due by probing with images from the latest task instead of a previous one and that the distillation constraint on the probabilities is too loose to enforce correlation in intermediate features.

Another limitation of LwF is that it was designed for task-aware classification with multiple heads. A naive extension to task-agnostic classification consists simply in selecting the class with maximum probability out of the aggregated outputs of all heads. However, each head of the classifier is unaware of the others and the resulting probabilities are not calibrated, leading to poor performance. This problem is not limited to LwF, but to all task-aware multi-head approaches. Generative replay provides a possible solution to this issue.

5.3.3 Generative image replay

The lack of training images for previous tasks in continual learning has been addressed with a generator of images from previous tasks and using them during the training of current and future tasks [97, 101, 131, 150]. We consider conditional GAN with Projection Discriminator [95], which can control the class of generated images. At time t, the image generator samples images $\hat{\mathbf{x}} = G_{t-1}(c,\mathbf{z})$ where c is the desired class and $\mathbf{z}$ is a random latent vector sampled from a simple distribution (typically a normalized Gaussian). These generated images are combined with current data in an augmented dataset $\mathscr{D}'_t = \{(\hat{\mathbf{x}}_i,\hat{l}_i)\}_{i=1}^{N_R}\cup\mathscr{D}_t$, where $\hat{\mathbf{x}}_i = G_{t-1}(\hat{l}_i,\mathbf{z}_i)$ and N_R is

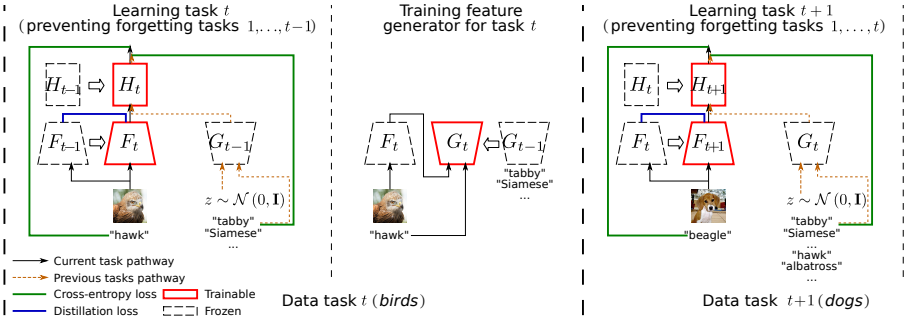

Figure 5.3 – Proposed framework. Distillation and feature generation are used during training to prevent forgetting previous tasks. Once the task is learned, the feature generator is updated with adversarial training and distillation to prevent forgetting in the generator.

the number of replayed images for previous tasks (typically distributed uniformly across tasks and classes).

Generative image replay, while appealing, has numerous limitations in practice. First, real images are high dimensional representations and the image distribution of a particular task lies in a narrow yet very complex manifold. This complexity requires deep generators with many parameters and are computationally expensive, difficult to train, and often highly dependent on initialization [81]. Training these models requires large amounts of images, which is rarely the case in continual learning. Even with enough training images, the quality of the generated images is often not unsatisfactory as training data for the classifier, since they may not capture relevant discriminative features. Figure 5.2 shows the CCA similarity for class-conditional GAN. It shows a similar pattern to LwF and fine tuning with the similarity decreasing especially in intermediate layers.

5.4 Feature distillation and generative feature replay

Motivated by the previous observations, we propose generative *feature* reply as an alternative to images. We combine feature distillation and feature replay in a hybrid model that is effective, efficient, and allows robust task-agnostic classification (see Figure 5.1 right). In particular, we use distillation at the output of the feature extractor in order to prevent forgetting in the feature extractor, while using feature replay of the same features to prevent forgetting in the classifier.

Our framework consists of three modules: feature extractor, classifier, and

feature generator. To prevent forgetting we also keep a copy of the previous feature extractor, classifier and feature generator. Figure 5.3 illustrates continual learning in our framework. The classifier H_t and feature extractor F_t for task t are implicitly initialized with H_{t-1} and F_{t-1} (which we duplicate and freeze) and trained using feature replay and feature distillation. When the feature extractor and classifier are trained, we then freeze them and then train the feature generator G_t (implicitly initialized as G_{t-1}, which we may copy, freeze, and then use to prevent forgetting in the generator).

5.4.1 Feature extractor with feature distillation

We prevent forgetting in F_t by distilling the features extracted by F_{t-1} via the following L2 loss:

$$\mathscr{L}_{F_t}^{\text{FD}}(\mathscr{X}_t) = \mathbb{E}_{\mathbf{x}\sim\mathscr{X}_t}\left[\|F_t(\mathbf{x}) - F_{t-1}(\mathbf{x})\|_2\right].$$

Note that in this case there are not multiple losses in multiple heads because the feature u is shared, and the loss can be applied on any features (e.g. tensors). The classifier heads of previous tasks remain unchanged. This enforces a more strict spatially-aware alignment than the temperature-scaled cross-entropy on the probabilities used by LwF. Note in Figure 5.2 (center) how the CCA similarity compared with LwF increases in the layers of the feature extractor. However, we observed that the accuracy drops almost to that of fine tuning, suggesting that the classifier is still critical to avoiding forgetting.

5.4.2 Feature generator

To prevent forgetting in the classifier we train a feature generator G_t to model the conditional distribution of features $p_{\mathbf{u}}(u|c)$ as $\mathbf{u} = G_t(\mathbf{z})$, and sample from it while learning future tasks. We evaluated two variants: *Gaussian class prototypes* and *conditional GAN with replay alignment.*

Gaussian class prototypes. We represent each class c of a task t as a simple Gaussian distribution $G_t(c,\mathbf{z}) = G_t^{(c)}(c,\mathbf{z}) = \mathscr{N}(u;\mu_t^{(c)},\Sigma_t^{(c)})$, where $\mathscr{N}(\cdot;\cdot,\cdot)$ is a Gaussian distribution whose parameters are estimated using $\{u = F_t(\mathbf{x}_i), \forall(\mathbf{x}_i,l_i) \in \mathscr{D}_t, l_i = c\}$. This variant has the advantage of compactness and efficient sampling.

