AI4media

ARTIFICIAL INTELLIGENCE FOR THE MEDIA AND SOCIETY

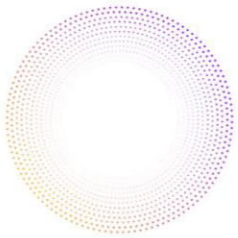
D3.1

Initial Outcomes of New Learning Paradigms Research

Project Title	AI4Media - A European Excellence Centre for Media, Society and Democracy
Contract No.	951911
Instrument	Research and Innovation Action
Thematic Priority	H2020-EU.2.1.1. - INDUSTRIAL LEADERSHIP - Leadership in enabling and industrial technologies - Information and Communication Technologies (ICT) / ICT-48-2020 - Towards a vibrant European network of AI excellence centres
Start of Project	1 September 2020
Duration	48 months

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 951911

info@ai4media.eu www.ai4media.eu



Deliverable title	Initial Outcomes of New Learning Paradigms Research
Deliverable number	D3.1
Deliverable version	1.0
Previous version(s)	N/A
Contractual date of delivery	August 31, 2021
Actual date of delivery	August 23, 2021
Deliverable filename	AI4Media_D3_1-final.pdf
Nature of deliverable	Report
Dissemination level	Public
Number of pages	64
Work Package	WP3
Task(s)	T3.1, T3.3, T3.7
Parner responsible	UNITN
Author(s)	Giuseppe Amato, Luca Ciampi, Claudio Gennaro, Alejandro Moreo, Andrea Pedrotti, Fabrizio Sebastiani (CNR), Cigdem Beyan, Elisa Ricci, Nicu Sebe, Kasim Sinan Yildirim (UNITN), Matteo Bruni (UNIFI), Dario Garcia-Gasulla (BSC), Federico Pernici (UNIFI), Grégoire Petit, Adrian Popescu (CEA), Ioannis Mademlis (AUTH)
Editor	Cigdem Beyan (UNITN), Nicu Sebe (UNITN)
Officer	Evangelia Markidou

Abstract	This document presents the initial outcomes of the research on new learning paradigms in WP3 of AI4Media. As such, the document summarizes the research advances of the contributing partners in tasks T3.1, T3.3, and T3.7 at month 12. For each task we present the contributions including the relevant publications and the link to the software (if available). Finally, we discuss the plans for ongoing activities and the future research.

Keywords	lifelong learning, online learning, transfer learning and domain adaptation, learning to count

Copyright

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 951911

Contributors

NAME	ORGANIZATION
Giuseppe Amato	CNR
Cigdem Beyan	UniTN
Matteo Bruni	UNIFI
Luca Ciampi	CNR
Dario Garcia-Gasulla	BSC
Claudio Gennaro	CNR
Ioannis Mademlis	AUTH
Alejandro Moreo	CNR
Andrea Pedrotti	CNR
Federico Pernici	UNIFI
Grégoire Petit	CEA
Adrian Popescu	CEA
Elisa Ricci	UniTN
Fabrizio Sebastiani	CNR
Nicu Sebe	UniTN
Kasim Sinan Yildirim	UniTN

Peer Reviews

NAME	ORGANIZATION
Ioannis (Yiannis) Patras	QMUL
Lorenzo Seidenari	UNIFI

Revision History

Version	Date	Reviewer	Modifications
0.1	03/06/2021	Nicu Sebe, Cigdem Beyan	First draft sent to partners for contributions
0.2	30/06/2021	Nicu Sebe, Cigdem Beyan	Updated version including inputs from UNITN, CEA, AUTH, BSC, CNR and UNIFI in sections 3, 4, and 5.
0.3	08/07/2021	Ioannis (Yiannis) Patras	Updated version including inputs from UNITN, CEA, AUTH, BSC, CNR and UNIFI in sections 3, 4, and 5, and inputs from UNITN for executive summary, introduction and conclusions.
0.4	10/07/2021	Lorenzo Seidenari	Updated version including inputs from UNITN, CEA, AUTH, BSC, CNR and UNIFI in sections 3, 4, and 5, and inputs from UNITN for executive summary, introduction and conclusions.
0.5	12/07/2021	Filareti Tsalakanidou	Updated version including inputs from UNITN, CEA, AUTH, BSC, CNR and UNIFI in sections 3, 4, and 5, and inputs from UNITN for executive summary, introduction and conclusions.
0.6	26/07/2021	Filareti Tsalakanidou	Updated version based on internal review comments.
1.0	27/07/2021	Nicu Sebe	Final version

