On Domain-Incremental Learning methods and its applications to forgery detection

by

Julien NICOLAS

THESIS PRESENTED TO ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
IN PARTIAL FULFILLMENT OF A MASTER'S DEGREE
WITH THESIS
M.A.Sc.

MONTREAL, SEPTEMBER 14, 2023

ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

BOARD OF EXAMINERS

THIS THESIS HAS BEEN EVALUATED

BY THE FOLLOWING BOARD OF EXAMINERS

M. Jose Dolz, Thesis supervisor
Département de génie logiciel et des TI, École de technologie supérieure

M. Christian Desrosiers, Thesis Co-Supervisor
Département de génie logiciel et des TI, École de technologie supérieure

M. Rafael Menelau Cruz, Chair, Board of Examiners
Département de génie logiciel et des TI, École de technologie supérieure

M. Marco Pedersoli , Member of the Jury
Département de génie des systèmes, École de technologie supérieure

THIS THESIS WAS PRESENTED AND DEFENDED

IN THE PRESENCE OF A BOARD OF EXAMINERS AND THE PUBLIC

ON AUGUST 25, 2023

AT ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

ACKNOWLEDGEMENTS

I would like to give a special thank to my thesis supervisors, Prof. Jose Dolz and Prof. Christian Desrosiers, for their guidance, patience, and very valuable insights throughout the course of this work. I also am grateful of the help given by my industrial supervisors, Dr. Ola Ahmad and Dr. Imtiaz Ziko, who provided technical expertise and practical perspectives that bridged the gap between academia and industry. I would last like to thank Dr. Florent Chiaroni, whose constructive feedback and encouragement have been really helpful throughout this work. I would also like to thank my family and my friends and colleagues at LIVIA for their warm support along the way.

Sur l’apprentissage incremental de domaine et son application à la détection de modifications dans des images numériques

Julien NICOLAS

RÉSUMÉ

Un accès facile à des appareils puissants et la diffusion rapide des réseaux sociaux ont entraîné une augmentation sans précédent de la quantité d’images numériques disponibles. Cela a facilité l’essor de la contrefaçon d’images numériques, qui peut être facilement exploitée par des criminels à des fins obscures (fraude à l’assurance, vol d’identité, etc.). Pour identifier les techniques de falsification d’images les plus répandues, des réseaux de neurones convolutifs (CNN) ont été proposés récemment dans la littérature. Néanmoins, ces approches utilisent des hypothèses fortes sur la disponibilité des données et leurs domaines. En particulier, elles supposent que i) les données d’entraînement et de test sont tirées des mêmes distributions, et ii) les domaines des données restent inchangés dans le temps. Nous soutenons que ces hypothèses peuvent cependant limiter l’applicabilité des méthodes de détection de modifications existantes à des scénarios très contraignants.

Pour pallier à ces limitations, nous présentons une nouvelle approche d’apprentissage incrémental de domaines (*Domain Incremental Learning* – DIL) basée sur un mélange de modèles CLIP adaptés par prompt (que l’on appelle MoP-CLIP), qui généralise le paradigme du S-Prompting pour gérer à la fois les données provenant de distributions connues et celles hors distribution lors de l’inférence. Au moment de l’entraînement, nous modélisons la distribution des caractéristiques de chaque classe pour chaque domaine connu, en apprenant des prompts textuels et visuels pour adapter le modèle CLIP à chaque domaine. Lors de l’inférence, les modélisations nous permettent d’identifier si une image appartient à un domaine connu et de sélectionner le bon prompt pour la tâche de classification, ou à un domaine inconnu et d’utiliser la technique à base de mélange proposée.

Notre évaluation empirique révèle les limitations des méthodes DIL existantes en présence de changement de domaine, et suggère que le modèle MoP-CLIP proposé a des performances compétitives dans le scénario standard, tout en surpassant les méthodes récentes dans des scénarios avec données hors distribution (*Out-of-distribution* – OOD). Ces résultats démontrent la supériorité de MoP-CLIP , offrant une solution robuste et générale au problème de l’apprentissage incrémental (de domaines).

Nous soulignons également que les algorithmes d’apprentissage incremental doivent être évalués sur des jeux de données représentant des problèmes réels et effectuons une évaluation de ces algorithmes et notre méthode en les appliquant au problème de détection de modifications dans des images naturelles.

Mots-clés: continual learning, prompt tuning, classification, CLIP, incremental learning

On Domain-Incremental Learning methods and its applications to forgery detection

Julien NICOLAS

ABSTRACT

The easy access to powerful devices and quick spread of social networks have led to an unprecedented increase of the amount of available digital images. This has facilitated the rise of digital image forgery, which can be leveraged easily by criminals with obscure purposes (i.e., insurance fraud, identity theft, etc). To identify the most prevalent image forgery techniques, convolutional neural networks (CNN) have been proposed recently in the literature. Nevertheless, these approaches make strong assumptions about the availability of data and its domains. In particular, they assume that i) training and testing data are drawn from the same domain distribution, and ii) the data domain remains unchanged over time. We argue that these assumptions, however, may limit the applicability of existing forgery detection methods to highly constrained scenarios.

To address these limitations, we present a novel Domain-Incremental Learning (DIL) approach based on a mixture of prompt-tuned CLIP models (MoP-CLIP), which generalizes the paradigm of S-Prompting to handle both in-distribution and out-of-distribution (OOD) data at inference. At the training stage, we model the feature distribution of every class in each domain, learning individual text and visual prompts to adapt to a given domain. At inference, the learned distributions allow us to identify whether a given test sample belongs to a known domain, selecting the correct prompt for the classification task, or from an unseen domain, leveraging a mixture of the prompt-tuned CLIP models. Our empirical evaluation reveals the limitations of existing DIL methods under domain shift, and suggests that the proposed MoP-CLIP performs competitively in the standard DIL settings while outperforming state-of-the-art methods in OOD scenarios. These results demonstrate the superiority of MoP-CLIP , offering a robust and general solution to the problem of domain incremental learning, while relaxing the assumptions previously made for data distributions.

We also emphasize that domain-incremental learning approaches must be benchmarked with challenging real-world datasets, and therefore conduct a realistic evaluation of the proposed method, as well as existing domain-incremental approaches, on a harder task, i.e., domain-incremental forgery detection, Our findings reveal that in this challenging scenario, the proposed method still yields competitive performance.

Keywords: continual learning, prompt tuning, classification, CLIP, incremental learning

TABLE OF CONTENTS

Page

INTRODUCTION 1
0.1 Limitations of Domain-Incremental Learning methods 3
0.2 Contributions and outline 4

CHAPTER 1 PROBLEM DEFINITION AND LITERATURE REVIEW 7
1.1 Problem definition 7
1.1.1 Image forgery 7
1.1.2 Feature extraction 9
1.1.3 Image forgery classification 10
1.1.4 Image manipulation segmentation 10
1.1.5 Domain-Incremental learning 11
1.2 Bibliographic study 12
1.2.1 Forgery detection 12
1.2.1.1 Classification 12
1.2.1.2 Segmentation 16
1.2.2 Domain-Incremental Learning 22
1.2.2.1 Regularization methods 22
1.2.2.2 Prompt learning 22
1.2.2.3 Prompt tuning in domain incremental learning 23
1.2.2.4 Domain generalization (DG) 24
1.2.2.5 Dynamic classifier or ensemble selection 24

CHAPTER 2 PROPOSAL: MOP-CLIP FOR GENERAL DOMAIN-INCREMENTAL LEARNING 25
2.1 Introduction 25
2.2 Method 26
2.2.1 Prompts Learning 28
2.2.2 Inference 28
2.2.2.1 Domain Selection. 29
2.2.2.2 Ensembling 30
2.3 Experiments 34
2.3.1 Experimental setup 34
2.3.1.1 Datasets 34
2.3.2 Comparison methods 35
2.3.2.1 Evaluation metrics and protocol 35
2.3.2.2 Implementation details 36
2.4 Results 37
2.4.1 In-domain distributions 37
2.4.2 Performance under domain distributional shift. 39
2.4.3 On the impact of the different components. 41

2.4.4 Strategy to select the domain prompts. 42
2.4.5 How much trade-off is sufficient? 43
2.5 Conclusion 45

CHAPTER 3 DOMAIN-INCREMENTAL FORGERY DETECTION 47
3.1 Introduction 47
3.2 Method 47
3.2.1 Domain-incremental learning methods 47
3.2.2 Forgery detection methods 48
3.3 Experiments 48
3.3.1 Metrics 48
3.3.2 Datasets 49
3.3.3 Experimental setup 49
3.3.4 Experiments 51
3.4 Results 51
3.4.1 Naive evaluation 51
3.4.2 Naive finetuning 52
3.4.3 Regularization methods 53
3.5 Conclusion 56

CONCLUSION AND RECOMMENDATIONS 57

APPENDIX I POTENTIAL CONTRIBUTIONS 59

BIBLIOGRAPHY 67

LIST OF TABLES

Page

Table 2.1 **Results on CDDB-Hard for both ID and OOD scenarios.** Evaluation of existing state-of-the-art DIL methods in the standard *seen-domain* setting and more challenging *unseen-domain* scenario. For the unseen-domain experiments, we only reproduced the results for related (i.e., *exemplar-free*) methods. Best results are highlighted in **bold**. 38

Table 2.2 **Results on DomainNet for both ID (AA metric) and OOD (CA metric) scenarios**. Best values are highlighted in **bold**. 38

Table 2.3 **Results on CORe50.** Note that CORe50 already provides separate training and testing domains, and thus results can only be computed on the **OOD scenario**. Results are reported as the Acc metric, where the best values are highlighted in **bold**. In our method, we use the same *q* as in the other datasets, whereas * indicates that *q* is fixed based on the validation set of CORe50, as typically done in all the other approaches. 40

Table 2.4 **Empirical motivation of resorting to the prediction ensembling scheme for OOD situations.** Classification accuracy across DomainNet domains using different specialized prompts, for both single and ensembling predictions. The results in blue denote the accuracy with the in-domain prompts, whereas results in magenta denote the accuracy using the best out-of-domain prompts (prompts from all domains except the current one). Furthermore, results in bold (*last column*) denote the highest accuracy amongst out-of-domain methods. For 5 out of 6 domain sets, the proposed prediction ensembling method yields higher accuracy than the best out-of-domain prompt. This suggests that the ensembling technique is overall relevant when test examples are from a novel domain (i.e. unseen during the training). 41

Table 2.5 **Impact of each design choice of MoP-CLIP .** *Maha* denotes the Mahanalobis distance, whereas GMM is used for a Gaussian Mixture Model. Furthermore, *Hybrid* denotes the nature of our approach, which uses an ensembling for OOD samples and a single domain prompt for ID samples. Results (on CDDB-Hard) show the average accuracy (AA), with the deviation from the baseline S-liPrompts Wang, Huang & Hong (2022b) in brackets. Best results in **bold**. 42

Table 3.1 **Results on iCasia for both ID (AA metric) and OOD (CA metric) scenarios**. Best values are highlighted in **bold**. 55

LIST OF FIGURES

Page

Figure 0.1 **Performance degradation under the presence of domain shift** between adaptation and testing samples, which shows that state of the art (SOTA) approaches for DIL do not generalize well. We employ SOTA domain-incremental learning S-Prompts Wang *et al.* (2022b) as use-case. The red line represents the performance across each test domain, when all domains have been seen by the model. In contrast, the blue dotted line shows the performance of the same model when the test domain remains unknown, highlighting the performance degradation under distributional shift. 4

Figure 1.1 Copy-Move 8

Figure 1.2 Splicing 8

Figure 1.3 Removal 9

Figure 1.4 Enhancement 9

Figure 2.1 **Overview of MoP-CLIP.** The training phase (*left*): class-wise prototypes are identified from in-distribution domains. Inference (*middle* and *right*): domain selection and ensembling (Mixture of Prompts), respectively, for in-distribution and out-of-distribution samples. For simplicity, we depict the pipeline for 2 classes (Real *vs* Fake). However, the procedure for multiple classes (e.g., DomainNet or CoRE50) is exactly the same. 26

Figure 2.2 **Proposed generalization scenario for domain incremental learning** Standard problem (*left*): Only in-domain examples are encountered at test time. Addressed problem (*right*): Both in-domain and out-of-domain examples are presented at test time. 27

Figure 2.3 **k-Means or class prototypes as domain centroids?** Ablation study that demonstrates the benefits of using class prototypes (our approach) rather than k-Means prototypes, as in Wang *et al.* (2022b). 43

Figure 2.4 **A controllable trade-off between in-domain and out-of-domain prediction performances.** Impact of the threshold q (Sec. 2.2.2) on the accuracy, evaluated on CDDB-Hard. 44

Figure 3.1 iCasia 50

Figure 3.2 F1 scores on $iCasia$ for image and pixel level predictions for images under different perturbations 52

Figure 3.3 Illustration of catastrophic forgetting for MVSS-Net: Scores on CASIA v1 clean shard after finetuning on domain i (x-axis) or naively evaluating the base model 53

Figure 3.4 Comparison of F1 score on clean CASIA 54

Figure 3.5 Comparison of F1 score on the i^{th} corrupted CASIA shard between training methods for OSN 55

LIST OF ABBREVIATIONS

ID	In-Domain
OOD	Out-of-Domain
MoP	Our proposed approach: Mixture of Prompts
CLIP	Contrastive Language-Image Pre-Training
DIL	Domain-Incremental Learning
KNN	K Nearest Neighbors
VPT	Visual Prompt Tuning
DG	Domain generalization
ViT	Visual Transformer
GMM	Gaussian Mixture Model
CDDB	Continual Deepfake Detection Benchmark
SOTA	State Of The Art

LIST OF SYMBOLS AND UNITS OF MEASUREMENTS

AA	Average Accuracy
AF	Average Forgetting
BWT	Backward transfer degradation
CA	Cumulative Accuracy

INTRODUCTION

Digital image forgery has become a worldwide problem, with many forms of forgery (e.g., insurance fraud, fake news, identity theft) negatively affecting our life. This effect could be attributed to two main factors. First, the accessible costs of mobile phones and digital cameras has led to an exponential proliferation of digital images. This is further magnified with the rise of social networks, where millions of users worldwide share their digital images systematically. And second, the availability of many image editing tools that allow easy manipulation of images can result in the craft of realistic-looking forgeries. While image manipulation can have marginal effects on some domains, they can have a significant impact in several other ones. Let us take, for example, the case of forgery of official ID documents, such as passports or driver licenses, where the non-detection of malicious actions may have catastrophic consequences in terms of security, privacy and economic damage. Despite the existence of many techniques for digital image forgery, copy-move and splicing are considered the most common methods for image manipulation Meena & Tyagi (2019). The main objective of forgery detection techniques is to determine whether a query image contains cloned or added regions, which will likely evidence a potential malicious intent.

Initial attempts for these techniques include the use of classical machine learning methods, such as PCA, Zernike moments (Ryu, Lee & Lee, 2010), blur moments (Mahdian & Saic, 2007), keypoint-based methods such as SIFT (Yang, Sun, Guo, Xia & Chen, 2018) or SURF (Bo, Junwen, Guangjie & Yuewei, 2010). Nevertheless, many of these traditional methods rely on strong assumptions about particular image characteristics, such as edge sharpness or local features. These assumptions are not guaranteed in forged images since transformations such as image resampling or compression might hide visible manipulation traces. This limits the use of these approaches to forgery with low level of sophistication, which are far from realistic and challenging scenarios. With the advent of deep learning, recent approaches involve end-to-end trainable solutions that can relax these assumptions (Wu, Abd-Almageed & Natarajan, 2018;

Bayar & Stamm, 2018). For example, Dang, Liu, Stehouwer, Liu & Jain (2020) proposed a dual branch deep architecture to localize potentially manipulated regions by means of visual artifacts and copy-move regions via visual similarities. Authors in (Wu, AbdAlmageed & Natarajan, 2019a) proposed a pipeline with two Siamese Convolutional Neural Networks (CNN) for image splice detection and localization in the context of fake news detection. Their proposed approach evaluated whether an image is self-consistent, adding photo meta-data as supervisory signal in addition to the photo content. More recently, a Generative Adversarial Network including an attention module was presented in (Li, Xie, Li, Wang & Zhang, 2021) to detect and localize copy-move forgeries.

Nevertheless, despite the relative success of these techniques to detect digital image manipulations, they assume that *i*) training and testing data are drawn from the same domain distribution, and that *ii*) the data domain remains unchanged over time. These assumptions, however, may limit the applicability of existing forgery detection methods to highly constrained scenarios. For instance, a forged image can be uploaded to social networks, where additional processing such as compression, blurring, lightning changes or resizing will take place. While these modifications may seem to be small in terms of distributional drift, recent evidence (Wang, Fink, Van Gool & Dai, 2022a; Song, Lee, Kweon & Choi, 2023) suggests that they are sufficient to degrade the performance of models trained on a source, unmodified domain.

A naive solution to alleviate the aforementioned issues could be to retrain the model each time a new set of labeled samples is available. Nevertheless, privacy concerns in many scenarios (e.g., forgery detection in ID documents) or higher complexity costs (e.g., retraining the model with the whole augmented dataset) make this strategy unrealistic. We also expect to devise models that are adapted to new domains or classes without forgetting their prior learned knowledge, a setting where standard transfer learning techniques are not applicable. To overcome these limitations of existing forgery detection methods Dang *et al.* (2020); Li *et al.* (2021); Dong, Chen, Hu, Cao & Li (2022), we resort to Domain-Incremental Learning approaches, which can continually learn and adapt throughout their lifetime without forgetting relevant past knowledge. Note that in the tackled scenario, the label space remains the same over time (i.e., classes do not change), but the stream of used data suffers from a continuous distributional shift.

0.1 Limitations of Domain-Incremental Learning methods

Several limitations hamper the use of Domain-Incremental Learning (DIL) methods in real-world scenarios. For instance, most state-of-the-art DIL methods perform satisfactorily in *known* domains, but typically fail when *unseen* domains are presented (see Fig. 0.1). More concretely, *known* domains refers to the scenario where a target domain A is used for adapting the model, and reported results include the performance of the model in the same domain A. In contrast, we denote as *unseen* domains, where the model is adapted with samples drawn from domain A, but it is also evaluated in a new domain B. This is particularly important in real-world scenarios where training and testing data of the *a priori* same domain may present distributional drifts that degrade the model performance. Moreover, most models see their performance drastically degrade after image resizing or downsampling. For instance, the F1 score of CAT-Net on CASIA v2 (Dong, Wang & Tan, 2013) drops from 0.7 to 0.15 after simply resizing the images.

In addition, we believe that most domain incremental learning methods are benchmarked on toy datasets, i.e. sequences of domains with domains on where it is easy to train classifiers from scratch [1], such as CORe50 Lomonaco & Maltoni (2017), Split CIFAR-100 van de Ven, Tuytelaars & Tolias (2022) or Permuted MNIST Zenke, Poole & Ganguli (2017). In contrast, forgery detection in digital images is a hard problem by itself and it is important to find solutions to alleviate the spread of forged images and, consequently, of false information that can have negative economical, societal and security impact. SOTA models specialized in the task of

[1] Note that in these datasets, it is possible to have accuracies in the range of 90-95% in a non-incremental setting.

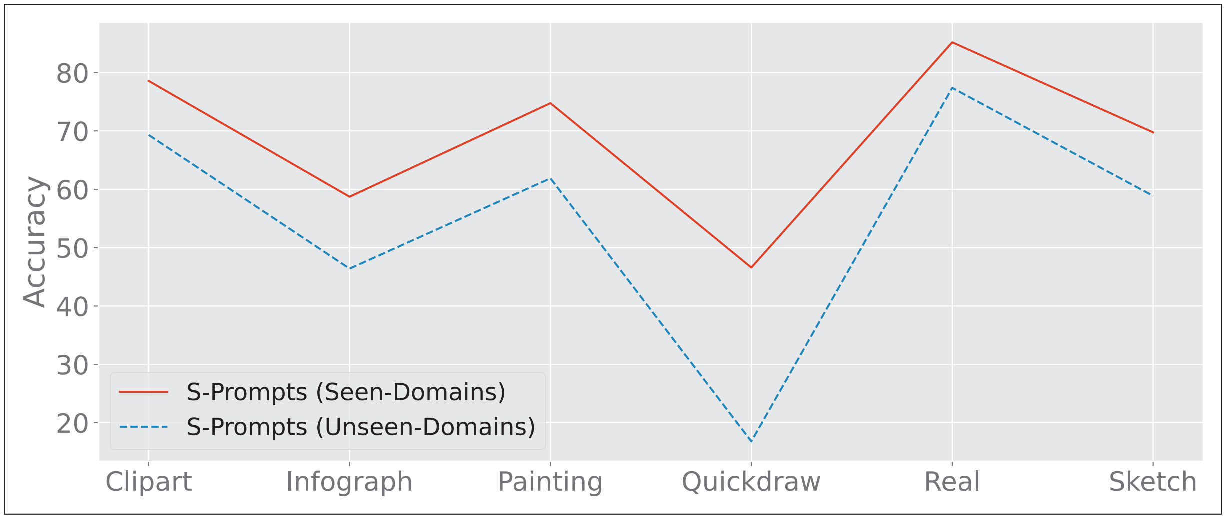


Figure 0.1 **Performance degradation under the presence of domain shift** between adaptation and testing samples, which shows that state of the art (SOTA) approaches for DIL do not generalize well. We employ SOTA domain-incremental learning S-Prompts Wang *et al.* (2022b) as use-case. The red line represents the performance across each test domain, when all domains have been seen by the model. In contrast, the blue dotted line shows the performance of the same model when the test domain remains unknown, highlighting the performance degradation under distributional shift.

forgery detection typically yield accuracies ranging from 60 to 80%, which highlights the difficulty of the task. Domain-incremental forgery detection therefore seems like a harder scenario to test domain incremental learning methods. Furthermore, most networks specialized for forgery detection use non standard architectures adapted for the task which restrict the number of domain-incremental learning methods that we can use.