Conditional GAN with replay alignment. To generate more complex distributions and share parameters across classes and tasks, we also consider GANs as generators. For example, the Wasserstein GAN adapted to feature generation and continual

learning uses the following losses (after learning task t and before task $t+1$):

$$\mathscr{L}_{D_t}^{\text{WGAN}}(\mathscr{X}_t) = -\mathbb{E}_{\mathbf{u}\sim\mathscr{D}_t}\left[D_t(c, F_t(\mathbf{x}))\right] + \mathbb{E}_{\mathbf{z}\sim p_z, c\sim\{1,\ldots,K_t\}}\left[D_t(c, G_t(c,\mathbf{z}))\right]$$
$$\mathscr{L}_{G_t}^{\text{WGAN}}(\mathscr{X}_t) = -\mathbb{E}_{\mathbf{z}\sim p_z, c\sim\{1,\ldots,K_t\}}\left[D_t(c, G_t(c,\mathbf{z}))\right]$$
$$\mathscr{L}_{G_t}^{\text{RA}} = \Sigma_{j=1}^{t-1}\Sigma_{c=1}^{K_j-1}\mathbb{E}_{\mathbf{z}\sim p_\mathbf{z}}\left[\|G_t(c,\mathbf{z}) - G_{t-1}(c,\mathbf{z})\|_2^2\right],$$

where $\mathscr{L}_{G_t}^{\text{RA}}$ is the replay alignment loss (which can be seen as a type of distillation) [150]. The replay alignment loss encourages the current generator G_t to replay exactly the same features as G_{t-1} when conditioned on a given previous class c and a given latent vector z [150]. We use a discriminator D_t during the adversarial training, which alternates updates of D_t and G_t (i.e. $\min_{D_t}\mathscr{L}_{D_t}^{\text{WGAN}}(\mathscr{X}_t)$ and $\min_{G_t}\mathscr{L}_{G_t}^{\text{WGAN}}(\mathscr{X}_t) + \mathscr{L}_{G_t}^{\text{RA}}$, respectively).

5.4.3 Task-agnostic classifier

We are interested in a single head architecture that provides well-calibrated, task-agnostic predictions, which naturally arises if all tasks are learned jointly when all data is available. In our case we extend the last linear layer V_{t-1} to V_t by increasing its size to accommodate the projection to $\mathscr{C}_t$. The softmax is also extended to this new size. During training we combine the available real data for the current task (fed to F_t) with generated features for previous tasks $\{(\hat{\mathbf{u}}_i, \hat{l}_i)\}_{i=1}^{N_R}$. Since we only train a linear layer with features, this process is efficient.

Figure 5.2 (far right) shows that our method (trained for task-agnostic classification) is capable of preserving similar representations for previous tasks at all layers, including the classifier. Our combination of distillation and replay maintains higher classification accuracy accross all tasks, effectively addressing the problems of forgetting and task aggregation.

5.5 Experimental results

Our evaluation involves classification datasets that we split into disjoint subsets of classes, which are learned sequentially as separate tasks.

Datasets. We evaluate performance on two datasets: CUB-200-2011 [149] and CIFAR-100 [57]. We resize images from CUB-200-2011 to 256×256, randomly sample 224×224 crops during training, and use the center crop during testing. CIFAR-100 images are padded with 4 pixels, from which 32×32 crops are randomly sampled. The original center crop is used for testing. Random horizontal flipping is used as data augmentation.

Table 5.1 – Comparison with other methods on CIFAR-100 for the 4-task scenario.

	Task-agnostic					Task-aware				
	T1	T2	T3	T4	Ave	T1	T2	T3	T4	Ave
LwF	83.5	42.5	27.1	21.5	43.7	83.5	61.0	41.5	37.6	55.9
EWC		53.8	33.0	26.2	49.1		69.0	54.2	54.0	65.2
MAS		47.8	33.2	27.8	48.1		66.7	52.9	54.9	64.5
Generative Image Replay		37.4	25.4	18.6	41.5		40.2	33.4	34.0	48.0
Feature Distillation + Gaussian		54.2	44.0	37.1	54.7		69.5	62.6	61.0	69.2
Feature Distillation + GAN		56.3	46.8	41.0	**56.9**		68.5	63.3	62.3	**69.4**

Table 5.2 – Ablation study of different regularization methods on CIFAR-100.

	Task-agnostic					Task-aware				
	T1	T2	T3	T4	Ave	T1	T2	T3	T4	T5
EWC + GAN	83.5	40.8	26.8	21.2	43.1	83.5	69.6	55.2	49.4	64.4
MAS + GAN		40.2	26.0	20.9	42.7		64.9	52.4	56.4	64.3
Feature Distillation + GAN		56.3	46.8	41.0	**56.9**		68.5	63.3	62.3	**69.4**

Training details. We use Pytorch as our framework [105]. We use ResNet-101 [40] pretrained on ImageNet for CUB-200-2011. For CIFAR-100, we modify the ResNet-18 network to use 3×3 kernels for the first convolutional layer, and train the model from scratch. We train each classification task for 200 epochs and GANs for 500 epochs. The Adam optimizer is used in all the experiments, and the learning rate on CIFAR-100 for classification is 1e-3 and for GANs is 1e-4. On CUB-200-2011, we reduce the learning rate to 1e-5.

Evaluation protocol. We evaluate the average overall accuracy of all previous tasks combined with current task for both task-aware and task-agnostic continual learning.

5.5.1 Continual learning on CIFAR-100 for the 4-task scenario

We first compare the best results obtained by our approach with other methods. Then we report on an ablation study the illustrated different aspects of our method.

Comparison with other methods. We divide CIFAR-100 dataset into 4 tasks and compare our approach with several state-of-the-art methods: LwF [71], EWC [55], MAS [1] and generative image replay[131, 150]. We train the first three methods using multi-head networks, where each task has a separate head since they will not work with single-head when there are no exemplars. For task-agnostic we simply pick the maximum probability across all heads. We found this to be a very strong baseline for task-agnostic learning, as shown in Table 5.1. Our method outperforms all baselines by a large margin, especially in the task-agnostic case. EWC and MAS perform similar in terms of average accuracy across all tasks, but our

Table 5.3 – Ablation study of replaying different feature on CIFAR-100 for the 4-task scenario.