Table of Abbreviations and Acronyms

Abbreviation	Meaning
ACC	Accuracy
BN	Batch Normalization Layer
CA	Convolutional Autoencoder
CC	Classify and Count
CIL	Class-Incremental Learning
CLC	cross-lingual text classification
CMC	cumulative matching characteristic
CNN	Convolutional Neural Network
DCL	Domain-aware Curriculum Learning
DG	Domain Generalization
DNN	Deep Neural Network
EHDS	Easy-To-Hard Domain Selection
FC	fully-connected layer
Fun	Funnelling
GAN	Generative Adversarial Network
GCN	Graph Convolutional Networks
GFun	Generalized Funnelling
HTL	Heterogeneous transfer learning
I2I	Image-to-Image Translator
ID	in-distribution
mAP	mean average precision
MetaBN	meta batch normalization layer
MTDA	Multi-target Domain Adaptation
NCD	Novel Class Discovery
NCL	Neighborhood Contrastive Loss
NMI	Normalized Mutual Information
OOD	Out-of-Distribution
ReID	re-identification
RePoNet	Regular Polytope Networks
SGD	Stochastic Gradient Descent
STDA	Single Target Domain
TL	transfer learning
UDA	Unsupervised Domain Adaptation
VGFs	view-generating functions
WCEs	Word-Class Embeddings

Contents

List of Tables

List of Figures

1. Executive Summary

This deliverable provides the research results at M12 of the activities in Task 3.1 (Lifelong and on-line learning), Task 3.3 (Transfer learning) and Task 3.7 (Learning to count). We present in detail the motivation, the developed methods, the obtained results making when available explicit references to the publications, software by the partners and the contributions to the WP8 use cases.

The presented work reflects the very good stage of the work done in WP3 so far. All the participants are very active and their results have been successfully published in top venues of the community. We also present a detailed description of the ongoing research with very positive perspectives for the future developments. In detail:

- The lifelong and on-line learning (Task 3.1) contributions presented in this deliverable include theoretical and experimental aspects and particularly focus on: (a) novel techniques for novel class discovery, (b) a new method for class-incremental learning, (c) investigation of dynamic adaptation framework for DNNs and (d) a new approach focusing on compatible feature learning. These are related to WP8 use cases 2A, 2B, 3A3, 4C1 and 7A3 providing support for archive exploration, image and video analysis, and organization of image and video collections. The ongoing work for Task 3.1 considers (i) memory-constrained online continual learning and (ii) class-incremental learning without memory of past data. Both approaches try to reduce the effects of catastrophic forgetting and to ensure a better balance between stability and plasticity by mixing fixed and adaptive feature representations.

- The transfer learning (Task 3.3) contributions presented in this deliverable include: (a) a novel approach for multi-target domain adaptation, (b) a novel method for multi-source domain generalization, and (c) a novel method for heterogeneous document embeddings for cross-lingual text classification. The developed approaches are relevant to the WP8 use cases 3A3, 3C2, 4C1, 4C5, and 7A3 providing support for archive exploration, content adaptation, image, video, and multimodal analysis, and organization of image and video collections. The ongoing and future work on Task 3.3 include: (i) unsupervised domain adaptation on videos, (ii) presenting a stable deep model pre-training pipeline to obtaining models which are transferable towards a large number of target tasks, and (iii) extending existing heterogeneous document embeddings by integrating view-generating function and vector averaging for cross-lingual text classification.

- Regarding learning to count and quantify (Task 3.7), this deliverable includes: (a) a novel approach developed for re-assessing quantification methods with quantification-oriented parameter optimisation and (b) a new method for counting objects by leveraging domain adaptation techniques. The tools developed in this task contributes to use cases 2B1, 3C2-6, 3C2-7, 4C2, by providing solutions to analyze visual content and to count objects contained in it. Among ongoing work of Task 3.7 some can be listed as: (i) an application of learning to quantify to monitor and mitigate the bias of classifiers and (ii) an application of learning to quantify to estimate recall in technology-assisted review.

The goal in the near future is to foster joint research that can result in developments that are mutually beneficial to the cooperating partners.

2. Introduction

In an ideal world, there should be abundant labeled training samples, which have the same distribution as the test data. However, in practice, collecting and annotating sufficient training data is often expensive, time-consuming, and it can be even unrealistic in some scenarios. *Transfer learning* aims at reusing and exploiting models, originally trained in a given task, to solve a second and potentially unrelated problem. Therefore, it transfers the knowledge across domains and promises to solve the aforementioned problems. According to the discrepancy between domains, transfer learning can be further divided into two categories, i.e., homogeneous and heterogeneous transfer learning [2]. Homogeneous transfer learning approaches are regarding situations where the domains are of the same feature space. Whereas heterogeneous transfer learning refers to the knowledge transfer process in the situations where the domains have different feature spaces. In addition to distribution adaptation, heterogeneous transfer learning typically requires feature space adaptation, which makes it more complicated than homogeneous transfer learning. The contributions of the AI4Media project regarding transfer learning methodologies are given in Section 4. These include investigating the impact of size of the data for transfer learning, and examining at which degree it is beneficial to use pre-trained representations and presenting novel methodologies for multi-target domain adaptation and multi-source domain generalization. In this context, in total three papers were accepted to be presented in peer-reviewed conferences and there is one ongoing work in preparation to be submitted.