0.2 Contributions and outline

Motivated by these limitations, we introduce in **Chapter 2 (2)** a novel general-purpose DIL solution, which generalizes the recent S-liPrompts approach (Wang *et al.*, 2022b) for both in-distribution (i.e., *seen domains*) and out-of-distribution (i.e., *unseen domains*) data, evaluated on common DIL benchmarks (CORe50 Lomonaco & Maltoni (2017), DomainNet Peng *et al.* (2019) and CDDB-Hard Li *et al.* (2023)). Furthermore, we perform a more realistic empirical validation, evaluating the proposed model in the context of domain-incremental forgery detection. Specifically, our contributions can be summarized as follows:

- We first expose that existing state-of-the-art domain incremental learning approaches suffer in the presence of distributional shift between samples used in the learning and testing phases, which hampers their generalization to unseen domains (Fig. 0.1).
- Based on these observations, we present a novel general DIL strategy based on a mixture of prompt-tuned (MoP) CLIP (Contrastive Language-Image Pre-Training Radford *et al.* (2021a)) models, generalizing the recent S-liPrompts approach (Wang *et al.*, 2022b) to work with both in-distribution and out-of-distribution data. In particular, the proposed approach learns class-wise features distributions for each known domain, allowing to detect whether a given inference sample comes from a known domain. We stress that the results from this contribution have been submitted to the *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV) 2024*.
- The proposed approach does not store exemplars (past training data) for replay, reducing the computational burden compared to conventional methods and alleviating potential privacy issues. Furthermore, our model is *agnostic to the sequence order*, which brings an extra level of flexibility compared to most existing DIL methods.
- Extensive experiments demonstrate that our approach performs at par with state-of-the-art general DIL methods on known domains, while largely outperforming them under distributional drifts.

In **Chapter 3 (3)**, we perform a realistic evaluation of our method and of some common incremental-learning methods in the context of domain-incremental forgery detection, apply our proposed MoP-CLIP method *without any hyperparameter tuning* on this use case and show that its competitive performance on this hard scenario as well. In particular, we advocate for the need to craft methods whose parameters transfer well to new tasks, which are adaptable to custom architectures and work on challenging classification tasks.

Last, we describe in **Appendix (I)** several perspectives for future work, which give a potential solution to these problems.

CHAPTER 1

PROBLEM DEFINITION AND LITERATURE REVIEW

1.1 Problem definition

In this section, we introduce the basic notions to understand the presented setting. In particular, we first present the most popular types of digital image forgeries in Section 1.1.1. Then, we formally define image classification (Subsection 1.1.3), segmentation (Subsection 1.1.4) and the domain incremental learning problem (Subsection 1.1.5) under the forgery detection scenario.

1.1.1 Image forgery

Image forgery refers to the manipulation of digital images to alter their semantic information and, therefore, their interpretation. It can be used maliciously, for instance, to forge proof, for political propaganda or profit. The most widely used image forgery techniques are copy-move, splicing and removal. Image forgery detection is difficult because one has to disentangle these malicious attempts and legitimate image modifications, such as compression, image resizing or contrast enhancement. Cryptographic methods have been proposed to guarantee the integrity of digital images but their seals break at the slightest modification, which is not realistic considering transmission and legitimate modifications.

- Copy-move forgery (Fig. 1.1) consists in copying a part of an image, which is then blended into the same image.
- In splicing (Fig. 1.2), a part of an image is copied and then blended into another image.
- Removal (Fig. 1.3) is generally used to hide something in a image.
- Retouching (Fig. 1.4) is used to drastically change some characteristics of the image. It can be, for instance, changing the contrast or luminosity in order to hide certain parts of the image.

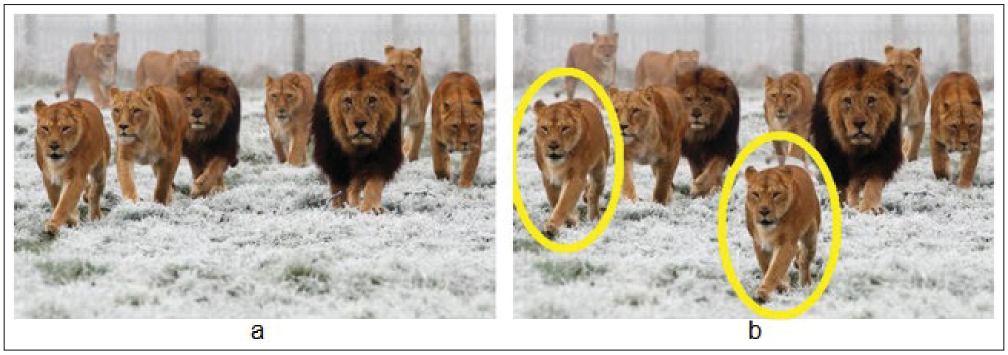


Figure 1.1 Example of copy–move forgery (a) Original image, (b) Forged image (duplicated object highlighted) (Dixit & Naskar, 2017)

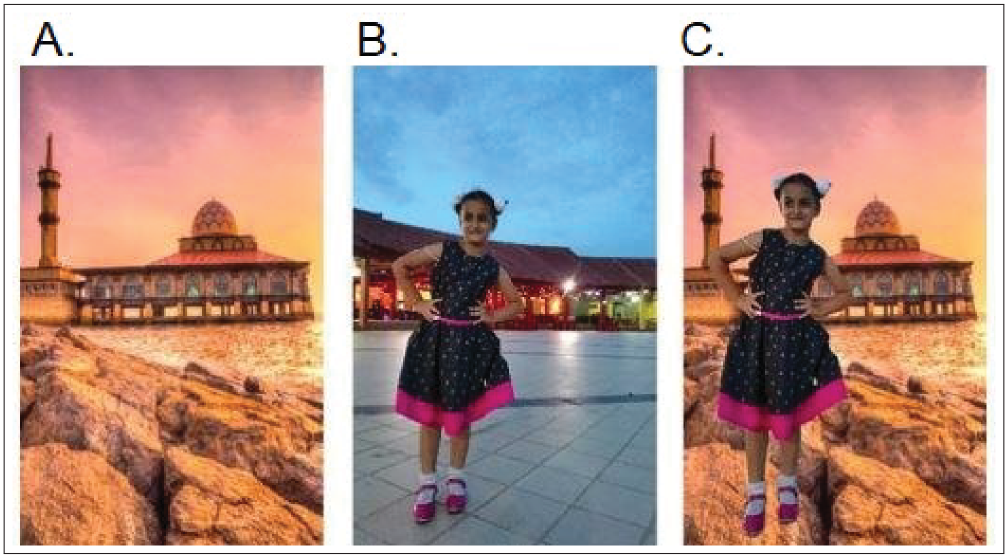


Figure 1.2 An example of image splicing (A) and (B) The genuine images (C) The resulting image. (Alamro & Yusoff, 2017)

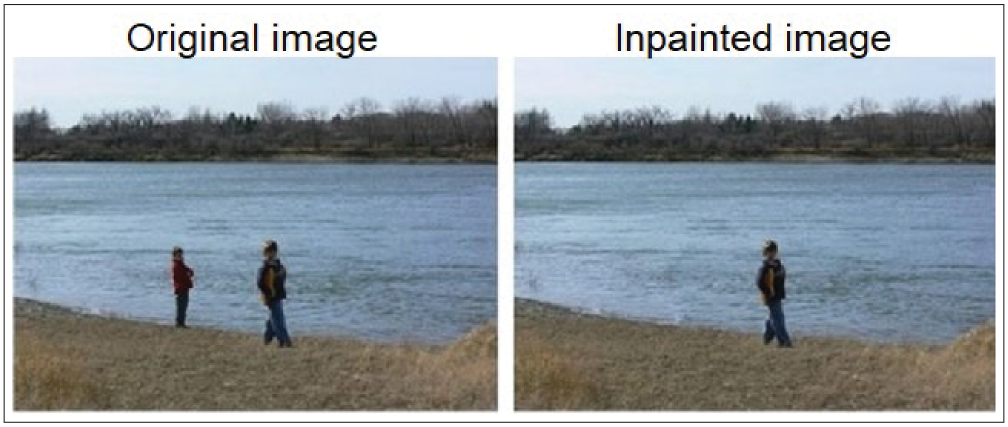


Figure 1.3 Image inpainting samples in case of object removal (Kumar & Meenpal, 2020)

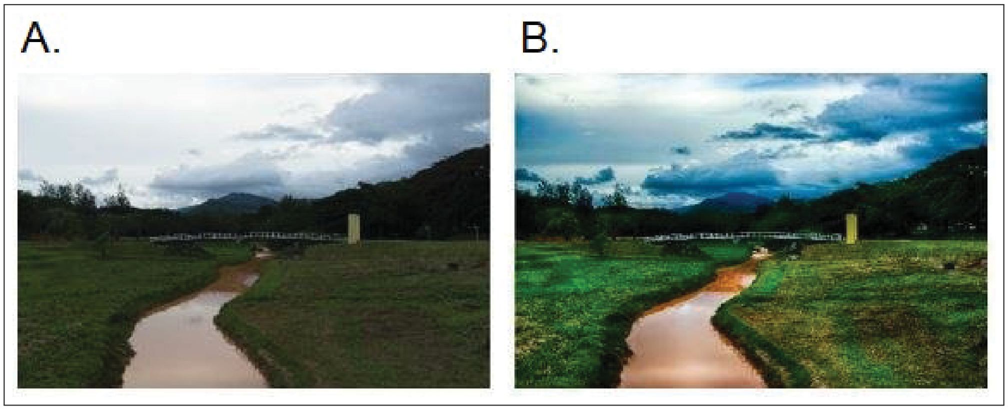


Figure 1.4 Samples of image retouching. A) The genuine image B) The forged image. (Alamro & Yusoff, 2017)

1.1.2 Feature extraction

Because of the size of the candidate images in forgery detection, it is computationally intractable to directly use their RGB representation in classifiers or segmentation models to train dense neural networks. Convolutional Neural Networks possess a great feature extraction power, which means that they are able to compress relevant image information and turn it into a feature representation. The reduced input can then be processed with fully connected layers, for instance to perform classification or segmentation.

Given an image input of height H and width W denoted $x \in \mathbb{R}^{H \times W \times 3}$, we will use $f_\theta(x) \in \mathbb{R}^{H' \times W' \times C}$ to represent C feature maps extracted by a convolutional network with parameters θ.

1.1.3 Image forgery classification

We call a set of examples images and labels $\mathcal{M} = \{\mathbf{x_i}, y_i\}$ with $\mathbf{x_i}$ being an image, and $y_i \in \mathcal{Y} = \{0, 1\}$. $\mathbf{x_i}$ is pristine if $y_i = 0$ and transformed by $t_i \in \mathcal{T}$ if $y_i = 1$ ($\mathcal{T}$ being a finite set of image manipulations).

The aim of image manipulation detection is to retrieve the label $y \in \mathcal{Y}$ associated to an image. Image manipulation classification is more fine-grained as the goal is to retrieve t_i with a classifier $c(\mathbf{x})$.

This classifier $c(\mathbf{x})$ allows us to compute the probabilities p_j of m being a pristine image o transformed by $t_j, \forall t_j \in \mathcal{T}$. In other words, we find the set $\mathcal{P} : \{p_j = P(m = t_j(o)), \forall t_j \in \mathcal{T}\}$.

1.1.4 Image manipulation segmentation

Image manipulation segmentation is closely related to image manipulation detection. While the former aims at describing each pixel of the candidate image (i.e., identifying those pixels with potential modifications), the latter aims at describing the entire image as a whole (i.e., whether the image has been manipulated). In this class of problems, we usually have a set of examples $\mathcal{S}_1 = \{\mathbf{x_i}, M_i\}$ with $\mathbf{x_i}$ an image of dimensions $H \times W$, and $M_i \in \{0, 1\}^{H \times W}$ a segmentation map. Each element of M_i corresponds to the probability that its associated pixel is manipulated.

The goal of image manipulation segmentation is to infer a manipulation map M given a previously unseen image X. Instead of having $M_i \in \{0,1\}^{H\times W}$, it is also possible to associate each pixel to C different manipulation classes and have $M_i \in \{0, C-1\}^{H\times W}$.

1.1.5 Domain-Incremental learning

Given $S = \{D_1 ... D_j ... D_n\}$ and denoting as A_i the accuracy of the model on the dataset D_i, the aim of Domain-Incremental Learning is to maximize the Cumulative Accuracy:

$$CA = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{i}\sum_{j=1}^{i}A_j \tag{1.1}$$

At each timestep j, the model can only train using D_j and not to the previous ones (e.g. to satisfy privacy, performance or memory constraints) but it is still evaluated on those past domains. The inference is domain-agnostic: Given one data sample, the model doesn't have access to the domain index to make its prediction (at test time). In other words, if we want to classify an unknown sample $x \in D_i$, we do not know D_i.

We want to make sure that the model generalizes to unseen domains while performing well on known ones, because as explained the model could encounter test data coming fro unseen domains. We therefore advocate for the need to maximize the following instead of (1.1):

$$\frac{1}{n}\sum_{i=1}^{n}\frac{1}{i}\sum_{j=1}^{n}A_j \tag{1.2}$$

1.2 Bibliographic study

The aim of this section is to introduce relevant previous attempts at forgery detection and domain-incremental learning.

1.2.1 Forgery detection

1.2.1.1 Classification

Feature maps fusion: In order to perform copy-move detection in images using a CNN as a feature extractor, Li *et al.* (2021) chose to consolidate the original image input by also using feature maps representing spatial intensity values associated with different frequency bands. As they observed empirically that forgery clues are present in some frequency bands, using those features directly along with the original images allows for a faster convergence of the network and lowers the amount of training data needed to create a good representation. Since the original input is conserved and these new feature maps are fused with the ones extracted by a CNN later, no knowledge is lost as it can be the case in traditional machine learning flows where only potentially sub-optimal hand-crafted features are used.

The new feature maps are fused with the extracted ones before the middle flow of a Xception network (Chollet, 2017) using a 1×1 convolutional layer. Abstract features are on the first hand extracted separately from the RGB image and the frequency information, then fused together and convolved again.

We note $\mathbf{x}_i$ as the original image, $g(\mathbf{x}_i)$ as the extracted frequency clues from $\mathbf{x}_i$, f_{θ_1} as Xception's entry flow transform, f_{θ_2} as Xception's middle and exit flows transform, and $\odot$ as a 1×1 convolution. Final features maps F in Li *et al.* (2021) are therefore:

$$F = f_{\theta_2}(f_{\theta_1}(\mathbf{x}_i) \odot g(\mathbf{x}_i)) \in \mathbb{R}^{H_1 \times W_1 \times C_1} \tag{1.3}$$

Feature fusion can also be performed at the beginning of the network when the handcrafted features are concatenated along with the image, forming a new input. In that case, we let the network find the most adequate kernel parameters to fuse the features as in (Liu *et al.*, 2021). If we note $[\cdot, \cdot]$ as feature maps concatenation and using Li *et al.* (2021)'s parameters, we would have the following fused feature maps:

$$f_{\theta_2}(f_{\theta_1}([\mathbf{x}_i, g(\mathbf{x}_i)])) \in \mathbb{R}^{H_2 \times W_2 \times C_2} \tag{1.4}$$

The fusion of the feature maps could be improved with an architecture with skip connections between layers similarly to HyperDenseNet's (Dolz *et al.*, 2018) which would give the network more flexibility on how to perform the fusion.

Capturing more general features: Without adequate consideration, forgery detection networks typically overfit to the training data Dong *et al.* (2022). It is a problem as one want to detect forgery clues in the images and not learn the semantic of what is a normal image. One therefore has to ensure that the extracted features summarize the input efficiently without losing information necessary to perform the downstream task. Feature extractors implemented as a convolutional neural network tend to generalize poorly. Indeed, it seems that, in a lot of cases, the extracted features only help classifying the examples present in the training set but are not always efficient at creating a useful representation for examples of unseen classes.

The feature extractor is sensitive to the choice of classes and in particular to its granularity. It has been empirically shown that initially training the feature extractor to create a representation to discriminate finer classes can help improve generalization (Wu *et al.*, 2019a). Indeed, by creating a hierarchy of coarse to fine classes and training a first model using the finer classes, classification performance using the most coarse level (which are therefore superclasses) seem to be improved and performance on unseen manipulation types as well. In Wu *et al.* (2019a), the hierarchy goes from the most used manipulation types such as copy-move, Gaussian blur or box blurring (being the most coarse levels) to refined manipulation types such as Gaussian blur with a kernel size of 3 (finest level). It makes the number of different classes go from 10 to 385, adding model complexity in the classifier part. This makes it harder to select an adequate baseline for the classifier, as algorithms performing the best on a 10-class problem are not necessarily the best as well for the 385-class problem. While it is tractable to compare the performances of algorithms for 10-class problems, it becomes more computationally expensive to do it for 385 classes.

Once the model with 385 classes is trained, one can choose to keep the parameters from the feature extractor frozen (which is now supposed to be creating more robust features) and to retrain a 10-way classifier on top of the frozen feature extractor.

Single center loss: Image forgery detection has traditionally been formulated as a classification problem where the aim is to identify existing manipulation types. As researchers only have access to a limited set of examples and are not aware of all the image manipulation types, one of the key challenges in classification is to improve the generalization of the network without overly deteriorating its discriminative power. Indeed, one must ensure that its network will still deem as forged the images manipulated using so far unknown techniques. Deep learning models supervised with Cross Entropy losses focus on finding a good decision boundary to discriminate known manipulation types and pristine examples in the feature space. While this could be sufficient if the training data and the inference data were well aligned with little noise, it is not enough in most real use cases. The margin between the classes will typically be small. As the margin between classes is small, it is easy to make a confident error.

Only the feature distribution associated with pristine images should not change in a dataset containing a new manipulation type. In Li *et al.* (2021), the authors focus on constraining this feature space instead of the manipulated ones. In order to create a margin between points representing pristine images and corrupt ones, a new loss is created with the aim of ensuring compactness of the pristine image's features in the feature space, and manipulated images feature points are pushed away, by a controllable margin δ.

This loss can be formulated as follow:

$$L \; = \; D_{nat} \; + \; \max(D_{nat} - D_{mod} + \delta, \; 0) \tag{1.5}$$

with D_{nat} representing the average Euclidean distance between representations of pristine images and their feature space class center, and D_{mod} representing the average Euclidean distance between representations of corrupt images and the class center of pristine images.

Introducing $\mathbf{m}$ as the feature space class center of pristine images, we can then formalize this as:

$$\begin{aligned} \mathbf{m} &= \frac{1}{\Omega_{nat}} \sum_{\mathbf{x}_i \in \Omega_{nat}} f_\theta(\mathbf{x}_i) && (1.6) \\ D_{nat} &= \frac{1}{\Omega_{nat}} \sum_{\mathbf{x}_i \in \Omega_{nat}} \| f_\theta(\mathbf{x}_i) - \mathbf{m} \|_2 && (1.7) \\ D_{mod} &= \frac{1}{\Omega_{mod}} \sum_{\mathbf{x}_i \in \Omega_{mod}} \| f_\theta(\mathbf{x}_i) - \mathbf{m} \|_2 && (1.8) \end{aligned}$$

Therefore, we will optimally have:

$$D_{nat} = 0 \text{ and } D_{nat} - D_{mod} + \delta = 0 \Rightarrow D_{mod} = \delta \tag{1.9}$$

When presented with an image transformed with an unseen forgery type, the classifier is supposed to be less confident about its prediction if the manipulation does indeed modify the features extracted previously.