	Task-agnostic					Task-aware				
	T1	T2	T3	T4	Ave	T1	T2	T3	T4	Ave
Pixel	83.5	37.4	25.4	18.6	41.5	83.5	40.2	33.4	34.0	48.0
Conv1		35.8	13.3	6.3	34.5		44.6	22.9	15.5	41.4
Block 1		37.7	22.1	10.2	38.2		48.9	33.4	20.3	46.3
Block 2		39.6	21.2	14.2	39.4		53.3	36.9	32.1	51.2
Block 3		61	40.4	35.8	55.2		75.7	61.8	56.4	69.3
Final layer		56.3	46.8	41.0	**56.9**		68.5	63.3	62.3	**69.4**

approach is about 7% higher than both. Note that all methods work much better in the task-aware setting, which confirms that task-agnostic continual learning is more challenging. Our method surpasses MAS by 4.9%. We combine the generated images and data from current task to train a classifier jointly. However, it does not perform well on previous tasks due to the mismatch between the generated and real distribution. Joint training refers to training with all data for all tasks, which serves as upper bound.

Ablation study on the 4-task scenario. As illustrated in Table 5.1, Gaussian class prototypes work surprisingly well compared to our GAN variant. It is 1.7% worse for task-agnostic and comparable accuracy for task-aware. In Table 5.2 we compare different regularization methods. It is clear that feature distillation works best. However, adding constraints on parameter space does guarantee generated features close to real ones. Finally, instead of generating final-layer features, we investigate how replay works when applied at intermediate layers. Replaying intermediate features achieves reasonable results compared to replaying features of last layer (about 2% worse for task-agnostic and only 0.3% worse for task-aware, as shown in Table 5.3). Note that generating images performs better than generating features until Block 3, but much worse than generating at deeper layers. In practice, we would rather generate features at the final layer than at complex intermediate ones.

5.5.2 Continual learning on CUB-200-2011

Here we analyze how our method performs when starting from a model pre-trained on ImageNet.

Ablation study on fixing different layers. We first evaluate the benefit of fixing the bottom layers of this pre-trained model. The first row of Table 5.4 shows the results when all layers are tunable. If we fix the first convolutional layer, the performance improves for task-agnostic classification, suggesting that fixing some shared layers

Table 5.4 – Ablation study of fixing parameters of residual blocks on CUB-200-2011 for 4-task scenario.

	Task-agnostic					Task-aware				
	T1	T2	T3	T4	Ave	T1	T2	T3	T4	Ave
Free	87.9	67.8	61.9	58.9	69.1	87.9	82.9	82.8	82.6	84.0
+Fix Conv	87.9	68.7	62.9	59.6	69.8	87.9	82.7	82.5	83.0	84.0
+Fix Block 1	87.3	67.9	63.3	60.0	69.6	87.3	82.8	82.5	83.2	84.0
+Fix Block 2	86.9	69.2	63.8	60.1	70.0	86.9	83.7	83.3	83.6	84.4
+Fix Block 3	87.5	71.0	65.5	61.2	**71.3**	87.5	83.9	83.8	83.3	**84.6**
Fix All	76.5	65.0	60.6	58.0	65.0	76.5	76.5	78.6	79.4	77.8

Table 5.5 – Comparison with other methods on CUB-200-2011 dataset for 4-task scenario

	Task-agnostic					Task-aware				
	T1	T2	T3	T4	Ave	T1	T2	T3	T4	Ave
LwF		65.0	50.1	36.8	59.9		82.9	78.0	69.0	79.4
EWC		55.2	47.2	34.3	56.1		79.4	77.3	65.9	77.5
MAS	87.5	62.5	59.3	54.8	66.0	87.5	86.6	84.9	84.7	**85.9**
Feature Distillation + Gaussian		68.3	63.4	59.4	69.6		83.0	82.7	82.5	83.9
Feature Distillation + GAN		71.0	65.5	61.2	**71.3**		83.9	83.8	83.3	84.6

could yield improvement. Going forward, fixing different blocks in ResNet shows that fixing the layers until Block 4 outperforms other variants in the task-aware setting. The last row shows that by fixing the feature extractor (like FearNet), the accuracy drops dramatically. This suggests that fine tuning the feature extractor is good for continual learning. We use the best configuration here to do the following experiments.

4-task scenario As shown in Table 5.5, our methods with both Gaussian and GAN replay surpass all other three methods by a large margin in the task-agnostic setting. However, MAS is slightly better than ours in the task-aware setting by 1.3%. Considering that MAS is optimized to have better accuracy in this case, our method performs reasonable good. It is also true that LwF and EWC perform much better in the case of task-aware setting than in task-agnostic.

5.6 Conclusions

In this chapter, we proposed a novel continual learning framework that combines gererative feature replay and feature distillation. We showed that it is suitable for both task-aware and task-agnostic classification, and that it is computationally efficient and scalable. Our analysis via CCA shows how catastrophic interference

and forgetting manifest at different layers. Feature distillation seems to benefit more than probability distillation from regularization at lower depths and therefore seems to be a more strict constraint that prevents intermediate and lower layers from changing dramatically. However, the regularization strength term must be carefully chosen to avoid limiting the ability to learn new tasks. The strength of our approach relies on the fact that the distribution of high-level features is significantly simpler than the distribution at the pixel level and therefore can be effectively trained with simpler generators and limited samples.

6 Conclusions and Future Work

6.1 Conclusions

Although deep CNNs have achieved superior performance on many computer vison tasks, it is still challenging for them to generalize well in small domains and to adapt learned knowledge to new domains without forgetting the previous knowledge. In this thesis, we aim at improving CNN performance in applications with limited data and at adapting learned models to new domains. For the first part, in chapter 2 we studied how learning from rankings can be used as a proxy task to leverage unlabelled data, how rankings can be learned efficiently, and how rankings can be applied to active learning. Then in chapter 3 we applied our learning from rankings approach to two different applications: image quality assessment and crowd counting. In the second part of the thesis, we focused on adapting models in sequential task learning scenarios. Two methods were proposed to mitigate catastrophic forgetting, one regularization-based and the other rehearsal-based in chapters 4 and 5, respectively.

The methods proposed and the results obtained in this thesis are:

- **Chapter 2: Self-supervised Learning to Rank.** We show how ranking can be used as a proxy task for some regression problems. As another contribution, we propose an efficient backpropagation technique for Siamese networks which prevents the redundant computation introduced by multi-branch network architecture. Additionally we demonstrate that rankings can be used as a measurement of importance of the samples to annotate.