1.2.1.2 Segmentation

Weak supervision: It can sometimes be useful to convert a segmentation problem into a classification one. Indeed, when one is presented with a classification algorithm and examples are only labeled at the pixel-level, one has to supervise the model without image level information. It is possible to supervise the model in such a fashion using some hypotheses. A cost function with only image-level labels is introduced in (Dang *et al.*, 2020) for this specific scenario:

$$\mathcal{L}_{img} = |\varsigma(\mathbf{M_i}) - 0|, \text{ if } y_i = 0 \tag{1.10}$$

$$\mathcal{L}_{img} = |\max(\varsigma(\mathbf{M_i})) - 0.75|, \text{ if } y_i = 1 \tag{1.11}$$

where $\varsigma(x)$ denotes the sigmoid function.

This loss can then be used to perform weakly supervised segmentation learning where some example are fully labeled with their ground truth manipulation mask and some only with image-level labeling. We use a classification and a segmentation loss for the former and only a classification loss for the latter.

Encoder-decoder architectures: Encoder-decoder architectures have been widely used in segmentation models and unsupervised learning Minaee *et al.* (2021); Ranzato, Huang, Boureau & LeCun (2007). The encoder part is usually similar to the feature extractor of classifiers, consisting of layers gradually narrowing down feature maps and expending their dimensionality. Although classifiers are usually made of dense layers, decoders usually consist of fully convolutional layers and upsampling operators.

The method of Wu *et al.* (2018) specializes in detecting copy-move forgeries, for which corrupt pixels are coming from a pristine zone of the same image and are duplicated in another zone.

In this case, it is useful to identify the origin and the target regions. Two encoder-decoder branches are used to meet that goal. The first one detects image manipulation and the second one uses a self-correlation module to detect similarity between parts of the image. Both branches are first optimized using a binary cross-entropy loss. Combining these two approaches, it is possible to infer the origin and the destination of the copy-move. Authors chose to use this direct supervision but also to optimize a fusion classifier module for these two branches using categorical cross-entropy loss. The whole network is then fine-tuned with the three losses.

A decoder architecture has multiple steps of upsampling. It is proved formally and empirically in (Liu *et al.*, 2021) that these steps usually leave traces in the frequency domain and mostly in the phase spectrum. It is therefore informative to consolidate the input features of the model with a representation of this phase spectrum.

Self-attention: Attention mechanisms in deep-learning pipelines generally allow for increased performances and interpretability. We can draw a parallel with human vision which focuses on specific relevant parts of the image and therefore can forget irrelevant parts (which can be considered as noise).

Attention mechanisms are usually comprised of two key parts: learning an attention map, i.e., a heat map from where the important features are supposed to be located, and using that attention map to forget useless information.

The attention map can be generated directly from the example images or using prior knowledge. In (Dang *et al.*, 2020), these two approaches are leveraged jointly. Indeed, the attention map space is approximated into a 10-dimensional one using the 10 most important components extracted by Principal Component Analysis on the ground truth masks. It is then far easier to estimate 10 parameters to generate one attention map than to recreate the attention map completely. This can be useful when we do not have a lot of training examples, or some of the examples lack pixel-level labeling (using 1.10 or 1.11).

Multiple orders of attention can be beneficial for the network. For instance, a first order attention map is used in (Islam, Long, Basharat & Hoogs, 2020) to reduce the feature space (but not necessarily its dimension) and to modulate the features according to their relative importance. A second order attention mechanism is used to consolidate pixel-level features with features of other relevant pixels that can be located far away. In a classical convolution setting it would have been harder to process these features together because of how far they can be and how narrow the receptive field associated with pixels usually is.

Attention is exerted after feature extraction where we have features $f_\theta(\mathbf{x}) \in \mathbb{R}^{H_1 \times W_1 \times C_1}$. In order to reduce memory constraints, attention is performed on patches instead of pixels.
The feature tensor then becomes $f_{2,\theta}(\mathbf{x}) \in \mathbb{R}^{H_2 \times W_2 \times C_2}$ with $H_2 = H_1/R$ and $W_2 = W_1/R$ and R the patch size. $C_2 = C_1$ in case of aggregation or $C_2 = C_1 \times R^2$ if we concatenate the features of all patches in channel-wise manner. Since we want to calculate similarity between features, this feature tensor is reshaped into $f_{3,\theta}(\mathbf{x}) \in \mathbb{R}^{H_2 W_2 \times C_2}$.

An affinity matrix

$$A = f_{3,\theta}(\mathbf{x}) \cdot f_{3,\theta}(\mathbf{x})^T \tag{1.12}$$

is then computed. As we are not interested by self-correlation between same patches, a Gaussian filter is applied on A to reduce the value of elements in or near the diagonal, producing $A_{corrected}$. Softmax (σ) is then applied for normalization:

$$A_2 = \sigma(A_{corrected}) \tag{1.13}$$

This is the second-order attention matrix that can be used to consolidate pixel-level features with features of other relevant pixels that can be located further away:

$$f_{4,\theta}(\mathbf{x}) = A_{normalized} \cdot f_{3,\theta}(\mathbf{x}) \tag{1.14}$$

First order attention map A_1 can be computed by locating patches highly correlated with other patches using A_2. These patches will have some high values row-wise (or column-wise) in A_2. It is then possible to reduce A_2 by creating a representation with the K largest values of each row (or column) and to reshape the computed matrix to the size of the original patch-wise feature representation, giving:

$$A^K \in \mathbb{R}^{H_2 \times W_2 \times K} \tag{1.15}$$

A_2 can be learned from A^K using 3 convolution layers followed respectively by BN+ReLU, BN+ReLU and Sigmoid. It is then possible to highlight the patches that are highly correlated with other patches:

$$f_{5,\theta}(\mathbf{x}) = A_2 \cdot f_{3,\theta}(\mathbf{x}) \tag{1.16}$$

Attention can also be specifically channel-wise or on the spatial dimension as it is the case in (Woo, Park, Lee & Kweon, 2018). Either way, these modules are integrated within the network and optimized jointly with it.

Channel-wise attention: Channel-wise attention is aimed at reinforcing the representation of some feature maps using a channel-wise attention map. This attention map is computed as follows:

$$A^c = \sigma(MLP(\text{ChannelAveragePool}(f_{3,\theta}(\mathbf{x}))) + MLP(\text{ChannelMaxPool}(f_{3,\theta}(\mathbf{x})))) \tag{1.17}$$

where ChannelAveragePool denotes average pooling on each feature map, resulting in one value per channel, and ChannelMaxPool maximum pooling on each feature map, resulting in one value per channel. MLP denotes a one hidden layer multi-layer-perceptron. This attention map can used on the features in the same way as a 1×1 convolution operation with equal input and output size and A^c as kernel.

Spatial attention: Spatial attention is aimed at reinforcing the representation of elements at some position in the feature maps using a spatial attention map:

$$A^s = \sigma(CONV_{7x7}(\text{SpatialAveragePool}(f_{3,\theta}(\mathbf{x}))^\frown\text{SpatialMaxPool}(f_{3,\theta}(\mathbf{x})))) \tag{1.18}$$

where SpatialAveragePool denotes average pooling across the feature maps, and SpatialMaxPool maximum pooling across the feature maps, compressing all the feature maps in one. $CONV_{7\times7}$ denotes a convolution with a 7×7 kernel. A^s is applied over the channel dimension using element-wise multiplication. If we have $f_{3,\theta}(\mathbf{x}) \in \mathbb{R}^{h_2 w_2 \times c_2}$ then A^s is concatenated c_2 times across the channel dimension, forming A^{s,c_2} and spatial attention is applied as below:

$$f_{3,att,\theta}(\mathbf{x}) = A^{s,c_2} \odot f_{3,\theta}(\mathbf{x}) \tag{1.19}$$

Anomaly detection: In most of the real use cases in forgery detection, it is hard to collect a set of paired (image,label) examples and we only have access to a set of pristine images $\mathcal{U} = \{\mathbf{x}_i)\}$. It is however possible to describe each of these images with features $f_\theta(\mathbf{x}_i)$. One can then create a model d that will compute the probability of an image x being corrupt based on the similarity of its features compared to the features of the images in the pristine images set (similarity of $f_\theta(\mathbf{x})$) with feature of the other images of $\mathcal{U}$ $(f_\theta(\mathbf{x}_i))$. This method aims at finding anomalous features and thus at detecting that the image was manipulated. However, it does not allow us to infer the image forgery type.

The choices of f and d depend on the different approaches of anomaly detection and require hypotheses on the data. Bammey, Gioi & Morel (2020) formulates the problem as an anomaly detection one. Their work is based on the assumption that most of the pristine images were taken by a camera model using an image sensor overlaid with the same color filter array. The CFA consists of a 2 pixels by 2 pixels square pattern repeating over the whole image sensor. It is composed of sensors sensible to red, green or blue light and allows for the interpolation of a RGB image from sparse RGB information. As reconstruction from the image from one channel leaves traces, it is possible to predict which color of the color filter array was over the cell associated with a pixel of the image. This phenomenon makes it possible to formulate a self-training task: given blocks of mostly non-manipulated images (more robust prediction than at pixel-level), predict its position modulo the pattern shape (2,2). It is then possible to infer the position of most blocks in non-manipulated images but harder for manipulated images since the restoration process has been altered. Manipulated blocks can therefore be detected using this fact.

Self-training: Wu *et al.* (2019a) uses a self-training task to learn a rich feature representation of the images and then performs patch-wise features anomaly detection. It does so with different reference windows sizes in a far-to-near fashion. In this case, we at first only have $\mathcal{U} = \{\mathbf{x}_i\}$, and labels are created by applying 385 different transforms ($t \in \mathcal{T}$) on the original dataset to create forged images, as explained in 1.2.1.1.

The features extracted by f_{θ_2} are compared at a local level to features in reference windows of different sizes (of lengths 7, 15, 31 and of the image size) using a Z-score. These Z-score are then compared in a sequential manner using a ConvLSTM2D layer (d_u) (Shi *et al.*, 2015) and allows for inference of anomalous patches.

1.2.2 Domain-Incremental Learning

As explained in 1.1.5, the aim of domain-incremental learning is to optimize the cumulative performance of one model through its lifetime and not just perform well on one domain at a time. We will now give some background on the existing literature on DIL.

1.2.2.1 Regularization methods

Regularization-based methods aim at identifying the important weights of a deep neural network and penalizing a change in those weights when learning on a new domain. Elastic Weight Consolidation (EWC) (Aich, 2021) is a regularization technique which uses a quadratic penalty based on the Fisher information metric's diagonal to penalize changing weights important to previous tasks. This method is well-principled but also makes strong assumptions, i.e. it is based on a second order Taylor expansion (therefore local) and uses a rough approximation of the Fisher information metric which could be improved. Synaptic Intelligence (SI) (Zenke *et al.*, 2017) computes an importance measure for each parameter of the network to capture the contribution of these parameters to the loss function. In particular, SI uses the integral of the product of the parameter's gradient and the change in the parameter value during the learning process. SI circumvents the computational burden of calculating a good approximation to the Fisher information matrix required in other methods like EWC.

1.2.2.2 Prompt learning

Driven by the advances in Natural Learning Processing, prompt learning has emerged as an appealing learning strategy to adapt large scale pre-trained models to downstream tasks Bahng, Jahanian, Sankaranarayanan & Isola (2022). While initial attempts to adapt language-vision models have centered on carefully designing handcrafted prompts (Brown *et al.*, 2020), recent works focus on optimizing a task-specific continuous vector, which is optimized via gradients during fine-tuning (Zhou, Yang, Loy & Liu, 2022b,a; Lu, Liu, Zhang, Liu & Tian, 2022; Ju, Han, Zheng, Zhang & Xie, 2022). An underlying limitation of these approaches arises from the inherent disparity between language and vision modalities, and thus fine-tuning only text prompts for visual recognition tasks may yield suboptimal performance. Motivated by this, visual prompt tuning (VPT) (Jia *et al.*, 2022) was proposed as a powerful alternative to text prompting. In this approach, authors propose to optimize task-specific learnable prompts in either the input or visual embedding space. Following the satisfactory results achieved by VPT, fine-tuning visual prompts has gained popularity recently, particularly for adapting pre-trained models to novel unseen categories (Sohn *et al.*, 2023; Chen, Yao, Chen, Zhang & Liu, 2023; Xing *et al.*, 2022; Xing *et al.*, 2022).

1.2.2.3 Prompt tuning in domain incremental learning

This paradigm protects against catastrophic forgetting by optimizing a small set of learnable prompts. This contrasts with classical approaches which modify all the network parameters (or a subset), or store *exemplars* in a buffer. Despite the success observed in other tasks, the literature on prompt tuning for domain incremental learning remains under-explored, with just a handful works addressing this problem (Wang *et al.*, 2022b; Wang *et al.*, 2022d; Douillard, Ramé, Couairon & Cord, 2022). For example, S-Prompts (Wang *et al.*, 2022b) learns in isolation a set of prompts per domain, and dynamically selects which set to use at test-time using a fixed key/value dictionary where the keys are computed with K-Means and the values represent the sets of prompts. L2P (Wang *et al.*, 2022d) uses an incrementally learnable key/value mechanism to select which prompts to prepend to the input image tokens at test-time, hence breaking the isolation between domains, which contrasts with our work, as it learns domain prompts independently. A main difference with these and conventional DIL approaches is that the proposed approach explicitly tackles the generalizability performance in domain incremental learning, while maintaining at par accuracy in known domains, which remains under-explored.

1.2.2.4 Domain generalization (DG)

The objective of domain generalization (DG) is to alleviate issues related to domain-shift in the absence of labeled target data. Existing literature on DG strongly relies on supervised knowledge from source domain data, regardless of whether it originates from a single domain (Wang, Luo, Qiu, Huang & Baktashmotlagh, 2021) or multiple domains (Yao *et al.*, 2022; Zhang, Li, Li, Jia & Zhang, 2022; Zhang *et al.*, 2023; Chen, Li, Han, Liu & Yu, 2022), which may not be realistic in continually changing scenarios, as knowledge comes in a sequential manner. Our scenario is closest to source-free domain generalization as we cannot store source data. Additionally, in scenarios involving distributional shifts, DG approaches primarily focus on performing well solely on the target domain, increasing the potential risk of catastrophic forgetting on previously learned domains (Liu *et al.*, 2023).

1.2.2.5 Dynamic classifier or ensemble selection

Dynamic classifier and ensemble selection aims at using the best learned classifiers (given a pool of classifiers) for a test sample. A pool or only one classifier can be selected and dynamically used to infer the predictions. The competence of each classifier can be computed using clustering, K-NN or potential functions derived distances Cruz, Sabourin & Cavalcanti (2018). In Dynamic Classifier Selection methods, the best learned classifier for a test sample only is selected. In Dynamic Weighting methods, the logits or probabilities returned by different classifiers trained on close datasets are linearly combined Štefka & Holeňa (2015); Tsymbal, Pechenizkiy, Cunningham & Puuronen (2008); Jiménez (1998); Cevikalp & Polikar (2008).

CHAPTER 2

PROPOSAL: MOP-CLIP FOR GENERAL DOMAIN-INCREMENTAL LEARNING

* The results presented in this chapter have been submitted to the *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV) 2024.*

2.1 Introduction

In this approach, domain-specific knowledge is preserved in the form of textual and visual prompts, alleviating the need of storing exemplars per domain. While some methods advocate for the joint learning of prompts across tasks (Douillard *et al.*, 2022; Wang *et al.*, 2022d), the recent work in (Wang *et al.*, 2022b) instead favors the learning of the prompts independently, suggesting that this leads to the best performance per domain. This learning paradigm, referred to as S-Prompting (Wang *et al.*, 2022b), circumvents the issue of using expensive buffers by optimizing per-domain prompts, which are leveraged at testing time. In particular, centroids for each domain are obtained during training by applying K-Means on the training image features, which are generated with the fixed pre-trained transformer without using any prompts. Then, during inference, the standard KNN algorithm is used to identify the nearest centroid to the test image, whose associated domain prompt is added to the image tokens for classification. Despite the empirical performance gains observed by prompt learning approaches (Douillard *et al.*, 2022; Wang *et al.*, 2022d,b), a current limitation hampering their generalization is that they perform satisfactorily in *known* domains, but typically fail on *unseen* domains.

We therefore introduce a novel *exemplar-free* DIL solution, based on prompt learning, generalizing the recent S-liPrompts approach (Wang *et al.*, 2022b) to perform well on unseen domains as well as known domains.

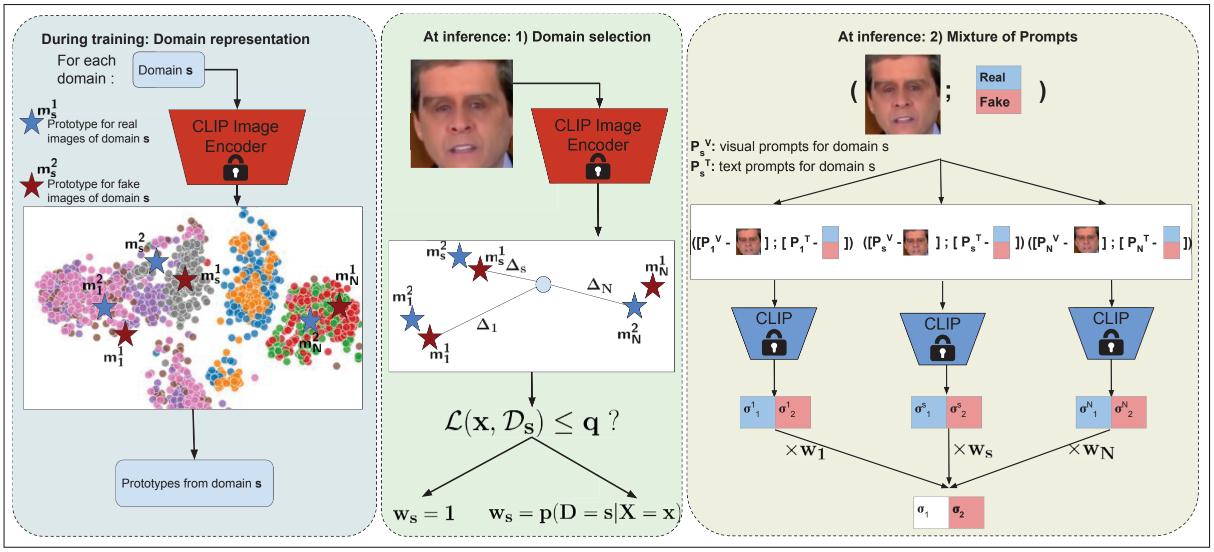

During training: Domain representation
For each domain :
Domain s
m_s^1
Prototype for real images of domain s
m_s^2
Prototype for fake images of domain s
CLIP Image Encoder
m_s^2
m_s^1
m_1^2
m_1^1
m_N^1
m_N^2
Prototypes from domain s
At inference: 1) Domain selection
CLIP Image Encoder
m_s^2
m_s^1
Δ_s
m_N^1
Δ_N
m_1^2
Δ_1
m_N^2
m_1^1
$\mathcal{L}(\mathbf{x}, \mathcal{D}_\mathbf{s}) \leq \mathbf{q}$?
$\mathbf{w_s = 1}$
$\mathbf{w_s = p(D = s|X = x)}$
At inference: 2) Mixture of Prompts
(; Real Fake)
P_s^V: visual prompts for domain s
P_s^T: text prompts for domain s
$([P_1^V -] ; [P_1^T -])$
$([P_s^V -] ; [P_s^T -])$
$([P_N^V -] ; [P_N^T -])$
CLIP
CLIP
CLIP
σ_1^1 σ_2^1
σ_1^s σ_2^s
σ_1^N σ_2^N
$\times w_1$
$\times w_s$
$\times w_N$
σ_1 σ_2

Figure 2.1 **Overview of MoP-CLIP.** The training phase (*left*): class-wise prototypes are identified from in-distribution domains. Inference (*middle* and *right*): domain selection and ensembling (Mixture of Prompts), respectively, for in-distribution and out-of-distribution samples. For simplicity, we depict the pipeline for 2 classes (Real *vs* Fake). However, the procedure for multiple classes (e.g., DomainNet or CoRE50) is exactly the same.

2.2 Method

An overview of MoP-CLIP is illustrated in Fig. 2.1, which contains two phases: *i)* learning of in-distribution domain-specific visual and text prompts (sec. 2.2.1) and *ii)* selection of optimal prompts for a given test sample (sec. 2.2.2). The proposed approach is summarized in Algorithm 1.

Problem definition

Let us denote as $\mathcal{S} = \{\mathcal{D}_s\}_{s=1}^{N}$ the sequence of datasets presented to the model in our incremental learning scenario, with N being the final number of domains. Each dataset is defined as $\mathcal{D}_s = \{\mathbf{x}_i^s, \mathbf{y}_i^s\}_{i=1}^{|\mathcal{D}_s|}$, where $\mathbf{x}_i \in \mathbb{R}^{W \times H \times C}$ represents an image of size $W \times H$ and C channels, and $\mathbf{y}_i \in \{0, 1\}^K$ is its corresponding one-hot label for K target classes. In this setting, we have access to only one domain $\mathcal{D}_s$ at a time and storing samples from previous seen domains, commonly

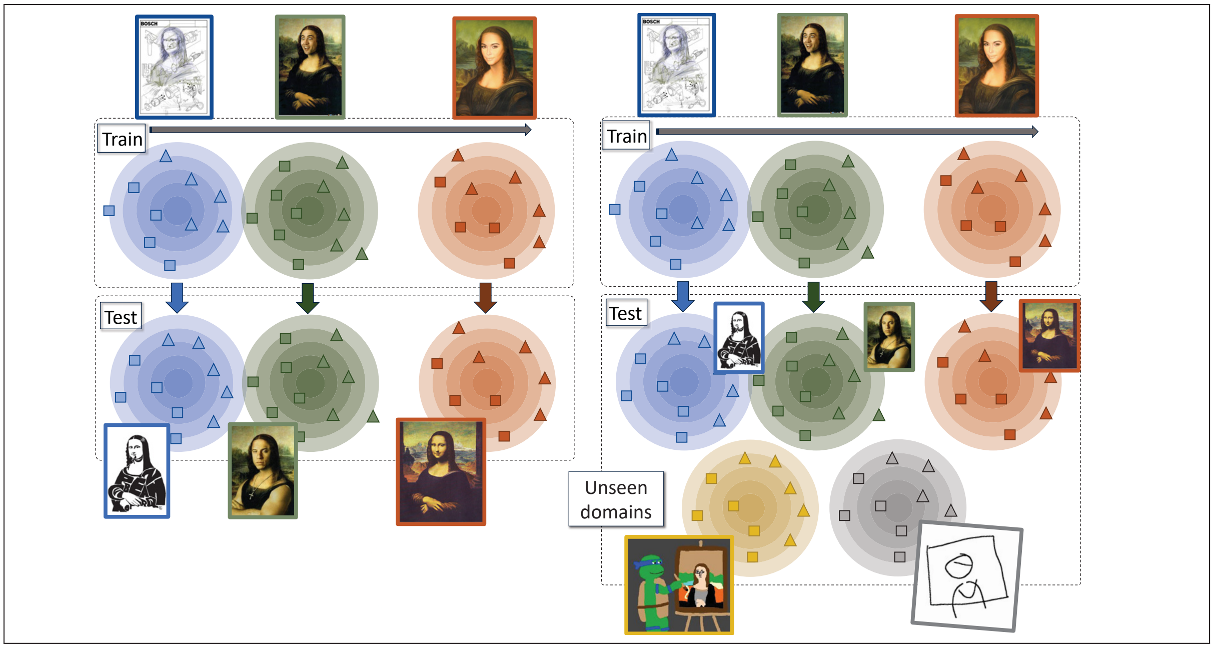


Figure 2.2 **Proposed generalization scenario for domain incremental learning** Standard problem (*left*): Only in-domain examples are encountered at test time. Addressed problem (*right*): Both in-domain and out-of-domain examples are presented at test time.

referred to as *exemplars*, is not allowed. Each time a new domain $\mathcal{D}_s$ becomes accessible, DIL aims to improve the model's performance on $\mathcal{D}_s$, while avoiding the loss of knowledge for past domains, $\mathcal{D}_{s-1}, \mathcal{D}_{s-2}, ...\mathcal{D}_1$. In the proposed setting, and in contrast to most existing literature on DIL, we assume that the model should also generalize well on unseen domains, i.e., $\mathcal{D}_{s+1}, \mathcal{D}_{s+2}, ..., \mathcal{D}_{|\mathcal{D}_s|}$ (Fig. 2.2). In other words, our learning scenario leverages *backward transfer* to avoid catastrophic forgetting on seen domains, while optimizing *forward transfer* to facilitate knowledge transfer to new tasks/domains. Our motivation behind this bi-directional performance assessment relies on the realistic assumption that a distributional drift between training and testing data always exists.

2.2.1 Prompts Learning

Following the setting of Wang *et al.* (2022b), we define f_θ as the pre-trained vision transformer that generates a visual embedding $\mathbf{z}^v = f_\theta(\mathbf{x}_{\text{tok}}) \in \mathbb{R}^L$, where $\mathbf{x}_{\text{tok}} \in \mathbb{R}^{WH/R^2 \times M^v}$ corresponds to the image tokens (or patches), WH/R^2 is the number of tokens, R is the width/height of the (square) patch and M^v is the dimension of the image tokens embedding. We also define f_ϕ, a pre-trained text transformer that generates text embeddings of dimension M^t from class names tokens $\mathbf{c}_k$ for $k \in \{1, ..., K\}$. For each new domain $\mathcal{D}_s$ in the sequence $\mathcal{S}$, we can adapt the model by learning a visual prompt $\mathbf{p}_s^v \in \mathbb{R}^{L^v \times M^v}$ and a text prompt $\mathbf{p}_s^t \in \mathbb{R}^{L^t \times M^t}$, following Wang *et al.* (2022b). In particular, these prompts are a set of continuous learnable parameters, where L^v, L^t are the visual and text prompt length. Thus, for the set of domains $\mathcal{S}$, we have a set of domain-specific visual and text prompts, denoted as $\mathcal{P}^v = \{\mathbf{p}_1^v, ..., \mathbf{p}_N^v\}$ and $\mathcal{P}^t = \{\mathbf{p}_1^t, ..., \mathbf{p}_N^t\}$. Now, with the domain-specific prompts, we can modify the embeddings that will be provided to the visual and text encoders, f_θ and f_ϕ. Concretely, for an image of domain s and class k, the input of the visual transformer is defined as $\tilde{\mathbf{x}}^v = [\mathbf{x}_{\text{tok}}, \mathbf{p}_s^v, \mathbf{x}_{\text{cls}}]$ with $\mathbf{x}_{\text{cls}}$ the classification token of the ViT. Similarly, the input of the text transformer is defined as $\tilde{\mathbf{c}}_k^t = [\mathbf{p}_s^t, \mathbf{c}_k]$. We then denote as $\tilde{\mathbf{z}}^v = f_\theta(\tilde{\mathbf{x}}^v)$ and $\tilde{\mathbf{z}}_k^t = f_\phi(\tilde{\mathbf{c}}_k^t)$ the embeddings of these inputs. The posterior probability of a given image $\mathbf{x}_i$ from $\mathcal{D}_s$ belonging to class k can be therefore defined as:

$$p(\mathbf{y}_k|\mathbf{x}, s) = \frac{e^{\cos(\tilde{\mathbf{z}}^v, \tilde{\mathbf{z}}_k^t)}}{\sum_{j=1}^{K} e^{\cos(\tilde{\mathbf{z}}^v, \tilde{\mathbf{z}}_j^t)}}, \tag{2.1}$$

where $\cos(\mathbf{a}, \mathbf{b}) = \frac{\mathbf{a} \cdot \mathbf{b}}{||\mathbf{a}||\,||\mathbf{b}||}$ is the cosine similarity between vectors $\mathbf{a}$ and $\mathbf{b}$.

2.2.2 Inference

At test time, the domain of the images to classify remains unknown. In S-liPrompts (Wang *et al.*, 2022b), the domain s^* closest to a given test sample is selected by finding the minimum distance between the visual embeddings and prototypes computed with K-Means over the domains $\mathcal{S}$.

This strategy is generally effective in finding the closest domain when $\mathbf{x} \in \mathcal{D}_s$ and $\mathcal{D}_s$ has been already presented to the model. In this setting, $p(\mathbf{y}_k|\mathbf{x}, s)$ yields satisfying predictions, as the domain of the sample $\mathbf{x}$ can be easily inferred and the scenario becomes a classification task under in-distribution data. Nevertheless, when the model has not been exposed to $\mathcal{D}_s$ during training or adaptation, the selection of an existing closest domain (other than $\mathcal{D}_s$) might not match with the real distribution of the new domain. In this case, S-liPrompts will use a model for prediction that may be sub-optimal. To overcome this issue, we propose to enhance the domain selection mechanism in two separate ways: *i)* dynamically allowing the model to select n close domains and *ii)* leveraging per-domain predictions in an ensembling scheme for samples of unseen domains.

To select the right prompt, we propose a strategy based on a set of class-specific prototypes for each domain, $\mathcal{E}_s = \{\boldsymbol{m}_s^k\}_{k=1}^K$, instead of prototypes obtained with K-Means as in Wang *et al.* (2022b). Let $\mathcal{D}_s^k \subset \mathcal{D}_s$ be the samples of domain $\mathcal{D}_s$ belonging to the class k, we compute the the prototype of class k for domain $\mathcal{D}_s$ by averaging the visual embeddings of examples in $\mathcal{D}_s^k$:

$$\boldsymbol{m}_s^k = \frac{1}{|\mathcal{D}_s^k|} \sum_{\{z^v \,|\, \mathbf{x} \in \mathcal{D}_s^k\}} \boldsymbol{z}^v \tag{2.2}$$

Next, we present how these prototypes are used to select the domain and how they are leveraged in our approach.

2.2.2.1 Domain Selection.

Given the class-specific prototypes, we select the domain s^* of a test example $\mathbf{x}$ as the one with the nearest prototype for any class:

$$s^* = \underset{1 \leq s \leq N}{\operatorname{argmin}} \, \Delta_s(\mathbf{x}) \tag{2.3}$$

with

$$\Delta_s(\mathbf{x}) = \min_{\boldsymbol{m}_s^k \in \mathcal{E}_s} \|\mathbf{z}^v - \boldsymbol{m}_s^k\|_2. \tag{2.4}$$

As mentioned before, test examples may also come from an out-of-distribution (OOD) domain (i.e., not part of any domains encountered at training time). To determine if a given sample $\mathbf{x}$ is from a previously-seen domain or is OOD, we compare its distance to the closest prototype of the selected domain, $\Delta_{s^*}(\mathbf{x})$, with the distances of training samples from that domain. Let $\Psi_s^k = \left\{\|\mathbf{z}^v - \boldsymbol{m}_s^k\|_2 \,|\, \mathbf{x} \in \mathcal{D}_s^k\right\}$ be the set of distances for domain $\mathcal{D}_s$ and class k. During training, the distribution of distances for each domain $\mathcal{D}_s$ and class k is estimated from Ψ_s^k with a Gaussian of mean μ_s^k and standard deviation σ_s^k.

At test time, we find the class corresponding the nearest prototype for the selected domain, i.e., $k^* = \text{argmin}_{1 \le k \le K} \|\mathbf{z}^v - \boldsymbol{m}_{s^*}^k\|_2$. We then use the distribution $P = \mathcal{N}(\cdot\,; \mu_{s^*}^{k^*}, \sigma_{s^*}^{k^*})$ to determine whether $\Delta_{s^*}(\mathbf{x})$ is normal. Specifically, we classify a sample $\mathbf{x}$ as in-distribution if $F(\Delta_{s^*}(\mathbf{x})) \le q$ where F is the cumulative distribution function of P, i.e., $F(x) = P(X \le x)$ and q is a specified threshold.

Afterwards, if $\mathbf{x}$ is in-distribution, we use $p(\mathbf{y}_k \,|\, \mathbf{x}, s^*)$ to classify $\mathbf{x}$. Otherwise, $\mathbf{x}$ belongs to a new (unseen) domain. In such case, we propose the following ensembling technique to classify it.

2.2.2.2 Ensembling

If $\mathbf{x} \in \mathcal{D}_{s'}$ and $\mathcal{D}_{s'}$ has not been encountered during training, we model $\mathbf{z}^v$ as being part of a mixture of the known domains. In particular, we resort to a Gaussian mixture model to estimate the mixture weights ($w_s = p(s|\mathbf{x})$). While this could be done with L-dimensional covariance and mean vectors per domain (on the features), it does not perform well as L increases. We propose the following model:

$$
\begin{aligned}
w_s &= p(\mathbf{x} \in \mathcal{D}_s) \\
&= p(s' = s|\mathbf{x}) \\
&= \frac{p(\mathbf{x}|s) \cdot p(s)}{p(\mathbf{x})} \text{ (Bayes theorem)} \\
&= \frac{p(\mathbf{x}|s) \cdot p(s)}{\sum_j p(\mathbf{x}|j) \cdot p(j)} \text{ (Marginalization)} \\
&= \frac{p(\mathbf{x}|s)}{\sum_j p(\mathbf{x}|j)} \;(\mathcal{H}_1) \\
&= \frac{p(\Delta_s(\mathbf{x})|s)}{\sum_j p(\Delta_j(\mathbf{x})|j)} \;(\mathcal{H}_2) \\
&= \frac{\mathcal{N}(\Delta_s(\mathbf{x}); \mu_s^{k^*}, \sigma_s^{k^*})}{\sum_j \mathcal{N}(\Delta_j(\mathbf{x}); \mu_j^{t^*}, \sigma_j^{t^*})},
\end{aligned}
\tag{2.5}
$$

where $t^* = \text{argmin}_{1 \leq k \leq K} \|\mathbf{z}^v - \boldsymbol{m}_j^k\|_2$.

We have to make three assumptions or hypothesis to derive this model:

- $\mathcal{H}_1$: Each domain is of equal importance in our scenario, i.e. if we consider the probability of the sample belonging to a certain domain uniform when we have no a priori on the sample.
- $\mathcal{H}_2$: $p(\mathbf{x}|s) \approx p(\Delta_s(\mathbf{x})|s)$, i.e. the distribution of $f_\theta(\mathbf{x}_{\text{tok}})$ with $x_{\text{tok}} \in \mathcal{D}_s$ is isotropic.
- $\mathcal{H}_3$: $\Delta_s(\boldsymbol{x})|s \sim \mathcal{N}(\cdot; \mu_s^{k^*}, \sigma_s^{k^*})$, i.e. $\boldsymbol{x})|s$ follows a Gaussian of mean $\mu_s^{k^*}$ and standard deviation $\sigma_s^{k^*}$.

$\mathcal{H}_1$ is reasonable in practice as test sample can come from any domain with equal probability. $\mathcal{H}_2$ and $\mathcal{H}_3$ are made to simplify the model, make it easy to store in memory and to compute. These hypothesis transform the mixture weights model into a Gaussian Mixture Model on the distances to the prototypes (L2-GMM). Please note that in our case the ensembling with the Mahanalobis distance is equivalent to the well known classical GMM using directly the features and the prototypes to derive $p(\mathbf{x} \in \mathcal{D}_s)$.

We empirically observe in the ablation study (Table (4) in the main paper) that the usage of this Gaussian Mixture Model on the distances to the prototypes yields superior performance compared to a GMM using directly the features and the prototypes. We suspect that these approximations are efficient because they reduce the coordinate-wise noise in the standard deviations inherent to the Mahanalobis distance. Gaussian seems like a good approximation of $\Delta_s(\boldsymbol{x})|s$, even though the approximation using other distributions could be investigated in the future, such as the Weibull Distribution or the Generalized Pareto Distribution.

We then combine the predictions using the different prompts ($p(\mathbf{y}_k|\mathbf{x}, s)$) based on those weights:

$$p(\mathbf{y}_k|\mathbf{x}) = \sum_{s=1}^{N} p(\mathbf{y}_k|\mathbf{x}, s) \cdot w_s \tag{2.6}$$

Algorithm 1 Inference procedure for the proposed method. $\mathbf{x}$ denotes the samples to be classified, f_θ and f_ϕ the visual and text encoder of the network and $\mathcal{P}^V$, $\mathcal{P}^T$ the sets of visual of text prompts and $\mathcal{E}$ the domains prototypes learned during training. $\mathcal{G} = \{(\mu_s^k; \sigma_s^k), s = 1..N, k = 1..K\}$ denotes the parameters of the Gaussian distributions learned for the different domains s and classes k.

```
Input: $\mathbf{x}$; $f_\theta$; $f_\phi$; $\mathcal{P}^V$; $\mathcal{P}^T$; $\mathcal{E}$; $\mathcal{G}$;
Init $E \in O^{K \times N}$
Compute image features: $f_x \leftarrow f_\theta(\mathbf{x}_{tok})$
Compute matrix $D : D_{i,j} \leftarrow ||f_x - \boldsymbol{m}_j^i||_2$
Compute matrix $D' : D'_j \leftarrow \min_i D_{i,j}$
if $F(\Delta_{s^*}(\mathbf{x})) \leq q$ ($\mathbf{x}$ is In-Domain) then
    $W_{s^*} = 1, \forall s \neq s^*, W_s = 0.$
    Compute prediction using the best prompt:
    for $k = 1, 2, ..., K$ do
        $\mathbf{x}_{pro} \leftarrow [\mathbf{x}_{tok}, \mathbf{p}_{s^*}^v, x_{cls}]$
        $t_j \leftarrow \left[\mathbf{p}_{s^*}^t, c_j\right]$
        $E_{k,s^*} \leftarrow \frac{\exp(cos(f_\theta(\mathbf{x}_{pro}), f_\phi(t_k)))}{\sum_{i=1}^{C} \exp(cos(f_\theta(\mathbf{x}_{pro}), f_\phi(t_i)))}$
    end for
else
    Compute $W$ using equation (2.5), $D'$ and $\{(\mu_s^{k^*}, \sigma_s^{k^*})\}_{s=1}^{N}$.
    Compute predictions using the different prompts:
    for $s = 1, 2, ..., N$ do
        for $k = 1, 2, ..., K$ do
            $\mathbf{x}_{pro} \leftarrow [\mathbf{x}_{tok}, \mathbf{p}_s^v, x_{cls}]$
            $t_j \leftarrow \left[\mathbf{p}_s^t, c_j\right]$
            $E_{k,s} \leftarrow \frac{\exp(cos(f_\theta(\mathbf{x}_{pro}), f_\phi(t_k)))}{\sum_{i=1}^{C} \exp(cos(f_\theta(\mathbf{x}_{pro}), f_\phi(t_i)))}$
        end for
    end for
end if
$P \leftarrow E \cdot W^T$ Return P the soft classification vector
```

2.3 Experiments

The experiments reported in this section validate empirically that MoP-CLIP yields competitive performance compared to state-of-the-art DIL when dealing with in-domain (ID) examples, while significantly outperforming these approaches in the presence of out-of-domain (OOD) examples. Furthermore, we perform a series of ablation experiments to better identify the impact of the key components of the proposed method.

2.3.1 Experimental setup

2.3.1.1 Datasets

To assess the performance of the proposed method, we resort to three popular DIL benchmarks which have been extensively used in the literature: CDDB-Hard from Li *et al.* (2023), DomainNet from Peng *et al.* (2019), and CORe50 from Lomonaco & Maltoni (2017), whose details are given below:

CDDB Dataset (Li *et al.*, 2023) is a continual (incremental) deepfake detection benchmark, whose goal is to identify real and fake images across different domains. In particular, in the proposed work we employ the Hard setting as in (Wang *et al.*, 2022b), which is the most challenging track of CDDB. This dataset contains a total of 27,000 images across 5 different domains: GauGAN, BigGAN, WildDeepfake, WhichFaceReal, and SAN. We also use Glow, StarGAN and CycleGAN to evaluate OOD performance.

DomainNet (Peng *et al.*, 2019) is a dataset for domain adaptation commonly used to benchmark DIL methods. It contains a total of 600,000 images across 6 different domains, each containing the same 345 classes. In particular, we use the experimental setup presented in CaSSLe (Fini *et al.*, 2022).

CORe50 (Lomonaco & Maltoni, 2017) is a dataset designed for continual object recognition. However, in this work we focus on its domain-incremental learning scenario. This setting is comprised of 11 distinct domains, each containing the same 50 object categories. From the 11 domains, 8 are composed of 120,000 images which are seen sequentially during training, whereas the remaining 3 domains compose the fixed unseen test set.

2.3.2 Comparison methods

We benchmark MoP-CLIP to several state-of-the-art DIL methods. These include **non-prompting** approaches (EWC (Kirkpatrick *et al.*, 2017), LwF (Li & Hoiem, 2017), ER (Chaudhry *et al.*, 2019), GDumb (Prabhu, Torr & Dokania, 2020), BiC (Wu *et al.*, 2019b), DER++ (Buzzega, Boschini, Porrello, Abati & Calderara, 2020) and Co^{2}L (Cha, Lee & Shin, 2021)), **prompting-based** methods (L2P (Wang *et al.*, 2022d), DyTox (Douillard *et al.*, 2022) and S-lilPrompts (Wang *et al.*, 2022b)) and a **self-supervised** learning method, CaSSLe (Fini *et al.*, 2022), following the experimental set-up in (Wang *et al.*, 2022b). For OOD experiments, we only evaluate those methods that are in direct competition with our approach, in terms of *exemplars* buffer use. In particular, we compare to the following methods, whose respective codes are publicly available: EWC[1], LwF[2], DyTox[3], L2P[4], and S-liprompts[5].