- **Chapter 3: Applications to Image Quality Assessment and Crowd Counting.** We apply our learning-to-rank framework to two regression problems: image quality assessment (IQA) and crowd counting. For both we show how to automatically generate ranked image sets from unlabeled data. Our results show that networks trained to regress to the ground truth targets for labeled data and to simultaneously learn to rank unlabeled data obtain significantly better, state-of-the-art results for both IQA and crowd counting. When we apply it on active learning, it reduces labeling effort by up to 50% for both applications.

- **Chapter 4: Rotated Elastic Weight Consolidation for Less Catastrophic Forgetting.** One limitations of the popular method Elastic Weight Consolidation is that the Fisher Information Matrix is assumed to be a diagonal matrix. We address this limitation and propose to rotate the parameter space in order to approximately diagonalizes the of the network parameters. Experimental results on the MNIST, CIFAR-100, CUB-200 and Stanford-40 datasets demonstrate that we significantly improve the results of standard elastic weight consolidation, and that we obtain competitive results when compared to other state-of-the-art in lifelong learning without forgetting.
- **Chapter 5: Generative Feature Replay for Task-agnostic Continual Learning.** We split the network in two parts, a feature extractor network and a classifier network. We visualize and analyze how and where the network is forgetting, and, based on this analysis, we design a hybrid model which combines generative feature replay at the classifier level and distillation in the feature extractor. We show that this effectively mitigates forgetting and is computationally efficient and scalable.

6.2 Future work

For future work we are interested in extending the idea of learning-to-rank to other regression problems where ranked images can be easily generated. These fields include depth estimation where we can use the fact that the depth estimate of a zoomed-in version of the image (rescaled to the same size) should be less than the original. Another application could be equipment calibration. For example MRI machine calibration, which is done after fixed periods to verify the SNR of the system. The measurement of which is ranked since the system is getting worse through time.

Moreover, fusing different self-supervised learning methods and supervised learning applications is a promising field to explore, We believe that self-supervised learning could largely reduce the labelling cost and improve overall performance.

For continual learning, the task-agnostic setting is more challenging than task-aware setting. It would be interesting to explore further the way networks train with unbalanced data, which could connect to few-shot and zero-shot learning. Finally, self-supervised learning has the nature of self-supervision and forgetting is less likely to happen. Therefore, self-supervised learning could potentially be applied to continual learning.

Summary of published works

1. **Xialei Liu**, Joost van de Weijer and Andrew D. Bagdanov. RankIQA: Learning from Rankings for No-reference Image Quality Assessment. In Proceedings of the IEEE International Conference on Computer Vision, pp. 1040-1049. 2017.

2. **Xialei Liu**, Joost van de Weijer and Andrew D. Bagdanov. Leveraging Unlabeled Data for Crowd Counting by Learning to Rank. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7661-7669. 2018.

3. **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), pp. 2262-2268. IEEE, 2018.

4. Chenshen Wu, Luis Herranz, **Xialei Liu**, Joost van de Weijer, Yaxing Wang and Bogdan Raducanu. Memory Replay GANs: learning to generate images from new categories without forgetting. In Advances In Neural Information Processing Systems, pp. 5962-5972. 2018.

5. **Xialei Liu**, Joost van de Weijer and Andrew D. Bagdanov. Exploiting Unlabeled Data in CNNs by Self-supervised Learning to Rank. IEEE transactions on pattern analysis and machine intelligence (2019).

6. Lu Yu, Vacit Oguz Yazici, **Xialei Liu**, Joost van de Weijer, Yongmei Cheng, and Arnau Ramisa. Learning metrics from teachers: Compact networks for image embedding. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 2907–2916, 2019.

A An appendix

We synthetically generate a range of distortions from reference images. The types of distortions generated depend on the target test dataset.

The TID2013 dataset consists of 25 reference images with 3000 distorted images from 24 different distortion types at 5 degradation levels. Mean Opinion Scores are in the range [0, 9]. Distortion types include a range of noise, compression, and transmission artifacts. We generate 17 out of the 24 distortions for training our networks. For the distortions which we could not generate, we apply fine-tuning from the network trained from the other ones. The generations details are as follows (distortions in **bold** are synthetically generated, while those in normal typeface we do not generate):

- **#01 additive white Gaussian noise**: The local variance of the Gaussian noise added in RGB color space is set to be [0.001, 0.005, 0.01, 0.05].
- **#02 additive noise in color components**: The local variance of the Gaussian noise added in the YCbCr color space is set to be [0.0140, 0.0198, 0.0343, 0.0524].
- #03 additive Gaussian spatially correlated noise: there was insufficient detail in the original TID2013 paper [109] about how spatially correlated noise was generated and added to reference images.
- #04 masked noise: there was insufficient detail in the original TID2013 paper [109] about how masks were generated.
- **#05 high frequency noise**: The local variance of the Gaussian noise added in the Fourier domain is set to be [0.001, 0.005, 0.01, 0.05] after which it is multiplied by a high-pass filter.
- **#06 impulse noise**: The local variance of “salt & pepper” noise added in RGB color space is set to be [0.005, 0.01, 0.05, 0.1].
- **#07 quantization noise**: The quantization step is set to be [27, 39, 55, 76].

- **#08 Gaussian blur**: 2D circularly symmetric Gaussian blur kernels are applied with standard deviations set to be [1.2, 2.5, 6.5, 15.2].
- **#09 image denoising**: The local variance of the Gaussian noise added in RGB color space is [0.001, 0.005, 0.01, 0.05]. Followed by the same denoising process as in [20].
- **#10 JPEG compression**: The quality factor that determines the DCT quantization matrix is set to be [43, 12, 7, 4].
- **#11 JPEG2000 compression**: The compression ratio is set to be [52, 150, 343, 600].
- #12 JPEG transmission errors: the precise details of how JPEG transmission errors were introduced was not clear and we were unable to reproduce this distortion type.
- #13 JPEG2000 transmission errors: the precise details of how JPEG2000 transmission errors were introduced was not clear and we were unable to reproduce this distortion type.
- **#14 non eccentricity pattern noise**: Patches of size 15x15 are randomly moved to nearby regions [109]. The number of patches is set to [30, 70, 150, 300].
- **#15 local blockwise distortion of different intensity**: Image patches of 32x32 are replaced by single color value (color block) [109]. The number of color blocks we distort is set to be [2, 4, 8, 16].
- **#16 mean shift**: Mean value shifting generated in both directions is set to be: [-60,-45,-30,-15] and [15, 30, 45, 60].
- **#17 contrast change**: Contrast change generated in both directions is set to be: [0.85, 0.7, 0.55, 0.4] and [1.2, 1.4, 1.6, 1.8].
- **#18 change of color saturation**: The control factor as in TID2013 paper [109] is set to be: [0.4, 0, -0.4, -0.8].
- **#19 multiplicative Gaussian noise**: The local variance of the Gaussian noise added is set to be [0.05, 0.09,0.13, 0.2].
- #20 comfort noise: the authors of [109] used a proprietary encoder unavailable to us.

- #21 lossy compression of noisy images: the authors of [109] used a proprietary encoder unavailable to us.
- **#22 image color quantization with dither**: The quantization step is set to be: [64, 32, 16, 8].
- **#23 chromatic aberrations**: The mutual shifting of in R and B channels is set to be [2, 6, 10, 14] and [1, 3, 5, 7], respectively.
- #24 sparse sampling and reconstruction: the authors of [109] used a proprietary encoder unavailable to us.

Bibliography

[1] 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*, pages 139–154, 2018.

[2] Rahaf Aljundi, Punarjay Chakravarty, and Tinne Tuytelaars. Expert gate: Lifelong learning with a network of experts. In *Computer Vision and Pattern Recognition*, 2017.

[3] Shun-Ichi Amari. Natural gradient works efficiently in learning. *Neural computation*, 10(2):251–276, 1998.

[4] Relja Arandjelovic and Andrew Zisserman. Look, listen and learn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 609–617, 2017.

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

[6] Deepak Babu Sam, Shiv Surya, and R. Venkatesh Babu. Switching convolutional neural network for crowd counting. In *Computer Vision and Pattern Recognition*, 2017.

[7] Simone Bianco, Luigi Celona, Paolo Napoletano, and Raimondo Schettini. On the use of deep learning for blind image quality assessment. *Signal, Image and Video Processing*, 12(2):355–362, 2018.

[8] Lokesh Boominathan, Srinivas SS Kruthiventi, and R Venkatesh Babu. Crowdnet: a deep convolutional network for dense crowd counting. In *Proc. ACM on Multimedia Conference*, 2016.

[9] Sebastian Bosse, Dominique Maniry, Thomas Wiegand, and Wojciech Samek. A deep neural network for image quality assessment. In *International Conference on Image Processing*, pages 3773–3777. IEEE, 2016.

[10] Andrew Brock, Jeff Donahuey, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2019.

[11] Xinkun Cao, Zhipeng Wang, Yanyun Zhao, and Fei Su. Scale aggregation network for accurate and efficient crowd counting. In *Proceedings of the European Conference on Computer Vision*, pages 734–750, 2018.

[12] Francisco M Castro, Manuel J Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari. End-to-end incremental learning. In *Proceedings of the European Conference on Computer Vision*, pages 233–248, 2018.

[13] Antoni B Chan, Zhang-Sheng John Liang, and Nuno Vasconcelos. Privacy preserving crowd monitoring: Counting people without people models or tracking. In *Computer Vision and Pattern Recognition*, 2008.

[14] Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. *arXiv preprint arXiv:1801.10112*, 2018.

[15] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with A-GEM. In *International Conference on Learning Representations*, 2019.

[16] Ting Chen, Xiaohua Zhai, Marvin Ritter, Mario Lucic, and Neil Houlsby. Self-supervised gans via auxiliary rotation loss. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 12154–12163, 2019.

[17] Wei Chen, Tie-Yan Liu, Yanyan Lan, Zhi-Ming Ma, and Hang Li. Ranking measures and loss functions in learning to rank. In *Advances in neural information processing systems*, 2009.

[18] Zhiyuan Chen and Bing Liu. Lifelong machine learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 10(3):1–145, 2016.

[19] Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *Computer Vision and Pattern Recognition*, 2005.

[20] Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Image denoising by sparse 3-d transform-domain collaborative filtering. *IEEE Transactions on image processing*, 16(8):2080–2095, 2007.

[21] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. Continual learning: A comparative study on how to defy forgetting in classification tasks. *arXiv preprint arXiv:1909.08383*, 2019.

[22] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition*, pages 248–255. IEEE, 2009.

[23] Guillaume Desjardins, Karen Simonyan, Razvan Pascanu, et al. Natural neural networks. In *Advances in neural information processing systems*, pages 2071–2079, 2015.

[24] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[25] Prithviraj Dhar, Rajat Vikram Singh, Kuan-Chuan Peng, Ziyan Wu, and Rama Chellappa. Learning without memorizing. *ArXiv*, abs/1811.08051, 2018.

[26] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *International Conference on Computer Vision*, 2015.

[27] James T Enns. *The thinking eye, the seeing brain: Explorations in visual cognition*. Recording for the Blind & Dyslexic, 2005.

[28] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

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

[30] Min Fu, Pei Xu, Xudong Li, Qihe Liu, Mao Ye, and Ce Zhu. Fast crowd density estimation with convolutional neural networks. *Engineering Applications of Artificial Intelligence*, 43:81–88, 2015.

[31] Fei Gao, Dacheng Tao, Xinbo Gao, and Xuelong Li. Learning to rank for blind image quality assessment. *Neural Networks and Learning Systems, IEEE Transactions on*, 26(10):2275–2290, 2015.

[32] Spyros Gidaris and Nikos Komodakis. Dynamic few-shot visual learning without forgetting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4367–4375, 2018.

[33] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. *arXiv preprint arXiv:1803.07728*, 2018.

[34] Lluis Gomez, Yash Patel, Marçal Rusiñol, Dimosthenis Karatzas, and CV Jawahar. Self-supervised learning of visual features through embedding images into text topic spaces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4230–4239, 2017.

[35] Roger Grosse and James Martens. A kronecker-factored approximate fisher matrix for convolution layers. In *ICML*, pages 573–582, 2016.

[36] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of Wasserstein GANs. In *Advances in neural information processing systems*, pages 5767–5777, 2017.