2.3.2.1 Evaluation metrics and protocol

To assess the performance of the proposed approach, we resort to standard metrics in the incremental learning literature. **In-domain setting:** On DomainNet and CDDB-Hard we follow the original work in Li *et al.* (2023) and employ the average classification accuracy (AA), as

[1] https://github.com/G-U-N/PyCIL/

[2] https://github.com/G-U-N/PyCIL/

[3] https://github.com/arthurdouillard/dytox

[4] https://github.com/JH-LEE-KR/l2p-pytorch

[5] https://github.com/iamwangyabin/S-Prompts

well as the average forgetting degree (AF), which is the mean of the popular backward transfer degradation (BWT). We formally define the average accuracy as $AA = \frac{1}{N}\sum_{i=1}^{N} A_{i,N}$ with $A_{i,N}$ the accuracy on domain i measured after having trained on N domains. This metric is computed at the end, i.e., after having seen all the domains, e.g., on CDDB: GauGAN → BigGAN→ WildDeepfake→ WhichFaceReal→ SAN.

Furthermore, the average forgetting degree on CDDB can be defined as $\frac{1}{N-1}\sum_{i=1}^{N-1} BWT_i$ with $BWT_i = \frac{1}{N-i-1}\sum_{j=i+1}^{N}(A_{i,j} - A_{i,i})$ as originally proposed in Li *et al.* (2023) (i.e., the forgetting degree is computed for each domain at each adaptation step, then averaged). **Out-of-domain setting:** We follow Lomonaco & Maltoni (2017) to compute the AA on CORe50 on the fixed test set, which contains 3 hold-out splits that can be considered as OOD with respect to the training set. Furthermore, as in Wang *et al.* (2022b), we compute the AA on 3 unseen domains (Glow, StarGAN and CycleGAN) in CDDB-Hard. Last, as no independent hold-out subset of unseen domains exists for DomainNet, we propose using the Cumulative Accuracy on the unseen domains during the incremental learning of the model (i.e., average accuracy on the unseen domains averaged on all the steps), defined as follows: $CA = \frac{1}{N-1}\sum_{i=1}^{N-1}\frac{1}{N-j-1}\sum_{i=j}^{N} A_{i,j}$.

2.3.2.2 Implementation details

We use the same setting as Wang *et al.* (2022b), i.e. use ViT-B/16 (Dosovitskiy *et al.*, 2021) as our base image encoder and the text encoder of CLIP, both initialized by CLIP pretraining on ImageNet (Russakovsky *et al.*, 2015). We follow Wang *et al.* (2022b) and use the same image encoder model as a backbone (i.e., ViT-B/16 (Dosovitskiy *et al.*, 2021) pretrained on ImageNet (Russakovsky *et al.*, 2015)) across all the compared methods, for a fair comparison. As suggested in Wang *et al.* (2022b), we use a more advanced backbone (i.e. ConViT pretrained on ImageNet (Russakovsky *et al.*, 2015)) on DyTox (Douillard *et al.*, 2022) as it underperforms a random model with ViT-B/16 as backbone. We empirically fix $q = 0.94$ for the 3 datasets, based on the ablation study in Figure 2.4, such that we do not deteriorate ID performance while improving OOD performance on CDDB-Hard. For EWC, LwF and CaSSLe, we use the same hyperparameters as in the original papers, whereas we keep the hyperparameters reported in Wang *et al.* (2022b) for DyTox, L2P and S-Prompts.

2.4 Results

2.4.1 In-domain distributions

We first evaluate the proposed approach in the standard DIL scenario where the testing samples are drawn from the same distribution as the training/adaptation images. These results, which are reported under the *Seen-Domains* columns of Tables 2.1 and 2.2, demonstrate that the proposed MoP-CLIP approach yields superior performance than existing *exemplar-free* methods. In particular, MoP-CLIP outperforms the very recent approaches DyTox (Douillard *et al.*, 2022) and L2P (Wang *et al.*, 2022d) by large margin, with improvement gains of around 20-30% in terms of average classification accuracy under the same storage conditions. Furthermore, the degree of knowledge forgetting is also largely reduced, going from -45.85 in DyTox to -0.79 in our approach. Furthermore, if storing exemplars is allowed, DyTox (Douillard *et al.*, 2022) significantly improves its performance, but still underperforms our approach yet incurring a non-negligible overhead. Last, it is noteworthy to highlight that the proposed approach reaches similar performance than S-liPrompts (Wang *et al.*, 2022b) in this scenario, with at par values in the CDDB-Hard dataset and remarkable performance gains in DomainNet. Note that this result is somehow expected, as our approach is a generalization of S-liPrompts for the OOD scenario, and differences in the in-distribution setting may come from the domain prompt selected.

An interesting observation is that prompting-based methods, which do not store exemplars from old tasks, typically outperform their buffer-storage counterparts. For example, S-liPrompts (Wang *et al.*, 2022b) and MoP-CLIP bring considerable improvements compared to LUCIR (between 6-8%) or iCaRL (ranging from 9 to 15%). We hypothesize that this phenomenon comes from the absence of interference between domains when doing the adaptation. In this scenario, the knowledge from previously learned domains remains isolated in the form of optimized domain prompts, and the only knowledge shared is derived from pre-trained transformers.

Table 2.1 **Results on CDDB-Hard for both ID and OOD scenarios.** Evaluation of existing state-of-the-art DIL methods in the standard *seen-domain* setting and more challenging *unseen-domain* scenario. For the unseen-domain experiments, we only reproduced the results for related (i.e., *exemplar-free*) methods. Best results are highlighted in **bold**.

Method	Prompts	Buffer size	Seen-Domains AA (↑)	Seen-Domains AF (↓)	Unseen-Domains AA (↑)
LRCIL $_{\text{IROS'20}}$(Pellegrini, Graffieti, Lomonaco & Maltoni, 2020b)	✗	*100ex/class*	76.39	-4.39	-
iCaRL $_{\text{WIFS'19}}$(Marra, Saltori, Boato & Verdoliva, 2019)	✗		79.76	-8.73	-
LUCIR $_{\text{CVPR'19}}$(Hou, Pan, Loy, Wang & Lin, 2019)	✗		82.53	-5.34	-
LRCIL $_{\text{IROS'20}}$(Pellegrini *et al.*, 2020b)	✗	*50ex/class*	74.01	-8.62	-
iCaRL $_{\text{WIFS'19}}$(Marra *et al.*, 2019)	✗		73.98	-14.50	-
LUCIR $_{\text{CVPR'19}}$(Hou *et al.*, 2019)	✗		80.77	-7.85	-
DyTox $_{\text{CVPR'22}}$(Douillard *et al.*, 2022)	✓		86.21	-1.55	-
EWC $_{\text{PNAS'17}}$ (Kirkpatrick *et al.*, 2017)	✗	*No buffer*	50.59	-42.62	-
LwF $_{\text{TPAMI'17}}$ (Li & Hoiem, 2017)	✗		60.94	-13.53	50.05
DyTox $_{\text{CVPR'22}}$(Douillard *et al.*, 2022)	✓		51.27	-45.85	50.46
L2P $_{\text{CVPR'22}}$(Wang *et al.*, 2022d)	✓		61.28	-9.23	57.34
S-liPrompts $_{\text{NeurIPS'22}}$(Wang *et al.*, 2022b)	✓		**88.65**	**-0.69**	76.79
MoP-CLIP (ours)	✓		88.54	-0.79	**82.02**

Table 2.2 **Results on DomainNet for both ID (AA metric) and OOD (CA metric) scenarios**. Best values are highlighted in **bold**.

Method	Prompt	Buffer size	Seen Domains	Unseen Domains
DyTox $_{\text{CVPR'22}}$(Douillard *et al.*, 2022)	✓	*50ex/class*	62.9	
DyTox $_{\text{CVPR'22}}$(Douillard *et al.*, 2022)	✓	*No buffer*	13.5	4.2
LwF $_{\text{TPAMI'17}}$ (Li & Hoiem, 2017)	✗		49.2	43.4
CaSSLe $_{\text{CVPR'22}}$(Fini *et al.*, 2022)(SimCLR Chen, Kornblith, Norouzi & Hinton (2020))	✗		48.1	45.4
CaSSLe $_{\text{CVPR'22}}$(Fini *et al.*, 2022)(BYOL Grill *et al.* (2020))	✗		52.9	48.7
CaSSLe $_{\text{CVPR'22}}$(Fini *et al.*, 2022)(Barlow TwinsZbontar, Jing, Misra, LeCun & Deny (2021))	✗		51.4	47.6
CaSSLe $_{\text{CVPR'22}}$(Fini *et al.*, 2022)(SupCon Khosla *et al.* (2020))	✗		54.2	50.5
L2P $_{\text{CVPR'22}}$(Wang *et al.*, 2022d)	✓		40.1	25.5
S-liPrompts $_{\text{NeurIPS'22}}$(Wang *et al.*, 2022b)	✓		67.7	66.4
MoP-CLIP (Ours)	✓		**69.7**	**67.0**

2.4.2 Performance under domain distributional shift.

We now want to assess the benefits of the proposed approach when the testing dataset presents a distributional drift over the training data. In particular, we advocated that the proposed approach is a generalization of Wang *et al.* (2022b) to be able to handle samples coming from an unseen distribution. To support this claim, and to demonstrate the superiority of our approach on unseen domains, we resort to the OOD experiments, which are reported in the right-most columns of Tables 2.1 and 2.2, as well as Table 2.3. From these results, we can observe that excluding S-liPrompts, the performance gains brought by the proposed approach are substantial compared to other *exemplar-free* methods, ranging from 17% (EWC in CORe50) to 40% (L2P (Wang *et al.*, 2022d) in DomainNet). Even when comparing to state-of-the-art competitors that store exemplars (e.g., DyTox (Douillard *et al.*, 2022) or Co^{2}L (Cha *et al.*, 2021) in CORe50), MoP-CLIP yields considerable improvements, ranging from 11% to nearly 17%. The clear superiority of our approach lies on the isolation of different domains during learning, which do not degenerate the generalization capabilities brought by the pre-trained transformers. Furthermore, when comparing the proposed MoP-CLIP to S-liPrompts (Wang *et al.*, 2022b), we observe that our method outperforms the latter by around 6%, 2% and 3% in CDDB-Hard, DomainNet and CORe50 benchmarks, respectively. These performance gains on OOD samples might likely come from the flexibility of MoP-CLIP in selecting a subset of similar domains for a given test sample, which allows the model to properly weight the contribution of each domain prompt. In contrast, S-liPrompts (Wang *et al.*, 2022b) forces the model to select only one domain from the seen domains, which impedes its scalability to novel distributions, as empirically shown in these results, as well as in Figure 0.1.

Table 2.4 emphasizes that S-Prompts performances degrade when evaluation is done on unseen domains, and shows that the proposed MoP-CLIP seems to generalize better, mitigating the performance degradation under domain distributions. In particular, the left-side section reports

Table 2.3 **Results on CORe50.** Note that CORe50 already provides separate training and testing domains, and thus results can only be computed on the **OOD scenario**. Results are reported as the Acc metric, where the best values are highlighted in **bold**. In our method, we use the same q as in the other datasets, whereas * indicates that q is fixed based on the validation set of CORe50, as typically done in all the other approaches.

Method	Prompt	Buffer size	AA
GDumb ECCV'20 (Prabhu *et al.*, 2020)	✗	*50ex/class*	74.92
BiC CVPR'19 (Wu *et al.*, 2019b)	✗		79.28
DER++ NeurIPS'20 (Buzzega *et al.*, 2020)	✗		79.70
Co^{2}L ICCV'21 (Cha *et al.*, 2021)	✗		79.75
DyTox CVPR'22 (Douillard *et al.*, 2022)	✓		79.21
L2P CVPR'22 (Wang *et al.*, 2022d)	✓		81.07
EWC PNAS'17 (Kirkpatrick *et al.*, 2017)	✗	*No buffer*	74.82
LwF TPAMI'17 (Li & Hoiem, 2017)	✗		75.45
L2P CVPR'22 (Wang *et al.*, 2022d)	✓		78.33
S-liPrompts NeurIPS'22 (Wang *et al.*, 2022b)	✓		89.06
MoP-CLIP (Ours)	✓		**91.43**
MoP-CLIP (Ours)*	✓		**92.29**

the results of S-Prompts trained separately on the different domains (*x-axis*) and evaluated in each of the domains (*y-axis*). For example, 67.41 denotes the accuracy of the model trained solely on Infograph domain and tested on the Clipart domain. We use blue to denote the performance of in-distribution samples (when train and test data are drawn from the same distribution), which can be considered as an upper bound, as there is no distributional drift between samples. Then, both results in black and magenta highlight the results for each tested domain, assuming that the tested domain remains unknown and all training samples come from the same domain (specified in each column). Note that across each test domain we highlight the results from the best model in magenta. If we look at the results obtained by S-Prompts under ID and OOD conditions (*S-Prompts (ID)* and *S-Prompts (OOD)* columns), we can observe that: *i)* its performance deteriorates under domain shift and *ii)*, the selection criterion of S-Prompts is not always optimal.

On the other hand, the proposed approach (*last column*) substantially outperforms S-Prompts in five out of six domains, as well as the best out-of-distribution model (in magenta).

Table 2.4 **Empirical motivation of resorting to the prediction ensembling scheme for OOD situations.** Classification accuracy across DomainNet domains using different specialized prompts, for both single and ensembling predictions. The results in blue denote the accuracy with the in-domain prompts, whereas results in magenta denote the accuracy using the best out-of-domain prompts (prompts from all domains except the current one). Furthermore, results in bold (*last column*) denote the highest accuracy amongst out-of-domain methods. For 5 out of 6 domain sets, the proposed prediction ensembling method yields higher accuracy than the best out-of-domain prompt. This suggests that the ensembling technique is overall relevant when test examples are from a novel domain (i.e. unseen during the training).

	Clipart	Infograph	Painting	Quickdraw	Real	Sketch	S-Prompts (ID)	S-Prompts (OOD)	Pred. Ens. (OOD)
Clipart	80.14	67.41	64.77	38.9	69.49	69.02	78,57	69,31	**73.48**$_{(+4.01)}$
Infograph	44.59	60.65	43.24	15.36	48.93	36.08	58,72	46.50	**50.40**$_{(+1.47)}$
Painting	59.56	61.88	78.00	24.97	64.43	57.32	74,76	61,88	**67.93**$_{(+3.50)}$
Quickdraw	16.8	13.11	8.30	46.65	13.58	**17.29**	46,59	16,79	16.78$_{(-0.51)}$
Real	78.35	79.38	75.83	45.44	87.94	71.79	85,19	77,38	**83.48**$_{(+4.10)}$
Sketch	61.51	59.18	55.22	30.43	61.59	72.97	69,76	58,87	**66.31**$_{(+4.72)}$

2.4.3 On the impact of the different components.

The empirical study in Table 2.5 justifies the need of employing the proposed approach over the strong baseline S-liPrompts (Wang *et al.*, 2022b), as well as showcases the impact of each choice. In a practical scenario, it is unrealistic to assume that the test samples always follow the same distribution as the data used for adaptation. Furthermore, the domain of each sample typically remains unknown. Thus, to align with real-world conditions, we will consider the average of in-distribution and out-of-distribution performance as our metric of reference to evaluate the impact of the different choices. We can observe that in nearly all the cases, the use of an ensembling strategy results in consistent improvements over the single model predictions (considering same distances). An interesting observation is that distances related to the L_2-norm typically degrade the performance on ID samples. We observe that in this scenario, the distributions overlap considerably and $p(s|\mathbf{x})$ (derived from the Gaussian mixture) is too far from 1 for most ID samples, making the discrimination of samples by these distance measures difficult. Nevertheless, this behavior is reversed in the presence of OOD samples. In particular, our simplification assumes an isotropic Gaussian distribution of the points around the prototypes and therefore reduces the noise in the coordinate-wise variances (which can explain the performance degradation observed when using the Mahanalobis distance), replacing it with distance-wise variances. Thus, the proposed approach combines the best of both worlds, leading to the best average performance across all the configurations.

Table 2.5 **Impact of each design choice of MoP-CLIP .** *Maha* denotes the Mahanalobis distance, whereas GMM is used for a Gaussian Mixture Model. Furthermore, *Hybrid* denotes the nature of our approach, which uses an ensembling for OOD samples and a single domain prompt for ID samples. Results (on CDDB-Hard) show the average accuracy (AA), with the deviation from the baseline S-liPrompts Wang *et al.* (2022b) in brackets. Best results in **bold**.

Method	Ensembling	Distance	Seen Domains	Unseen Domains	Mean
S-liPrompts (Wang *et al.*, 2022b)	✗	L1	88.65	76.79	82.72
MoP-CLIP - no ens. (a)	✗	L2	**89.48**	76.95	83.22$_{(+0.50)}$ ↑
-	✗	Maha	80.45	76.66	78.56$_{(-4.16)}$↓
-	✗	L2-GMM	75.72	75.76	75.74$_{(-6.98)}$↓
-	✓	Uniform	67.55	83.61	75.58$_{(-7.14)}$ ↓
-	✓	L1	89.29	80.05	84.67$_{(+1.95)}$↑
-	✓	L2	68.37	84.07	76.22$_{(-6.50)}$↓
-	✓	Maha	80.48	77.56	79.02$_{(-3.70)}$↓
MoP-CLIP - ens. (b)	✓	L2-GMM	72.51	**89.21**	80.86$_{(-1.86)}$↓
MoP-CLIP (Proposed)	Hybrid	ID (a)/ OOD (b)	88.54	82.02	**85.28**$_{(+2.56)}$↑

2.4.4 Strategy to select the domain prompts.

As emphasized in Sec. 2.2.2, Wang *et al.* (2022b) uses K-Means over the features extracted with a pre-trained ViT to compute the prototypes which are used to dynamically select which prompt to use at test time. While this strategy is memory efficient, it lacks flexibility, as the number of clusters needs to be adjusted according to the dataset employed. To alleviate this issue, we instead use class-wise prototypes as a *hyperparameter-free* alternative to compute representative prototypes. The effect of using either k-Means or class-prototypes is depicted in Fig. 2.3. From these results, we empirically observe that this choice improves performance in both in-distribution and out-of-distribution domains, leading to a higher average performance. Furthermore, it is noteworthy to mention that using class-wise prototypes makes the distribution of points around prototypes Gaussian, which explains the satisfactory performance of MoP-CLIP, particularly on samples from unseen domains.

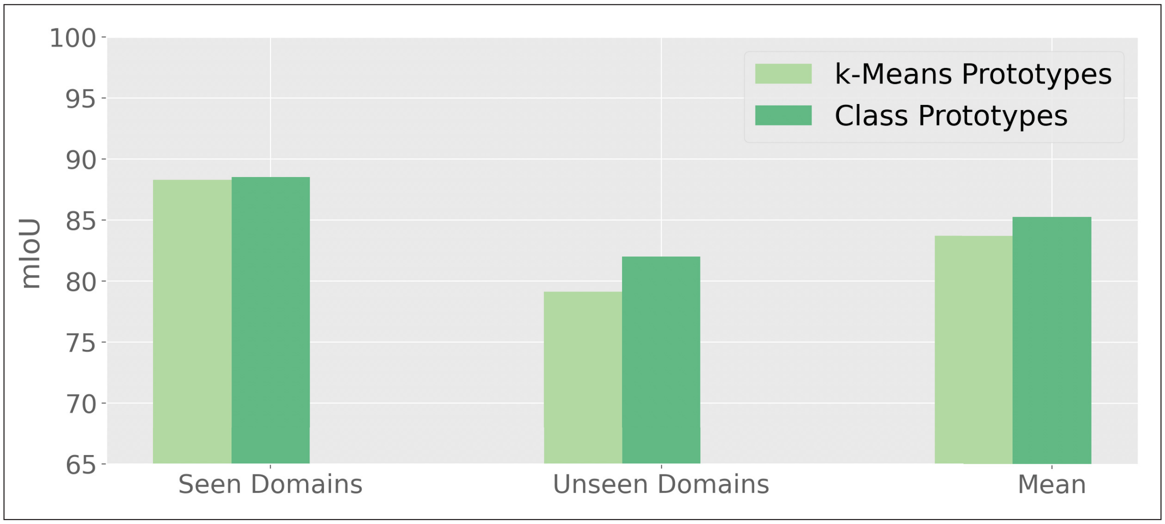


Figure 2.3 **k-Means or class prototypes as domain centroids?** Ablation study that demonstrates the benefits of using class prototypes (our approach) rather than k-Means prototypes, as in Wang *et al.* (2022b).