[37] Raia Hadsell, Pierre Sermanet, Jan Ben, Ayse Erkan, Marco Scoffier, Koray Kavukcuoglu, Urs Muller, and Yann LeCun. Learning long-range vision for autonomous off-road driving. *Journal of Field Robotics*, 26(2):120–144, 2009.

[38] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[39] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Computer Vision and Pattern Recognition*, 2016.

[40] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[41] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *NeurIPS Deep Learning and Representation Learning Workshop*, 2015.

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

[43] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[44] T Huang. Computer vision: Evolution and promise. 1996.

[45] Ferenc Huszár. Note on the quadratic penalties in elastic weight consolidation. *Proceedings of the National Academy of Sciences*, 115(11):E2496–E2497, 2018.

[46] Haroon Idrees, Imran Saleemi, Cody Seibert, and Mubarak Shah. Multi-source multi-scale counting in extremely dense crowd images. In *Computer Vision and Pattern Recognition*, 2013.

[47] Haroon Idrees, Muhmmad Tayyab, Kishan Athrey, Dong Zhang, Somaya Al-Maadeed, Nasir Rajpoot, and Mubarak Shah. Composition loss for counting, density map estimation and localization in dense crowds. In *Proceedings of the European Conference on Computer Vision*, 2018.

[48] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

[49] Longlong Jing and Yingli Tian. Self-supervised visual feature learning with deep neural networks: A survey. *arXiv preprint arXiv:1902.06162*, 2019.

[50] Di Kang, Debarun Dhar, and Antoni B Chan. Crowd counting by adapting convolutional neural networks with side information. *arXiv preprint arXiv:1611.06748*, 2016.

[51] Le Kang, Peng Ye, Yi Li, and David Doermann. Convolutional neural networks for no-reference image quality assessment. In *Computer Vision and Pattern Recognition*, pages 1733–1740, 2014.

[52] Le Kang, Peng Ye, Yi Li, and David Doermann. Simultaneous estimation of image quality and distortion via multi-task convolutional neural networks. In *International Conference on Image Processing*, pages 2791–2795. IEEE, 2015.

[53] 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*, 2018.

[54] Ronald Kemker and Christopher Kanan. Fearnet: Brain-inspired model for incremental learning. *International Conference on Learning Representations*, 2018.

[55] 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*, pages 3521–3526, 2017.

[56] Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

[57] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, 2009.

[58] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[59] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Tom Duerig, and Vittorio Ferrari. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *arXiv:1811.00982*, 2018.

[60] Jean Lafond, Nicolas Vasilache, and Leon Bottou. Diagonal rescaling for neural networks. *arXiv preprint arXiv:1705.09319*, 2017.

[61] Issam H Laradji, Negar Rostamzadeh, Pedro O Pinheiro, David Vazquez, and Mark Schmidt. Where are the blobs: Counting by localization with point supervision. In *Proceedings of the European Conference on Computer Vision*, 2018.

[62] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Colorization as a proxy task for visual understanding. In *Computer Vision and Pattern Recognition*, 2017.

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

[64] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[65] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4681–4690, 2017.

[66] Jeongtae Lee, Jaehong Yun, Sungju Hwang, and Eunho Yang. Lifelong learning with dynamically expandable networks. In *International Conference on Learning Representations*, 2018.

[67] Sang-Woo Lee, Jin-Hwa Kim, Jaehyun Jun, Jung-Woo Ha, and Byoung-Tak Zhang. Overcoming catastrophic forgetting by incremental moment matching. In *Advances in neural information processing systems*, pages 4655–4665, 2017.

[68] David D Lewis and Jason Catlett. Heterogeneous uncertainty sampling for supervised learning. In *Machine Learning Proceedings 1994*, pages 148–156. Elsevier, 1994.

[69] Jinyu Li. Restructuring of deep neural network acoustic models with singular value decomposition. In *Interspeech*, January 2013.

[70] Yuhong Li, Xiaofan Zhang, and Deming Chen. Csrnet: Dilated convolutional neural networks for understanding the highly congested scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1091–1100, 2018.

[71] Zhizhong Li and Derek Hoiem. Learning without forgetting. *pami*, 40(12):2935–2947, 2018.

[72] Yudong Liang, Jinjun Wang, Xingyu Wan, Yihong Gong, and Nanning Zheng. Image quality assessment using similar scene as reference. In *European Conference on Computer Vision*, pages 3–18. Springer, 2016.

[73] Jiang Liu, Chenqiang Gao, Deyu Meng, and Alexander G Hauptmann. Decidenet: Counting varying density crowds through attention guided detection and density estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5197–5206, 2018.

[74] Tie-Yan Liu et al. Learning to rank for information retrieval. *Foundations and Trends® in Information Retrieval*, 3(3):225–331, 2009.

[75] Xialei Liu. Learning from rankings for no-reference image quality assessment by siamese network. *Master thesis of the Universitat Autonoma de Barcelona*, 2016.

[76] 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 *International Conference on Pattern Recognition*, 2018.

[77] Xialei Liu, Joost van de Weijer, and Andrew D Bagdanov. Rankiqa: Learning from rankings for no-reference image quality assessment. In *International Conference on Computer Vision*, 2017.

[78] Xialei Liu, Joost van de Weijer, and Andrew D Bagdanov. Leveraging unlabeled data for crowd counting by learning to rank. In *Computer Vision and Pattern Recognition*, 2018.

[79] Xialei Liu, Joost Van De Weijer, and Andrew D Bagdanov. Exploiting unlabeled data in cnns by self-supervised learning to rank. *IEEE transactions on pattern analysis and machine intelligence*, 2019.

[80] David Lopez-Paz and Marc Aurelio Ranzato. Gradient episodic memory for continual learning. In *Advances in neural information processing systems*, pages 6470–6479, 2017.

[81] Mario Lucic, Karol Kurach, Marcin Michalski, Sylvain Gelly, and Olivier Bousquet. Are GANs created equal. *A Large-Scale Study. ArXiv e-prints*, 2(4), 2017.

[82] Mario Lucic, Michael Tschannen, Marvin Ritter, Xiaohua Zhai, Olivier Bachem, and Sylvain Gelly. High-fidelity image generationwith fewer labels. In *ICML*, 2019.

[83] Kede Ma, Wentao Liu, Tongliang Liu, Zhou Wang, and Dacheng Tao. dipiq: Blind image quality assessment by learning-to-rank discriminable image pairs. *IEEE Transactions on Image Processing*, 26(8):3951–3964, 2017.