2.4.5 How much trade-off is sufficient?

The influence of the threshold q from our simple out-of-distribution criterion (Sec. 2.2.2) to select between seen and unseen domains is shown in Figure 2.4. As stressed earlier, we aim for a compromise between ID and OOD performance, in order to provide generalizable models. As target domains should remain unknown at inference, we selected a fixed q value that provided the optimal average performance across both settings. Nevertheless, these plots reveal two interesting findings. First, the average performance of the model is not very sensitive to the choice of q. For example, the performance of ID samples decreases as q decreases, whereas

OOD performance improves. On the other hand, if q increases, the accuracy in the ID scenario increases, while it decreases for OOD samples. And second, if prior knowledge about the target domain is available –an assumption made by all existing DIL literature– the performance of MoP-CLIP is further increased, enlarging the gap with SOTA methods.

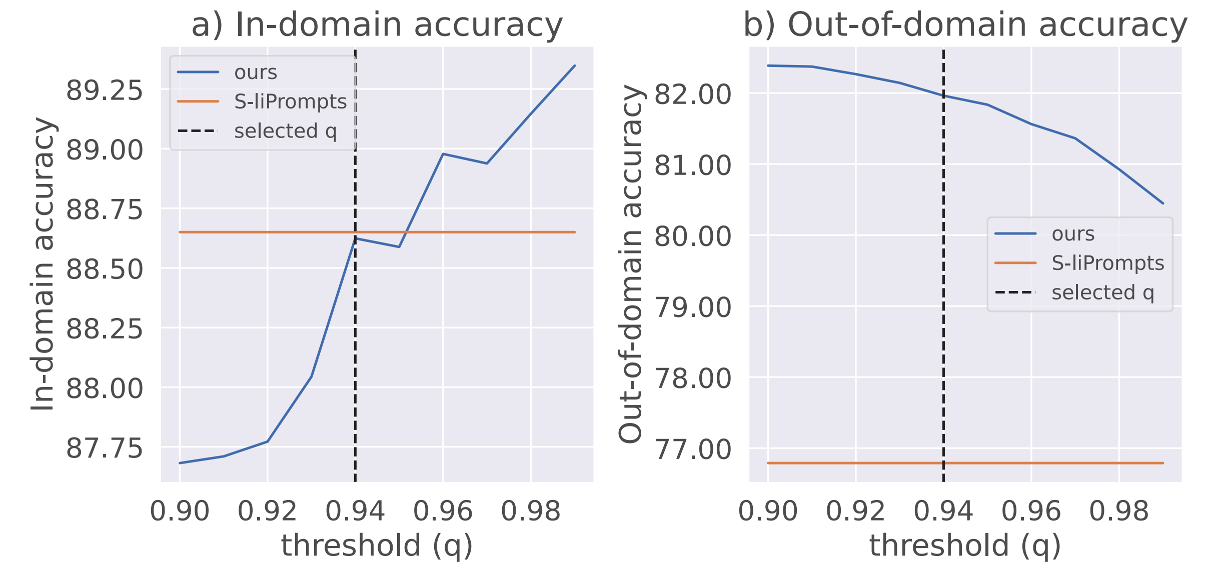


Figure 2.4 **A controllable trade-off between in-domain and out-of-domain prediction performances.** Impact of the threshold q (Sec. 2.2.2) on the accuracy, evaluated on CDDB-Hard.

2.5 Conclusion

Findings from this work reveal that existing literature on domain incremental learning suffers under the presence of distributional drift, hampering their scalability to practical scenarios. To overcome this issue, we have proposed a generalization of the recent S-ilPrompts (Wang *et al.*, 2022b) approach, that further handles out-of-distribution samples. In addition to outperforming current state-of-the-art, particularly in the unseen domain setting, our method brings several interesting benefits compared to most existing DIL method. First, MoP-CLIP is *exemplar-free*, eliminating the limitations of conventional DIL approaches in terms of storage and privacy. Furthermore, as prompts are learned independently on each domain, and the model parameters remain fixed during the adaptation, the performance of our approach is insensitive to the ordering of the seen domains. This contrasts with a whole body of the literature, where the choice of the sequence order can significantly impact the final performance. Our comprehensive evaluation shows the empirical gains provided by MoP-CLIP, pointing to visual prompt tuning as an appealing alternative for general domain incremental learning. Finally, we stress that while powerful, the proposed approach retains the spirit of S-ilPrompts (Wang *et al.*, 2022b), which advocates for a simple yet elegant method. The main limitation of our method is the increase in computational complexity, as our method requires one more model forward pass to be performed for each domain in the sequence.

CHAPTER 3

DOMAIN-INCREMENTAL FORGERY DETECTION

3.1 Introduction

The aim of this chapter is to test the applicability of the previously explored domain-incremental learning methods in a real world scenario. In such scenario, each domain is introduced sequentially and it is non-trivial to tune hyperparameters before deploying the model. Moreover, it is important that the deployed model generalizes to unseen domains, as attackers might use strategies to attempt to induce a domain shift between the known forgeries and the new counterfeit image. We show that SOTA forgery detection models are not robust to image corruptions and propose resorting to DIL methods to improve their robustness. We also emphasize that common DIL methods improve their robustness under this setting, but are outperformed by prompt learning methods such as S-Prompts (Wang *et al.*, 2022b) and our proposed MoP-CLIP.

3.2 Method

3.2.1 Domain-incremental learning methods

We adapt SOTA forgery detection network OSN (Wu, Zhou, Tian & Liu, 2022) using three well-known regularization methods in continual learning: EWC from Aich (2021), LwF from Li & Hoiem (2017) and Synaptic Intelligence from Zenke *et al.* (2017). It has been empirically shown that these methods perform competitively on a wide range of DIL scenarios (Oren & Wolf, 2021) and necessitate no architecture change nor large computation overhead or storage of replay buffer. The former is very important in our case as forgery detection approaches have very specific architectures, making the adaptation of some continual learning methods (such as our proposed MoP-CLIP (2) using a ViT) to this setting not straightforward. It can be noted that the architecture of our proposed MoP-CLIP is not specialized in forgery detection but still performs competitively compared to SOTA forgery detection methods adapted to DIL.

3.2.2 Forgery detection methods

We use four state-of-the-art forgery detection and localization networks introduced in the literature review (1.2.1) as base models: MVSS-Net (Dong *et al.*, 2022), OSN (Wu *et al.*, 2022), CAT-Net (Kwon, Yu, Nam & Lee, 2021) and ManTraNet (Wu *et al.*, 2019a). We chose these networks as they are recent, are strong baselines and share their inference code and pretrained model checkpoints.

3.3 Experiments

3.3.1 Metrics

We use pixel and image level F1-score to measure the forgery detection performances of the models, following previous works (Salloum, Ren & Kuo, 2018; Zhou *et al.*, 2020). For a dataset of N images, we have:

$$\text{Image level F1} = \frac{2 \times TP}{2 \times TP + FN + FP} \tag{3.1}$$

with

$$\text{TP} = \sum_{k}^{N} \mathbb{1}_{\hat{y}_k=1, y_k=1}\ , \text{FP} = \sum_{k}^{N} \mathbb{1}_{\hat{y}_k=1, y_k=0}\ , \text{FN} = \sum_{k}^{N} \mathbb{1}_{\hat{y}_k=0, y_k=1} \tag{3.2}$$

and y_k the true image level label of an image x_k and $\hat{y}_k$ the image level prediction of the network for this image.

The pixel level F1 score is computed as:

$$\text{Pixel Level F1} = \frac{1}{N} \sum_{k}^{N} \frac{2 \times TP}{2 \times TP + FN + FP} \tag{3.3}$$

with

$$\mathrm{TP} = \sum_{i,j} \mathbb{1}_{\hat{M}^k_{i,j}=1, M^k_{i,j}=1}\ , \mathrm{FP} = \sum_{i,j} \mathbb{1}_{\hat{M}^k_{i,j}=1, M^k_{i,j}=0}\ , \mathrm{FN} = \sum_{i,j} \mathbb{1}_{\hat{M}^k_{i,j}=0, M^k_{i,j}=1} \tag{3.4}$$

and $M^k \in \{0,1\}^{H\times W}$ the true pixel level label map of the image x_k and $\hat{M}^k \in [0,1]^{H\times W}$ the segmentation map inferred by the network. $M^k_{i,j}$ and $\hat{M}^k_{i,j}$ denotes elements (i, j) of M_k and $\hat{M}_k$.

3.3.2 Datasets

CASIA v2 (Dong *et al.*, 2013) is a dataset for forgery classification and segmentation comprised of 4,795 images, 1,701 pristine and 3,274 forged. WEI (Sun, Zhou, Li, Cheung & She, 2020) is another forgery detection dataset of 1,000 manipulated images. DRESDEN (Gloe & Böhme, 2010) was initially created for media forensics and camera identification. It is comprised of 14,000 images from 73 different cameras. MS-COCO (Lin *et al.*, 2014) is a dataset of 328, 000 images for object captioning, captioning and segmentation. IMD2020 (Novozamsky, Mahdian & Saic, 2020) is a large scale forgery detection dataset of 35,000 real images and 70,000 manipulated ones.

3.3.3 Experimental setup

We use the official code implementations and checkpoints of the four forgery detection networks. For MVSS-Net, we use the checkpoints of the model pretrained on CASIA v2 (Dong *et al.*, 2013). OSN different components are pretrained on WEI (Sun *et al.*, 2020), DRESDEN (Gloe & Böhme, 2010) and MS-COCO (Lin *et al.*, 2014). CAT-NET uses CASIA v2 (Dong *et al.*, 2013), MS-COCO (Lin *et al.*, 2014) and IMD2020 (Novozamsky *et al.*, 2020) and MantraNet uses a custom dataset based on MS-COCO (Lin *et al.*, 2014).

We perform the DIL experiments on OSN as it is not originally trained on a variant of CASIA and it is the only method for which the authors release the training code necessary to adapt the model to $iCasia$, and the other models perform very poorly when finetuned in $iCasia$. We use the methodology of Dong *et al.* (2022) to give a classification prediction from segmentation maps.

We use the default hyperparameters of EWC (Aich, 2021), LwF (Li & Hoiem, 2017), SI (Zenke *et al.*, 2017), S-Prompts (Wang *et al.*, 2022b) and our proposed MoP-CLIP (2) to evaluate their performance on an unknown sequence, where it is unrealistic to perform hyperparameter tuning.

To test the generalization abilities of the different models, we split the forgery detection dataset CASIA v1 (Dong *et al.*, 2013) in 13 disjoint shards. We use 1 split as a clean training set representative of the original dataset. We transform 11 disjoint splits using the corruptions in Figure 3.1.

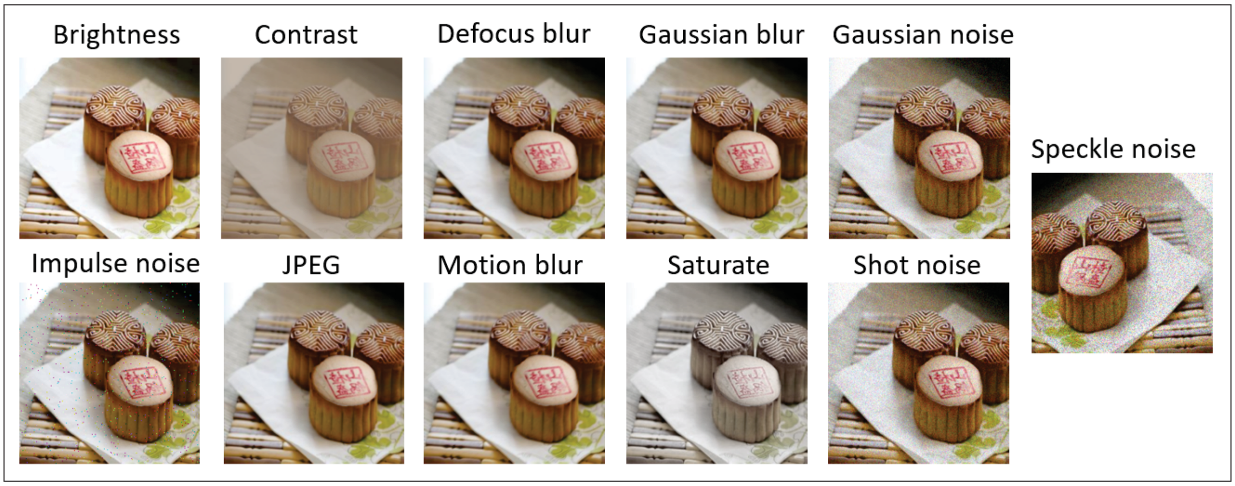


Figure 3.1 Different corruptions used to construct *iCasia*

We use the last split to generate 11 different test sets, transforming it with the introduced corruptions. This is sufficient as we do not need to perform any hyper-parameter tuning. We call the sequence comprised of the clean split and the corrupted splits *iCasia*.

We use the SOTA forgery detection network OSN (Wu *et al.*, 2022) as example in the DIL scenario, since the authors released the training code and we empirically observed that the other models failed in the training on $iCasia$ (we suspect that it is due to the small size of each of its splits).

3.3.4 Experiments

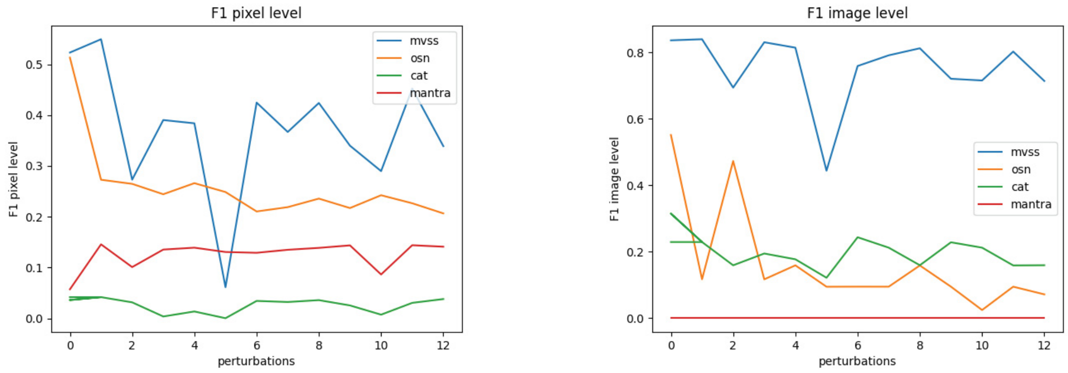
We first put forward the sensibility of the different SOTA forgery detection networks to domain shifts by evaluating them on all the domains presented (clean CASIA v2 and $iCasia$). As their performance is not satisfactory (Fig. 3.2), it is then necessary to resort to strategies to alleviate this performance drop.

Prototypical approaches such as (Snell, Swersky & Zemel, 2017) are widely-used to adapt a model to a dataset after a domain-shift. They are, however, not efficient in our case as the domain-shift is low-level (Lee *et al.*, 2022), i.e. the semantics of the images do not change but the image can be blurred or contain noise. This is because the most important parameters to adapt the model for these specific domain shifts are in the first layers, making of the standard last layer adaptation strategy an ineffective solution. We therefore use well-known regularization methods: EWC, LwF and SI, adapting deeper layers of the model. We also assess the performance of S-Prompts and the proposed MoP-CLIP on this problem.

3.4 Results

3.4.1 Naive evaluation

We can note that the pixel-wise F1 and the image level F1 scores (Fig. 3.2) of the four different models on the corrupted splits drop compared to those on the clean splits (index 0 and 1). Indeed, we can observe how the different models are not robust to the different corruptions.



a) Image Level F1 score

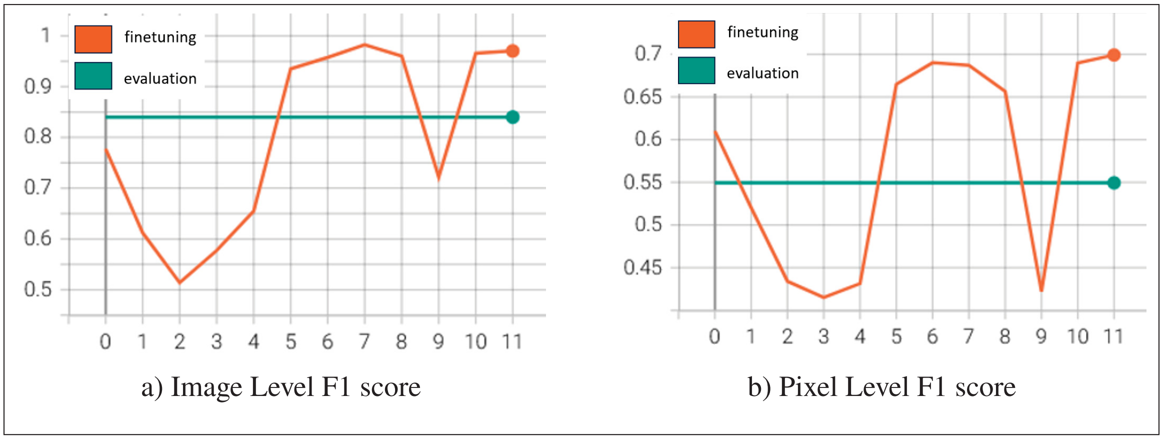
b) Pixel Level F1 score

Figure 3.2 F1 scores on *iCasia* for image and pixel level predictions for images under different perturbations

3.4.2 Naive finetuning

To show the need for DIL methods, we sequentially finetune MVSS-Net (Dong *et al.*, 2022) on the 12 *iCasia* domains. We sequentially compute the image and pixel level F1 scores on the clean CASIA v1 split after each finetuning step (for the 12 domains of the sequence).

We can see in Figure 3.3 that the image and pixel level F1 scores on clean CASIA v1 vary considerably after finetuning on some domains. For instance, image level F1 socre is ~0.50 after finetuning on the contrast corrupted domain while it is ~0.95 after finetuning on the JPEG corrupted domain. We can see the performance of the model on the previously seen domains is therefore unstable and not always preserved, which is another instance of *catastrophic forgetting*. We hypothesize that some domains are too different from the clean CASIA v1 split (such as the brightness, contrast and defocus blur domains) while some others act as data augmentation and boost the performance on the clean CASIA v1 split (such as the JPEG compression or the motion blur).



a) Image Level F1 score

b) Pixel Level F1 score

Figure 3.3 Illustration of catastrophic forgetting for MVSS-Net: Scores on CASIA v1 clean shard after finetuning on domain i (x-axis) or naively evaluating the base model

3.4.3 Regularization methods

Figures 3.4 and 3.5 show the comparisons of F1 scores obtained by different training strategies: direct inference after training on the base domain (clean CASIA v1), naive finetuning on $iCasia$, Elastic Weight Consolidation, Learning Without Forgetting (LwF) and Synaptc Intelligence (SI). Fig. 3.4 shows the F1 score on the base domain (clean CASIA v1) after application of the strategy on the i^{th} corrupted CASIA shard. Figure 3.4 depicts the F1 score on the i^{th} corrupted CASIA shard after adaptation on it.

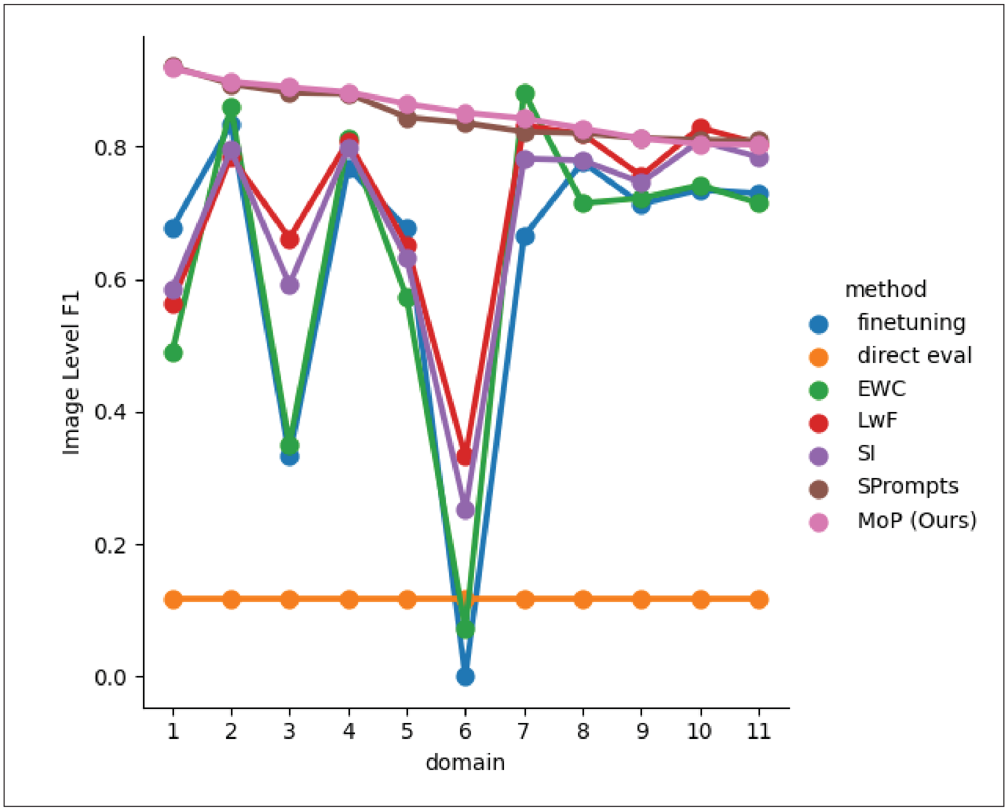


Figure 3.4 Comparison of F1 score on clean CASIA v1, using different training methods and the OSN model

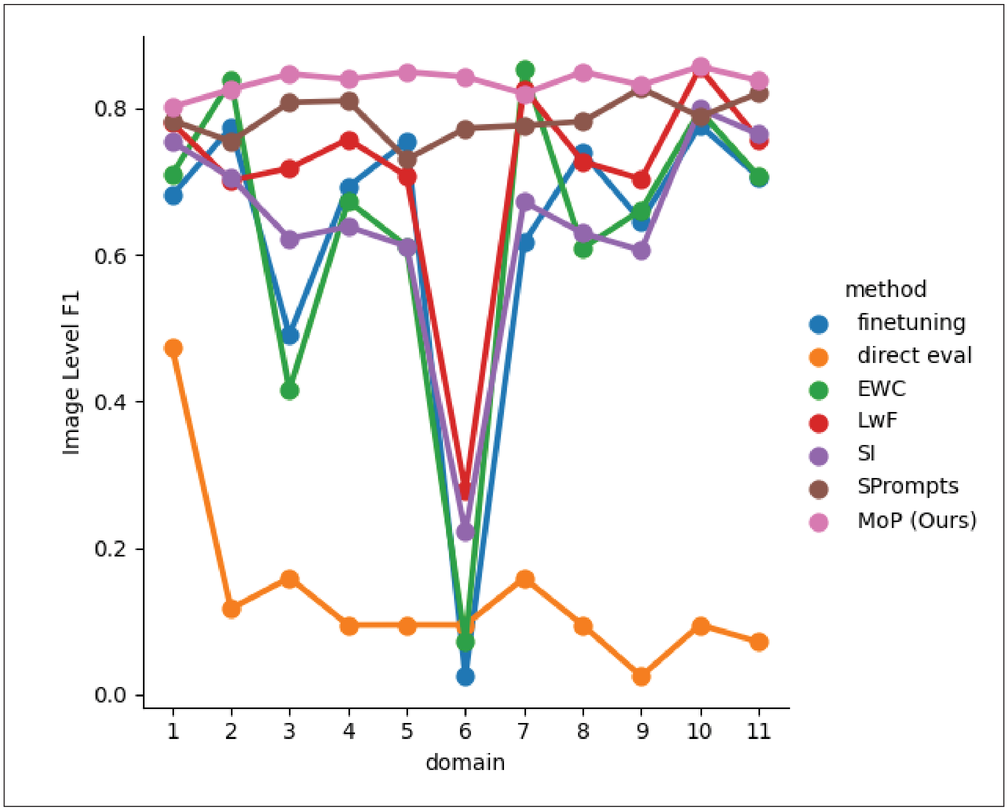


Figure 3.5 Comparison of F1 score on the i^{th} corrupted CASIA shard between training methods for OSN

Table 3.1 **Results on iCasia for both ID (AA metric) and OOD (CA metric) scenarios**. Best values are highlighted in **bold**.

Method	Prompt	Seen Domains	Unseen Domains
OSN (EWC)	✗	67.45	66.39
OSN (SI)	✗	69.57	67.21
OSN (LwF)	✗	71.39	68.86
S-liPrompts $_{\text{NeurIPS'22}}$(Wang *et al.*, 2022b)	✓	83.05	73.59
MoP-CLIP (Ours)	✓	**86.38**	**78.90**

We can see that prompt-based S-Prompts and MoP-CLIP methods are the most stable across domains and give the best F1 scores for most domains on the base domains and on the different shards, although forgetting on the base domain still happens. We show in 3.1 that MoP-CLIP still outperforms its most serious competitor S-Prompts on seen domains and unseen domains using the metrics introduced in 2 for DomainNet Peng *et al.* (2019).

3.5 Conclusion

Prompt-based models such as S-Prompts and the proposed MoP-CLIP seem to perform well on the challenging problem of forgery detection *without the need to tune their hyperparameters and even though their architecture is not specialized*. A bigger sequential experiment needs to be carried to confirm these findings. However, the compute power needed to carry this experiment is exponential in the number of datasets in the sequence as one needs to test multiple permutations of the datasets to confirm the findings for EWC, LwF ans SI as their performance depend on the domains order. It would also be interesting to compare the different regularization methods with more baselines.

CONCLUSION AND RECOMMENDATIONS

This thesis centered around the challenges in machine learning related to distributional shift, particularly the phenomenon of catastrophic forgetting. The storage and privacy concerns linked to common DIL solutions led us to consider exemplar-free DIL approaches such as prompt-learning and distillation-based methods.

Prompt-learning emerged as a viable alternative with its potential benefits discussed in Chapter 2 (2). It offered a compelling strategy for mitigating knowledge forgetting without the need for exemplar storage, demonstrating its effectiveness across various benchmarks.

However, in our opinion the common benchmarks for DIL rely on simple datasets which do not accurately represent real-world complexity. To address this, we introduced a more challenging problem – forgery detection in digital images – and performed a realistic evaluation of common DIL methods on it. We empirically demonstrated the validity of our proposed method MoP-CLIP in this context in Chapter 3 (3).

In conclusion, while prompt-learning presents a promising direction, it is clear that further work is needed to adapt incremental learning methods to complex, real-world problems like forgery detection.

APPENDIX I

POTENTIAL CONTRIBUTIONS

1. Pre-processing

We could add priors to the model loss given domain expertise on image manipulations. This would be particularly useful to train with few data for guiding the network to have a better generalization.

We know that the image manipulations are compact, and it is possible to draw a fully connected border of the manipulated zone where every pixel inside the zone are manipulated. As in Yuan & Xu (2021), this can be taken into account instead of treating every pixel in isolation for segmentation tasks. Such strategy could be combined seamlessly with our method.

It also could be interesting to add feature views to the model to create a hybrid neural network. While in traditional CNN models the representation and classifiers are learned end to end, as we have seen, it can be interesting to add priors to the models to help the model training converge in a low data regime. These priors could be leveraged in the form of views, which means that a fixed transformation is applied to the RGB image and the transformation result is added as input along with the classical RGB one (changing the input channel number).

For instance, MVSS-Net Dong *et al.* (2022) leverages this idea and add a constrained convolution view to the classical RGB image view. This constained convolution view (BayarConv (Bayar & Stamm, 2018)) is meant to reconstruct the image locally and to maximize the reconstruction error for regions that are statistically far from the rest of the image. It also would be possible to add Error Level Analysis (compression artifacts analysis, a feature directly used in the forensics community to detect manipulation, without a CNN) features to the model and Discrete Cosine Transform or Wavelet transform views to help the model find manipulation frequency clues.

It would be even more powerful to model these transformations as convolutions in order to be able to train the model truly end to end. This approach would be equivalent to finding a better initialization for the model weights more suited to the task (instead of the traditional Xavier (Glorot & Bengio, 2010) one).

Another possibility is to perform domain translations from the original domain with which the model was performed using an autoencoder as part of the design of the model. This can be achieved as follows. Let us denote as $a(x) = f(g(x))$ an autoencoder, with $g(x)$ being its encoder and $f(x)$ its decoder. $g(x)$ creates a compressed yet ideally lossless representation of the input x and $f(x)$ approximates $g^{-1}(x)$.

We can train a classification model $c(x)$ jointly with this autoencoder by using the intermediate features produced by $f(x)$ and feeding them into $c(x)$.

$a(x)$ is optimised to perform well on the domain it was trained on. We could adapt it to other domains by adding adaptation layers before and after the original ones and training them solely on the new domain (freezing the original ones). The adapter would then also adapt the new domain to the old one for the classifier.

We do not have access to the domain index at inference time as part of the constraints of the problem. For multiple new domains it is possible to select the adequate adapter by reconstructing the images with the different autoencoders (+adapters) and choosing the one minimizing the reconstruction error of the input.

This idea is slighty similar to Zhu, Park, Isola & Efros (2017), which uses a Generative Adversarial Neural Network (which needs much more compute power to be trained) to perform this domain translation.

Invertible Network The model architecture could be chosen to be part of a class of models called "invertible networks" (for instance (Ardizzone *et al.*, 2018)). These models' representations can be transformed back to their inputs. Given the knowledge about the distribution of the representations, it could be possible to directly leverage the classifier to generate past domain samples (without having to train a separate generative model modeling the past domain) and to integrate them into the training to avoid deteriorating performances on past domains. Moreover, as this class of networks have small Lipschitz constants, they are supposed to be more robust to slight changes in the input such as the corruptions/ perturbations from our problem.

2. Post-processing

The ideas proposed so far were part of pre-processing choices and had to be implemented before training the chosen model. It is also possible to try to make existing models predictions better without changing their weights, thus operating in a black box setting. Using a Conditional Random Field on the model predictions at the pixel level to spot spatial inconsistencies would be the post-processing equivalent of the Neighborhood loss Yuan & Xu (2021). This could also be done using superpixels.

3. Regularization methods

We need to avoid catastrophic forgetting of the previous domain knowledge. The surrogate of the model performance in deep learning is the loss value. The aim of many regularization based continual learning methods is to model this loss using little information and memory, such as using the current and ideal parameters of the models. It is then possible to penalize deteriorating the past domains performance.

The penalty function of the Elastic Weight Consolidation (Aich, 2021) method is as follows:

$$R(\theta) = KL(l(\cdot;\theta^*) : l(\cdot;\theta)) \tag{A I-1}$$

with the model loss on the past domain D_{i-1}, this is approximated using a second order Taylor's expansion and Fisher's information matrix (Ly, Marsman, Verhagen, Grasman & Wagenmakers, 2017) into a quadratic penalty using only the weights and not the past samples to approximate the KL divergence :

$$R(\theta) = (\theta - \theta^*_{D_{i-1}})^T \cdot F_{D_{i-1}} \cdot (\theta - \theta^*_{D_{i-1}}) \tag{A I-2}$$

This means that if we optimize the model's parameters by Stochastic Gradient Descent,

$$\theta' = \theta - \eta \cdot (\nabla_\theta L_{\text{total}}(x, y, \theta, \theta^*)) \tag{A I-3}$$

then the individual parameter update becomes:

$$\theta'_k = \theta_k - \eta \cdot (\nabla_\theta L_{\text{total}}(x, y, \theta, \theta^*))_k \tag{A I-4}$$

$$L_{\text{total}}(x, y, \theta, \theta^*) = L_{\text{learning}}(x, y) + \lambda \cdot L_{\text{penalty}}(\theta, \theta^*) \tag{A I-5}$$

$$\begin{aligned}
\theta'_k &= \theta_k - \eta \cdot (\nabla_\theta (L_{\text{total}}(x, y, \theta, \theta^*)))_k \\
&= \theta_k - \eta \cdot (\nabla_\theta (L_{\text{learning}}(x, y) + \lambda \cdot L_{\text{penalty}}(\theta, \theta^*))_k \\
&= \theta_k - \eta \cdot (\nabla_\theta (L_{\text{learning}}(x, y))_k + \lambda \cdot \nabla_\theta (L_{\text{penalty}}(\theta, \theta^*))_k) \\
&= \theta_k - \eta \cdot \nabla_\theta L_{\text{learning}}(x, y)_k - \eta \cdot \lambda \cdot \nabla_\theta ((\theta - \theta^A) \cdot F^A \cdot (\theta - \theta^A))_k \\
&= \theta_k - \eta \cdot \nabla_\theta L_{\text{learning}}(x, y)_k - \eta \cdot \lambda \cdot (\theta_k - \theta^A_k) \cdot F^A_k \\
&= \theta_k \cdot (1 - \eta \cdot \lambda \cdot F^A_k) + \theta^A_k \cdot \eta \cdot \lambda \cdot F^A_k - \eta \cdot \nabla_\theta L_{\text{learning}}(x, y)_k
\end{aligned} \tag{A I-6}$$

Moreover, the latter could also be seen as:

$$\begin{aligned}&\theta_k \cdot (1 - \eta \cdot \lambda \cdot F_k^A) + \theta_k^A \cdot \eta \cdot \lambda \cdot F_k^A - \eta \cdot \nabla_\theta L_{\text{learning}}(x, y)_k \\ &= \text{interpolation}(\theta_k, \theta_k^A, \eta \cdot \lambda \cdot F_k^A) - \eta \cdot \nabla_\theta L_{\text{learning}}(x, y)|_{\theta k}\end{aligned} \tag{A I-7}$$

This means that EWC (Aich, 2021) is equivalent to doing an interpolation between the ideal parameters of the previous task and the current ones, with a coefficient proportional to the importance of the previous task parameters, and doing a gradient descent with a gradient evaluated on the current parameters.

We propose evaluating the gradient at the interpolation point for faster convergence:

$$\theta'_k = \text{interpolation}(\theta_k, \theta_k^A, \eta \cdot \lambda \cdot F_k^A) - \eta \cdot \nabla_\theta L_{\text{learning}}(x, y)|_{\theta_{\text{interpol}}\, k} \tag{A I-8}$$

It remains unknown how to choose the coefficient to calibrate the Fisher matrix.

This hyperparameter selection is done through validation on all the sequence datasets for EWC. However, this procedure is unrealistic for the continual learning setting as one doesn't have access to all the sequence datasets in the process of training and deploying a model and it is not straightforward that the hyperparameter will transfer.

We then have a closed form of this hyperparameter based on the likelihood region we want to stay in for the past tasks.

The likelihood region for a parameter vector θ and optimal parameter vector θ^* is:

$$\theta \quad \text{t.q.} \quad \frac{\mathcal{L}(\theta \mid x)}{\mathcal{L}(\hat{\theta} \mid x)} = V(\theta, \theta^*) \geq r \tag{A I-9}$$

It can be proven Daxberger *et al.* (2021) that

$$p(\theta|\mathcal{X}) = \frac{1}{Z} \cdot \exp(-\mathcal{L}(X;\theta)) \tag{A I-10}$$

with $\mathcal{L}(X;\theta)$ the training loss on the previous domain for parameters θ.

We then have

$$\begin{aligned} V(\theta,\theta^*) &= \frac{p(\theta|\mathcal{X})}{p(\theta^*|\mathcal{X})} \\ &\text{if we use a second order Taylor expension of the loss} \\ &= \exp(-\frac{1}{2}(\theta - \theta^*_{D_{i-1}})^T \cdot \nabla^2_\theta \mathcal{L}(X;\theta) \cdot (\theta - \theta^*_{D_{i-1}})) \\ &= \exp(-\frac{1}{2}(\theta - \theta^*_{D_{i-1}})^T \cdot F_{D_{i-1}} \cdot (\theta - \theta^*_{D_{i-1}})) \end{aligned} \tag{A I-11}$$

We used the fact that the Hessian of the log likelihood for the optimal parameters is the Fisher information matrix.

We need to ensure that the new parameters stay in that likelihood region. This can be done through projection via an interpolation proportional to the Fisher matrix coefficients (as seen before).

For an interpolation between two points θ_k and θ'_k, we have

$$\theta_{interpol} = \theta_k(1 - \lambda \cdot F_k) + \theta'_k(\lambda \cdot F_k) \tag{A I-12}$$

It can be shown that to stay within a likelihood region with likelihood ratio greater or equal to r, then $lambda$ must satisfy this condition:

$$\lambda \geq \sqrt{\frac{\sum_k(\theta_k - \theta^*_k)^2 \cdot F_k + 2 \cdot log(r)}{N \cdot \sum_k(\theta_k - \theta^*_k)^2 \cdot F^3_k}} \tag{A I-13}$$

We can then project the current parameters in the likelihood region at each iteration.

It also could be possible to enforce this constraint with a log-barrier function added to the training loss:

$$R(\theta, \theta^*, F) = -log(-2 \cdot log(r) - \sum_k (\theta_k - \theta_k^*)^2 \cdot F_k) = -log(b - a(\theta, \theta^*)) \qquad \text{(A I-14)}$$

This function takes very large values when $a(\theta, \theta^*)$ gets too close to b, which means that the parameters getting out of the likelihood region would substantially penalize the loss function.

As this is a second order approximation, it is supposed to be only valid for θ' in the neighborhood of θ. It could be possible to consolidate this approximation by using the value of the Fisher Matrix for neighborhood points. This could be done effortlessly for the trajectory points of the parameters during the learning process (in an online manner), to avoid having to compute this Fisher matrix post-training. The Fisher Matrix coefficients could then be an Exponential Moving Average of the ones in the training process with a coefficient giving more importance to parameters points at the end of the learning process.

We could also construct a worst-case neighborhood approximation by taking the max of the Fisher Matrix coefficients (by component during the whole trajectory) instead of the Exponential Moving Average.

Moreover, the Fischer Matrix is approximated to be diagonal while it is far from always being the case, rotating the model weights can help (Liu *et al.*, 2018). It could also improve performance to create a better approximation of the Fisher matrix, for instance by taking into account the intra layer correlations between parameters, resulting in a block diagonal Fisher Matrix.

A modified version of Adam as algorithm is proposed to clarify our approach (Algo. 2). $regularize(\theta_{t-1}, \theta^*, F)$ corresponds to the projection part exposed previously.

Algorithm 2 *Our algorithm, modified from Adam*

```
Require: $\alpha$: Stepsize
Require: $\beta_1, \beta_2 \in (0, 1], \lambda \in (0, 1)$ : Exponential decay rates for the moment estimates
Require: $(1 - \beta_1)^2/\sqrt{1 - \beta_2} < 1$: Constrain from the convergence analysis
Require: $f(\theta)$: Stochastic objective function with parameters $\theta$
Require: $\theta_0$: Initial parameter vector
Require: $C$: Total number of datasets
  $c \leftarrow 0$ (Initialize dataset counter)
  $F \leftarrow 0$ (Empty Fisher matrix diagonal)
  $\theta^* \leftarrow 0$ (Initialize optimal parameters vector to 0)
  while $c \neq C$ do
    $m_0 \leftarrow 0$ (Initialize initial first moment vector)
    $v_0 \leftarrow 0$ (Initialize initial second moment vector)
    $t \leftarrow 0$ (Initialize timestep)
    while $\theta^t$ not converged do
      $t \leftarrow t + 1$
      $\beta_{1,t} \leftarrow 1 - (1 - \beta_1)\lambda^{t-1}$ (Decay the first moment running average coefficient)
      $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$ (Get gradients w.r.t. stochastic objective at timestep $t$)
      $m_t \leftarrow \beta_{1,t} \cdot g_t + (1 - \beta_{1,t}) \cdot m_{t-1}$ (Update biased first moment estimate)
      $v_t \leftarrow \beta_2 \cdot g_t^2 + (1 - \beta_2) \cdot v_{t-1}$ (Update biased second raw moment estimate)
      $\widehat{m}_t \leftarrow m_t/(1 - (1 - \beta_1)^t)$ (Compute bias-corrected first moment estimate)
      $\widehat{v}_t \leftarrow v_t/(1 - (1 - \beta_2)^t)$ (Compute bias-corrected second raw moment estimate)
      if $c \geq 1$ then
        $\theta_{t-1} \leftarrow regularize(\theta_{t-1}, \theta^*, F)$ (Regularize parameters)
      end if
      $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \widehat{m}_t/(\sqrt{\widehat{v}_t} + \epsilon)$ (Update parameters)
    end while
    $F \leftarrow c \cdot F + \widehat{v}_t$ (Update Fisher matrix for dataset c)
    $c \leftarrow c + 1$ (dataset counter)
```

BIBLIOGRAPHY

Ahn, H., Kwak, J., Lim, S., Bang, H., Kim, H. & Moon, T. (2021). Ss-il: Separated softmax for incremental learning. *Proceedings of the IEEE/CVF International conference on computer vision*, pp. 844–853.

Aich, A. (2021). Elastic Weight Consolidation (EWC): Nuts and Bolts. arXiv. Retrieved from: https://arxiv.org/abs/2105.04093.

Alamro, L. & Yusoff, N. (2017). Copy-move forgery detection using integrated DWT and SURF. *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, 9, 67-71.

Aljundi, R., Lin, M., Goujaud, B. & Bengio, Y. (2019). Gradient based sample selection for online continual learning. *Advances in neural information processing systems*, 32.