[84] Kede Ma, Qingbo Wu, Zhou Wang, Zhengfang Duanmu, Hongwei Yong, Hongliang Li, and Lei Zhang. Group MAD competition – a new methodology to compare objective image quality models. In *Computer Vision and Pattern Recognition*, 2016.

[85] David JC MacKay. A practical bayesian framework for backpropagation networks. *Neural computation*, 4(3):448–472, 1992.

[86] Aravindh Mahendran, James Thewlis, and Andrea Vedaldi. Cross pixel optical-flow similarity for self-supervised learning. In *Asian Conference on Computer Vision*, pages 99–116. Springer, 2018.

[87] 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*, pages 67–82, 2018.

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

[89] Mark Marsde, Kevin McGuinness, Suzanne Little, Ciara E Keogh, and Noel E O'Connor. People, penguins and petri dishes: adapting object counting models to new visual domains and object types without forgetting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[90] Marc Masana, Joost van de Weijer, Luis Herranz, Andrew D Bagdanov, and Jose M Alvarez. Domain-adaptive deep network compression. In *International Conference on Computer Vision*, 2017.

[91] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. *Psychology of learning and motivation*, 24:109–165, 1989.

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

[93] Anish Mittal, Anush Krishna Moorthy, and Alan Conrad Bovik. No-reference image quality assessment in the spatial domain. *Image Processing, IEEE Transactions on*, 21(12):4695–4708, 2012.

[94] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018.

[95] Takeru Miyato and Masanori Koyama. cGANs with projection discriminator. In *International Conference on Learning Representations*, 2018.

[96] Anush Krishna Moorthy and Alan Conrad Bovik. Blind image quality assessment: From natural scene statistics to perceptual quality. *IEEE Transactions on Image Processing*, 20(12):3350–3364, 2011.

[97] Cuong V Nguyen, Yingzhen Li, Thang D Bui, and Richard E Turner. Variational continual learning. In *International Conference on Learning Representations*, 2018.

[98] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European Conference on Computer Vision*, pages 69–84. Springer, 2016.

[99] Mehdi Noroozi, Hamed Pirsiavash, and Paolo Favaro. Representation learning by learning to count. In *International Conference on Computer Vision*, 2017.

[100] Daniel Onoro-Rubio and Roberto J López-Sastre. Towards perspective-free object counting with deep learning. In *European Conference on Computer Vision*. Springer, 2016.

[101] 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 Conference on Computer Vision and Pattern Recognition*, pages 11321–11329, 2019.

[102] Dim P Papadopoulos, Jasper RR Uijlings, Frank Keller, and Vittorio Ferrari. We don't need no bounding-boxes: Training object class detectors using only human verification. In *Computer Vision and Pattern Recognition*, 2016.

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

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

[105] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in PyTorch. In *Advances in neural information processing systems Autodiff Workshop*, 2017.

[106] Yash Patel, Lluis Gomez, Raul Gomez, Marçal Rusiñol, Dimosthenis Karatzas, and CV Jawahar. Texttopicnet-self-supervised learning of visual features through embedding images on semantic text spaces. *arXiv preprint arXiv:1807.02110*, 2018.

[107] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *Computer Vision and Pattern Recognition*, 2016.

[108] Lerrel Pinto, James Davidson, and Abhinav Gupta. Supervision via competition: Robot adversaries for learning tasks. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1601–1608. IEEE, 2017.

[109] Nikolay Ponomarenko, Oleg Ieremeiev, Vladimir Lukin, Karen Egiazarian, Lina Jin, Jaakko Astola, Benoit Vozel, Kacem Chehdi, Marco Carli, Federica

Battisti, et al. Color image database tid2013: Peculiarities and preliminary results. In *Visual Information Processing (EUVIP), 2013 4th European Workshop on*, pages 106–111. IEEE, 2013.

[110] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *International Conference on Learning Representations*, 2015.

[111] Maithra Raghu, Justin Gilmer, Jason Yosinski, and Jascha Sohl-Dickstein. Svcca: Singular vector canonical correlation analysis for deep learning dynamics and interpretability. In *Advances in Neural Information Processing Systems*, pages 6076–6085, 2017.

[112] Viresh Ranjan, Hieu Le, and Minh Hoai. Iterative crowd counting. In *Proceedings of the European Conference on Computer Vision*, 2018.

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

[114] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Computer Vision and Pattern Recognition*, pages 5533–5542. IEEE, 2017.

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

[116] 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.

[117] Michele A Saad, Alan C Bovik, and Christophe Charrier. Blind image quality assessment: A natural scene statistics approach in the dct domain. *Image Processing, IEEE Transactions on*, 21(8):3339–3352, 2012.

[118] Tobias Scheffer, Christian Decomain, and Stefan Wrobel. Active hidden markov models for information extraction. In *International Symposium on Intelligent Data Analysis*, pages 309–318. Springer, 2001.

[119] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Computer Vision and Pattern Recognition*, pages 815–823, 2015.

[120] D Sculley. Large scale learning to rank. In *Advances in neural information processing systems Workshop on Advances in Ranking*, pages 1–6, 2009.

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

[122] Pierre Sermanet, Corey Lynch, Yevgen Chebotar, Jasmine Hsu, Eric Jang, Stefan Schaal, Sergey Levine, and Google Brain. Time-contrastive networks: Self-supervised learning from video. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1134–1141. IEEE, 2018.

[123] Joan Serrà, Dídac Surís, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *Proceedings of International Conference on Machine Learning*, 2018.

[124] Burr Settles. Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 6(1):1–114, 2012.

[125] Burr Settles, Mark Craven, and Lewis Friedland. Active learning with real annotation costs. In *Proceedings of the Neural Information Processing Systems workshop on cost-sensitive learning*, pages 1–10. Vancouver, Canada, 2008.

[126] Burr Settles, Mark Craven, and Soumya Ray. Multiple-instance active learning. In *Neural Information Processing Systems*, pages 1289–1296, 2008.

[127] H. R. Sheikh, Z. Wang, L. Cormack, and A. C. Bovik. Live image quality assessment database. http://live.ece.utexas.edu/research/quality.

[128] Hamid Rahim Sheikh, Muhammad Farooq Sabir, and Alan Conrad Bovik. A statistical evaluation of recent full reference image quality assessment algorithms. *Image Processing, IEEE Transactions on*, 15(11):3440–3451, 2006.