Ardizzone, L., Kruse, J., Wirkert, S., Rahner, D., Pellegrini, E. W., Klessen, R. S., Maier-Hein, L., Rother, C. & Köthe, U. (2018). Analyzing inverse problems with invertible neural networks. *arXiv preprint arXiv:1808.04730*.

Bahng, H., Jahanian, A., Sankaranarayanan, S. & Isola, P. (2022). Exploring visual prompts for adapting large-scale models. *arXiv preprint arXiv:2203.17274*.

Bammey, Q., Gioi, R. G. v. & Morel, J.-M. (2020). An adaptive neural network for unsupervised mosaic consistency analysis in image forensics. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14194–14204.

Bang, J., Kim, H., Yoo, Y., Ha, J.-W. & Choi, J. (2021). Rainbow memory: Continual learning with a memory of diverse samples. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8218–8227.

Bayar, B. & Stamm, M. C. (2018). Constrained Convolutional Neural Networks: A New Approach Towards General Purpose Image Manipulation Detection. *IEEE Transactions on Information Forensics and Security*, 13(11), 2691-2706. doi: 10.1109/TIFS.2018.2825953.

Bo, X., Junwen, W., Guangjie, L. & Yuewei, D. (2010). Image copy-move forgery detection based on SURF. *2010 International Conference on Multimedia Information Networking and Security*, pp. 889–892.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A. et al. (2020). Language models are few-shot learners. *Advances in neural information processing systems*, 33, 1877–1901.

Buzzega, P., Boschini, M., Porrello, A., Abati, D. & Calderara, S. (2020). Dark experience for general continual learning: a strong, simple baseline. *NeurIPS*.

Cevikalp, H. & Polikar, R. (2008). Local classifier weighting by quadratic programming. *IEEE Transactions on Neural Networks*, 19(10), 1832–1838.

Cha, H., Lee, J. & Shin, J. (2021). Co2l: Contrastive continual learning. *Proceedings of the IEEE/CVF International conference on computer vision*, pp. 9516–9525.

Chaudhry, A., Dokania, P. K., Ajanthan, T. & Torr, P. H. (2018a). Riemannian walk for incremental learning: Understanding forgetting and intransigence. *Proceedings of the European conference on computer vision (ECCV)*, pp. 532–547.

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

Chaudhry, A., Rohrbach, M., Elhoseiny, M., Ajanthan, T., Dokania, P. K., Torr, P. H. & Ranzato, M. (2019). On tiny episodic memories in continual learning. *arXiv preprint arXiv:1902.10486*.

Chen, A., Yao, Y., Chen, P.-Y., Zhang, Y. & Liu, S. (2023). Understanding and improving visual prompting: A label-mapping perspective. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19133–19143.

Chen, C., Li, J., Han, X., Liu, X. & Yu, Y. (2022). Compound domain generalization via meta-knowledge encoding. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7119–7129.

Chen, T., Kornblith, S., Norouzi, M. & Hinton, G. (2020). A simple framework for contrastive learning of visual representations. *International conference on machine learning*, pp. 1597–1607.

Chollet, F. (2017). Xception: Deep learning with depthwise separable convolutions. *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1251–1258.

Comaniciu, D. & Meer, P. (2002). Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on pattern analysis and machine intelligence*, 24(5), 603–619.

Cruz, R. M., Sabourin, R. & Cavalcanti, G. D. (2018). Dynamic classifier selection: Recent advances and perspectives. *Information Fusion*, 41, 195-216. doi: https://doi.org/10.1016/j.inffus.2017.09.010.

Dang, H., Liu, F., Stehouwer, J., Liu, X. & Jain, A. K. (2020). On the detection of digital face manipulation. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern recognition*, pp. 5781–5790.

Daxberger, E., Kristiadi, A., Immer, A., Eschenhagen, R., Bauer, M. & Hennig, P. (2021). Laplace Redux – Effortless Bayesian Deep Learning. arXiv. Retrieved from: https://arxiv.org/abs/2106.14806.

Dixit, R. & Naskar, R. (2017). Review, Analysis and Parameterization of Techniques for Copy-Move Forgery Detection in Digital Images. *IET Image Processing*, 11. doi: 10.1049/iet-ipr.2016.0322.

Dolz, J., Gopinath, K., Yuan, J., Lombaert, H., Desrosiers, C. & Ayed, I. B. (2018). HyperDense-Net: a hyper-densely connected CNN for multi-modal image segmentation. *IEEE transactions on medical imaging*, 38(5), 1116–1126.

Dong, C., Chen, X., Hu, R., Cao, J. & Li, X. (2022). Mvss-net: Multi-view multi-scale supervised networks for image manipulation detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(3), 3539–3553.

Dong, J., Wang, W. & Tan, T. (2013). Casia image tampering detection evaluation database. *2013 IEEE China summit and international conference on signal and information processing*, pp. 422–426.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S. et al. (2021). An image is worth 16x16 words: Transformers for image recognition at scale. *International Conference on Learning Representations*.

Douillard, A., Ramé, A., Couairon, G. & Cord, M. (2022). Dytox: Transformers for continual learning with dynamic token expansion. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9285–9295.

Fini, E., Da Costa, V. G. T., Alameda-Pineda, X., Ricci, E., Alahari, K. & Mairal, J. (2022). Self-supervised models are continual learners. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9621–9630.

Gloe, T. & Böhme, R. (2010). The'Dresden Image Database'for benchmarking digital image forensics. *Proceedings of the 2010 ACM symposium on applied computing*, pp. 1584–1590.

Glorot, X. & Bengio, Y. (2010, 13–15 May). Understanding the difficulty of training deep feedforward neural networks. *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 9(Proceedings of Machine Learning Research), 249–256. Retrieved from: https://proceedings.mlr.press/v9/glorot10a.html.

Grill, J.-B., Strub, F., Altché, F., Tallec, C., Richemond, P., Buchatskaya, E., Doersch, C., Avila Pires, B., Guo, Z., Gheshlaghi Azar, M. et al. (2020). Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33, 21271–21284.

Hou, S., Pan, X., Loy, C. C., Wang, Z. & Lin, D. (2018). Lifelong learning via progressive distillation and retrospection. *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 437–452.

Hou, S., Pan, X., Loy, C. C., Wang, Z. & Lin, D. (2019). Learning a unified classifier incrementally via rebalancing. *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pp. 831–839.

Huang, Q., Dong, X., Chen, D., Zhang, W., Wang, F., Hua, G. & Yu, N. (2023). Diversity-Aware Meta Visual Prompting. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10878–10887.

Islam, A., Long, C., Basharat, A. & Hoogs, A. (2020). Doa-gan: Dual-order attentive generative adversarial network for image copy-move forgery detection and localization. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4676–4685.

Jia, M., Tang, L., Chen, B.-C., Cardie, C., Belongie, S., Hariharan, B. & Lim, S.-N. (2022). Visual prompt tuning. *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXIII*, pp. 709–727.

Jiménez, D. (1998). Dynamically weighted ensemble neural networks for classification. *1998 IEEE International Joint Conference on Neural Networks Proceedings. IEEE World Congress on Computational Intelligence (Cat. No. 98CH36227)*, 1, 753–756.

Ju, C., Han, T., Zheng, K., Zhang, Y. & Xie, W. (2022). Prompting visual-language models for efficient video understanding. *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXV*, pp. 105–124.

Kaur, H. & Jindal, N. (2020). Image and video forensics: A critical survey. *Wireless Personal Communications*, 112, 1281–1302.

Khosla, P., Teterwak, P., Wang, C., Sarna, A., Tian, Y., Isola, P., Maschinot, A., Liu, C. & Krishnan, D. (2020). Supervised contrastive learning. *Advances in neural information processing systems*, 33, 18661–18673.

Kim, M., Tariq, S. & Woo, S. S. (2021). Cored: Generalizing fake media detection with continual representation using distillation. *Proceedings of the 29th ACM International Conference on Multimedia*, pp. 337–346.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A. et al. (2017). Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13), 3521–3526.

Kumar, N. & Meenpal, T. (2020). Semantic Segmentation-Based Image Inpainting Detection (pp. 665-677). doi: 10.1007/978-981-15-4692-1_51.

Kwon, M.-J., Yu, I.-J., Nam, S.-H. & Lee, H.-K. (2021). CAT-Net: Compression artifact tracing network for detection and localization of image splicing. *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 375–384.

Le Scao, T. & Rush, A. M. (2021). How many data points is a prompt worth? *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 2627–2636.

Lee, K., Lee, K., Shin, J. & Lee, H. (2019). Overcoming catastrophic forgetting with unlabeled data in the wild. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 312–321.

Lee, Y., Chen, A. S., Tajwar, F., Kumar, A., Yao, H., Liang, P. & Finn, C. (2022). Surgical fine-tuning improves adaptation to distribution shifts. *arXiv preprint arXiv:2210.11466*.

Li, C., Huang, Z., Paudel, D. P., Wang, Y., Shahbazi, M., Hong, X. & Van Gool, L. (2023). A continual deepfake detection benchmark: Dataset, methods, and essentials. *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 1339–1349.

Li, J., Xie, H., Li, J., Wang, Z. & Zhang, Y. (2021). Frequency-aware discriminative feature learning supervised by single-center loss for face forgery detection. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6458–6467.

Li, L., Bao, J., Zhang, T., Yang, H., Chen, D., Wen, F. & Guo, B. (2020). Face x-ray for more general face forgery detection. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 5001–5010.

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

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P. & Zitnick, C. L. (2014). Microsoft coco: Common objects in context. *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pp. 740–755.

Liu, C., Wang, L., Lyu, L., Sun, C., Wang, X. & Zhu, Q. (2023). Deja Vu: Continual Model Generalization for Unseen Domains. *International Conference on Learning Representations*.

Liu, H., Li, X., Zhou, W., Chen, Y., He, Y., Xue, H., Zhang, W. & Yu, N. (2021). Spatial-phase shallow learning: rethinking face forgery detection in frequency domain. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 772–781.

Liu, X., Masana, M., Herranz, L., Van de Weijer, J., Lopez, A. M. & Bagdanov, A. D. (2018). Rotate your networks: Better weight consolidation and less catastrophic forgetting. *2018 24th International Conference on Pattern Recognition (ICPR)*, pp. 2262–2268.

Liu, X., Liu, Y., Chen, J. & Liu, X. (2022). PSCC-Net: Progressive spatio-channel correlation network for image manipulation detection and localization. *IEEE Transactions on Circuits and Systems for Video Technology*, 32(11), 7505–7517.

Lomonaco, V. & Maltoni, D. (2017). CORe50: a New Dataset and Benchmark for Continuous Object Recognition. *CoRR*, abs/1705.03550. Retrieved from: http://arxiv.org/abs/1705.03550.

Lopez-Paz, D. & Ranzato, M. (2017). Gradient episodic memory for continual learning. *Advances in neural information processing systems*, 30.

Lu, Y., Liu, J., Zhang, Y., Liu, Y. & Tian, X. (2022). Prompt distribution learning. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5206–5215.

Ly, A., Marsman, M., Verhagen, J., Grasman, R. P. & Wagenmakers, E.-J. (2017). A tutorial on Fisher information. *Journal of Mathematical Psychology*, 80, 40–55.

Mahdian, B. & Saic, S. (2007). Detection of copy–move forgery using a method based on blur moment invariants. *Forensic science international*, 171(2-3), 180–189.

Marra, F., Saltori, C., Boato, G. & Verdoliva, L. (2019). Incremental learning for the detection and classification of gan-generated images. *2019 IEEE international workshop on information forensics and security (WIFS)*, pp. 1–6.

Masi, I., Killekar, A., Mascarenhas, R. M., Gurudatt, S. P. & AbdAlmageed, W. (2020). Two-branch recurrent network for isolating deepfakes in videos. *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VII 16*, pp. 667–684.

Meena, K. B. & Tyagi, V. (2019). Image forgery detection: survey and future directions. *Data, Engineering and Applications: Volume 2*, 163–194.

Minaee, S., Boykov, Y., Porikli, F., Plaza, A., Kehtarnavaz, N. & Terzopoulos, D. (2021). Image segmentation using deep learning: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 44(7), 3523–3542.

Novozamsky, A., Mahdian, B. & Saic, S. (2020). IMD2020: A large-scale annotated dataset tailored for detecting manipulated images. *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision Workshops*, pp. 71–80.

Oren, G. & Wolf, L. (2021). In defense of the learning without forgetting for task incremental learning. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2209–2218.

Pellegrini, L., Graffieti, G., Lomonaco, V. & Maltoni, D. (2020a). Latent replay for real-time continual learning. *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 10203–10209.

Pellegrini, L., Graffieti, G., Lomonaco, V. & Maltoni, D. (2020b). Latent replay for real-time continual learning. *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 10203–10209.

Peng, X., Bai, Q., Xia, X., Huang, Z., Saenko, K. & Wang, B. (2019). Moment matching for multi-source domain adaptation. *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 1406–1415.

Prabhu, A., Torr, P. H. & Dokania, P. K. (2020). Gdumb: A simple approach that questions our progress in continual learning. *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pp. 524–540.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G. & Sutskever, I. (2021a, 18–24 Jul). Learning Transferable Visual Models From Natural Language Supervision. *Proceedings of the 38th International Conference on Machine Learning*, 139(Proceedings of Machine Learning Research), 8748–8763. Retrieved from: https://proceedings.mlr.press/v139/radford21a.html.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J. et al. (2021b). Learning transferable visual models from natural language supervision. *International conference on machine learning*, pp. 8748–8763.

Ranzato, M., Huang, F. J., Boureau, Y.-L. & LeCun, Y. (2007). Unsupervised Learning of Invariant Feature Hierarchies with Applications to Object Recognition. *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1-8. doi: 10.1109/CVPR.2007.383157.

Rolnick, D., Ahuja, A., Schwarz, J., Lillicrap, T. & Wayne, G. (2019). Experience replay for continual learning. *Advances in Neural Information Processing Systems*, 32.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M. et al. (2015). Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115, 211–252.

Ryu, S.-J., Lee, M.-J. & Lee, H.-K. (2010). Detection of copy-rotate-move forgery using Zernike moments. *Information Hiding: 12th International Conference, IH 2010, Calgary, AB, Canada, June 28-30, 2010, Revised Selected Papers 12*, pp. 51–65.

Salloum, R., Ren, Y. & Kuo, C.-C. J. (2018). Image splicing localization using a multi-task fully convolutional network (MFCN). *Journal of Visual Communication and Image Representation*, 51, 201–209.

Shi, X., Chen, Z., Wang, H., Yeung, D.-Y., Wong, W.-K. & Woo, W.-c. (2015). Convolutional LSTM network: A machine learning approach for precipitation nowcasting. *Advances in neural information processing systems*, 28.

Shin, H., Lee, J. K., Kim, J. & Kim, J. (2017). Continual learning with deep generative replay. *Advances in neural information processing systems*, 30.

Snell, J., Swersky, K. & Zemel, R. (2017). Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30.

Sohn, K., Chang, H., Lezama, J., Polania, L., Zhang, H., Hao, Y., Essa, I. & Jiang, L. (2023). Visual prompt tuning for generative transfer learning. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19840–19851.

Song, J., Lee, J., Kweon, I. S. & Choi, S. (2023). EcoTTA: Memory-Efficient Continual Test-time Adaptation via Self-distilled Regularization. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11920–11929.

Štefka, D. & Holeňa, M. (2015). Dynamic classifier aggregation using interaction-sensitive fuzzy measures. *Fuzzy Sets and Systems*, 270, 25–52.

Sun, W., Zhou, J., Li, Y., Cheung, M. & She, J. (2020). Robust high-capacity watermarking over online social network shared images. *IEEE Transactions on Circuits and Systems for Video Technology*, 31(3), 1208–1221.

Tsymbal, A., Pechenizkiy, M., Cunningham, P. & Puuronen, S. (2008). Dynamic integration of classifiers for handling concept drift. *Information fusion*, 9(1), 56–68.

van de Ven, G. M., Tuytelaars, T. & Tolias, A. S. (2022). Three types of incremental learning. *Nature Machine Intelligence*, 4(12), 1185–1197.

Wang, Q., Fink, O., Van Gool, L. & Dai, D. (2022a). Continual test-time domain adaptation. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7201–7211.

Wang, Y., Huang, Z. & Hong, X. S-Prompts Learning with Pre-trained Transformers: An Occam's Razor for Domain Incremental Learning (Supplementary Material).

Wang, Y., Huang, Z. & Hong, X. (2022b). S-Prompts Learning with Pre-trained Transformers: An Occam's Razor for Domain Incremental Learning. *Advances in Neural Information Processing Systems*. Retrieved from: https://openreview.net/forum?id=ZVe_WeMold.

Wang, Z., Zhang, Z., Ebrahimi, S., Sun, R., Zhang, H., Lee, C.-Y., Ren, X., Su, G., Perot, V., Dy, J. et al. (2022c). Dualprompt: Complementary prompting for rehearsal-free continual learning. *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXVI*, pp. 631–648.

Wang, Z., Zhang, Z., Lee, C.-Y., Zhang, H., Sun, R., Ren, X., Su, G., Perot, V., Dy, J. & Pfister, T. (2022d). Learning to prompt for continual learning. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 139–149.

Wang, Z., Luo, Y., Qiu, R., Huang, Z. & Baktashmotlagh, M. (2021). Learning to diversify for single domain generalization. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 834–843.

Woo, S., Park, J., Lee, J.-Y. & Kweon, I. S. (2018). Cbam: Convolutional block attention module. *Proceedings of the European conference on computer vision (ECCV)*, pp. 3–19.

Wu, H., Zhou, J., Tian, J. & Liu, J. (2022). Robust image forgery detection over online social network shared images. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13440–13449.

Wu, Y., Abd-Almageed, W. & Natarajan, P. (2018). Busternet: Detecting copy-move image forgery with source/target localization. *Proceedings of the European conference on computer vision (ECCV)*, pp. 168–184.

Wu, Y., AbdAlmageed, W. & Natarajan, P. (2019a). Mantra-net: Manipulation tracing network for detection and localization of image forgeries with anomalous features. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9543–9552.

Wu, Y., Chen, Y., Wang, L., Ye, Y., Liu, Z., Guo, Y. & Fu, Y. (2019b). Large scale incremental learning. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 374–382.

Xie, J., Yan, S. & He, X. (2022). General incremental learning with domain-aware categorical representations. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14351–14360.

Xing, Y., Wu, Q., Cheng, D., Zhang, S., Liang, G. & Zhang, Y. (2022). Class-aware visual prompt tuning for vision-language pre-trained model. *arXiv preprint arXiv:2208.08340*.

Yang, B., Sun, X., Guo, H., Xia, Z. & Chen, X. (2018). A copy-move forgery detection method based on CMFD-SIFT. *Multimedia Tools and Applications*, 77, 837–855.

Yang, S., Wang, Y., Van De Weijer, J., Herranz, L. & Jui, S. (2021). Generalized source-free domain adaptation. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8978–8987.

Yao, X., Bai, Y., Zhang, X., Zhang, Y., Sun, Q., Chen, R., Li, R. & Yu, B. (2022). PCL: Proxy-based Contrastive Learning for Domain Generalization. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7097–7107.

Yuan, W. & Xu, W. (2021). NeighborLoss: a loss function considering spatial correlation for semantic segmentation of remote sensing image. *IEEE Access*, 9, 75641–75649.

Zbontar, J., Jing, L., Misra, I., LeCun, Y. & Deny, S. (2021). Barlow twins: Self-supervised learning via redundancy reduction. *International Conference on Machine Learning*, pp. 12310–12320.

Zenke, F., Poole, B. & Ganguli, S. (2017). Continual learning through synaptic intelligence. *International conference on machine learning*, pp. 3987–3995.

Zhang, X., He, Y., Xu, R., Yu, H., Shen, Z. & Cui, P. (2023). Nico++: Towards better benchmarking for domain generalization. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16036–16047.

Zhang, Y., Li, M., Li, R., Jia, K. & Zhang, L. (2022). Exact feature distribution matching for arbitrary style transfer and domain generalization. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8035–8045.

Zhou, K., Yang, J., Loy, C. C. & Liu, Z. (2022a). Conditional prompt learning for vision-language models. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16816–16825.

Zhou, K., Yang, J., Loy, C. C. & Liu, Z. (2022b). Learning to prompt for vision-language models. *International Journal of Computer Vision*, 130(9), 2337–2348.

Zhou, P., Chen, B.-C., Han, X., Najibi, M., Shrivastava, A., Lim, S.-N. & Davis, L. (2020). Generate, segment, and refine: Towards generic manipulation segmentation. *Proceedings of the AAAI conference on artificial intelligence*, 34(07), 13058–13065.

Zhu, J.-Y., Park, T., Isola, P. & Efros, A. A. (2017). Unpaired image-to-image translation using cycle-consistent adversarial networks. *Proceedings of the IEEE international conference on computer vision*, pp. 2223–2232.