[129] Zan Shen, Yi Xu, Bingbing Ni, Minsi Wang, Jianguo Hu, and Xiaokang Yang. Crowd counting via adversarial cross-scale consistency pursuit. In *Computer Vision and Pattern Recognition*, pages 5245–5254, 2018.

[130] Zenglin Shi, Le Zhang, Yun Liu, Xiaofeng Cao, Yangdong Ye, Ming-Ming Cheng, and Guoyan Zheng. Crowd counting with deep negative correlation learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5382–5390, 2018.

[131] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. In *Advances in Neural Information Processing Systems*, pages 2994–3003, 2017.

[132] Konstantin Shmelkov, Cordelia Schmid, and Karteek Alahari. Incremental learning of object detectors without catastrophic forgetting. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3400–3409, 2017.

[133] Daniel Silver and Robert Mercer. The task rehearsal method of life-long learning: Overcoming impoverished data. *Advances in Artificial Intelligence*, pages 90–101, 2002.

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

[135] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *International Conference on Learning Representations*, 2015.

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

[137] Vishwanath A Sindagi and Vishal M Patel. A survey of recent advances in cnn-based single image crowd counting and density estimation. *Pattern Recognition Letters*, 2017.

[138] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In *Advances in neural information processing systems*, pages 1857–1865, 2016.

[139] Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding. In *Computer Vision and Pattern Recognition*, 2016.

[140] Rupesh K Srivastava, Jonathan Masci, Sohrob Kazerounian, Faustino Gomez, and Jürgen Schmidhuber. Compete to compute. In *Advances in neural information processing systems*, pages 2310–2318, 2013.

[141] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[142] Ruben Villegas, Jimei Yang, Seunghoon Hong, Xunyu Lin, and Honglak Lee. Decomposing motion and content for natural video sequence prediction. In *International Conference on Learning Representations*, 2017.

[143] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. (CNS-TR-2011-001), 2011.

[144] Elad Walach and Lior Wolf. Learning to count with cnn boosting. In *European Conference on Computer Vision*. Springer, 2016.

[145] Chuan Wang, Hua Zhang, Liang Yang, Si Liu, and Xiaochun Cao. Deep people counting in extremely dense crowds. In *Proceedings of ACM int. conf. on Multimedia*. ACM, 2015.

[146] Xiaolong Wang and Abhinav Gupta. Unsupervised learning of visual representations using videos. In *International Conference on Computer Vision*, pages 2794–2802, 2015.

[147] Zhou Wang, Alan C Bovik, and Ligang Lu. Why is image quality assessment so difficult? In *Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE International Conference on*, volume 4, pages IV–3313. IEEE, 2002.

[148] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[149] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona. Caltech-UCSD Birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology, 2010.

[150] 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*, 2018.

[151] Yue Wu, Yan-Jia Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume abs/1905.13260, 2019.

[152] Yongqin Xian, Tobias Lorenz, Bernt Schiele, and Zeynep Akata. Feature generating networks for zero-shot learning. In *Computer Vision and Pattern Recognition*, pages 5542–5551, 2018.

[153] Yongqin Xian, Saurabh Sharma, Bernt Schiele, and Zeynep Akata. f-VAEGAN-D2: A feature generating framework for any-shot learning. In *Computer Vision and Pattern Recognition*, 2019.

[154] Jingtao Xu, Peng Ye, Qiaohong Li, Haiqing Du, Yong Liu, and David Doermann. Blind image quality assessment based on high order statistics aggregation. *IEEE Transactions on Image Processing*, 25(9):4444–4457, 2016.

[155] Long Xu, Jia Li, Weisi Lin, Yongbing Zhang, Lin Ma, Yuming Fang, and Yihua Yan. Multi-task rank learning for image quality assessment. *IEEE Transactions on Circuits and Systems for Video Technology*, 27(9):1833–1843, 2017.

[156] Lin Yang, Yizhe Zhang, Jianxu Chen, Siyuan Zhang, and Danny Z Chen. Suggestive annotation: A deep active learning framework for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 399–407. Springer, 2017.

[157] Bangpeng Yao, Xiaoye Jiang, Aditya Khosla, Andy Lai Lin, Leonidas Guibas, and Li Fei-Fei. Human action recognition by learning bases of action attributes and parts. In *International Conference on Computer Vision*, pages 1331–1338, 2011.

[158] Peng Ye, Jayant Kumar, Le Kang, and David Doermann. Unsupervised feature learning framework for no-reference image quality assessment. In *Computer Vision and Pattern Recognition*, pages 1098–1105. IEEE, 2012.

[159] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *ICML*, pages 3987–3995, 2017.

[160] Mengyao Zhai, Lei Chen, Fred Tung, Jiawei He, Megha Nawhal, and Greg Mori. Lifelong gan: Continual learning for conditional image generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[161] Cong Zhang, Hongsheng Li, Xiaogang Wang, and Xiaokang Yang. Cross-scene crowd counting via deep convolutional neural networks. In *Computer Vision and Pattern Recognition*, 2015.

[162] Lin Zhang, Lei Zhang, Xuanqin Mou, and David Zhang. Fsim: a feature similarity index for image quality assessment. *IEEE transactions on Image Processing*, 20(8):2378–2386, 2011.

[163] Peng Zhang, Wengang Zhou, Lei Wu, and Houqiang Li. Som: Semantic obviousness metric for image quality assessment. In *Computer Vision and Pattern Recognition*, pages 2394–2402, 2015.

[164] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *European Conference on Computer Vision*, 2016.

[165] Yingying Zhang, Desen Zhou, Siqin Chen, Shenghua Gao, and Yi Ma. Single-image crowd counting via multi-column convolutional neural network. In *Computer Vision and Pattern Recognition*, 2016.

[166] Bolei Zhou, Agata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. Learning deep features for scene recognition using places database. In *Advances in neural information processing systems*, pages 487–495, 2014.

[167] Zongwei Zhou, Jae Shin, Lei Zhang, Suryakanth Gurudu, Michael Gotway, and Jianming Liang. Fine-tuning convolutional neural networks for biomedical image analysis: actively and incrementally. In *Computer Vision and Pattern Recognition*, pages 7340–7349, 2017.