Transfer learning is related to *Lifelong Learning* which stands for the ability to continuously acquiring, fine-tuning and transferring knowledge and skills [3]. Lifelong learning algorithms allow AI systems to update their capabilities in order to understand novelty. The *Lifelong Learning* [3] contribution of the AI4media partners are related to the practical cases in which: (1) access to past data is limited or impossible, (2) computation needs should be as close to constant as possible and (3) learning of new data eeds to be fast. Therefore, novel online learning strategies have been investigated in the context of lifelong learning. The current contributions of the AI4Media project regarding lifelong and online learning methodologies are given in Section 3. These include (a) two methods tackling novel class discovery, which is an open set problem where classes of unlabeled data are not predefined; (b) a dynamic adaptation framework for neural networks whose objective is to unify out-of-distribution detection, lifelong learning and neural distillation. Regarding out-of-distribution detection a survey of existing methods is presented. Moreover, (c) a novel feature alignment method is introducing novel ways to learn internal feature representations which are compatible with previously learned ones in lifelong learning. Thus, the feature comparability throughout the whole learning process is provided. In total two papers were accepted to be presented in peer-reviewed conferences, two papers were accepted to peer-reviewed journals and there is one submitted paper in a peer-reviewed conference.

The ideal world mentioned above is also free from dataset shift, i.e., the joint distribution of the independent and the dependent variables is not the same in the training data and in the unlabelled data for which predictions must be issued. When this occurs, estimating the prevalence of the classes of interest in the unlabelled data is difficult, since "traditional" learning methods (i.e., ones based on the IID assumption) assume these prevalence values to stay approximately constant. *Learning to Count* is concerned with developing techniques for estimating quantities in unlabelled data possibly affected by dataset shift, where these quantities can be the prevalence values (i.e., relative frequencies) of the classes of interest (as needed in applications such as, e.g., monitoring consensus for a certain policy or political candidate in social media) or the number of physical objects in instances of visual media (such as estimating car park occupancy from surveillance camera images, or monitoring traffic volumes from road cameras). The contributions of the AI4Media project regarding lifelong and online learning methodologies are given in Section 5.

All the methods presented in this deliverable can be applied to media-related areas and applications. Indeed after describing each method, we also present their relations to WP8 Use Cases. Finally, Section 6 concludes the deliverable by summarizing the work covered as well as presenting the ongoing work regarding each task addressed in this deliverable.

3. Lifelong and on-line learning (Task 3.1)

Lifelong and on-line learning are two open research topics in artificial intelligence whose overall objective is to optimize model training from dynamic data. Handling such data is important insofar as data are not all available from the beginning and trained models should augment their capabilities in order to accommodate new data as they arrive. This task is not straightforward since standard deep learning models are not designed to deal with non-static data. Advances in these two closely related fields are potentially important for a large number of applications which deal with dynamic data. This is particularly the case for media, where new concepts appear constantly, e.g., in news and should be learned in order to ensure their appropriate processing.

3.1. Overview of our lifelong and on-line learning contributions (Task 3.1)

Withing this task partners propose works which tackle both theoretical and experimental facets of lifelong and on-line learning. The contributions summarized here and discussed in more details in the next subsections are designed so as to be usable by AI4Media use cases.

In Subsection 3.2, UNITN introduces two methods which tackle novel class discovery, an open set problem where classes of unlabeled data are not predefined. The first method consists in an optimized usage of deep models which are pretrained on known classes during the novel class discovery step. The second method introduces a holistic learning framework which exploits contrastive loss to learn discriminative features from labeled and unlabeled data.

In Subsection 3.3, CEA proposes a comparison of Class-Incremental Learning (CIL) algorithms. These algorithms are analyzed along six dimensions which are deemed important in incremental learning. In addition, a simple but effective method inspired by imbalanced data learning methods is proposed as a strong baseline of the future works.

In Subsection 3.4, AUTH discusses a contribution which is related to deep neural network education. This work will consist of a dynamic adaptation framework for neural models whose objective is to unify out-of-distribution detection, lifelong learning and neural distillation. The first contribution is a survey of out-of-distribution detection, which is the first core component of the education framework.

In Subsection 3.5, UNIFI investigates novel ways to learn internal features representations which are compatible with previously learned ones in lifelong learning. This feature alignment is important insofar as it allows for feature compatibility throughout the whole learning process in a life-long setting. The main innovation is to train representations exploiting pre-allocated fixed classifiers based on regular polytopes in order to enable stationarity of representations without performance loss.

3.2. Novel class discovery (Task 3.1)

Contributing partners: UNITN

Since collecting human annotated data for every single task is both challenging and expensive, the machine learning community has shifted the attention to the techniques that can learn with limited or completely non-annotated data. To this end, many semi-supervised [4, 5] and unsupervised learning [6, 7, 8, 9] methods have been proposed, which achieve promising results compared to supervised learning methods. Nonetheless, not much effort has been made to exploit prior knowledge from existing labeled datasets and use it to discover unknown classes that are not present in the labeled data.

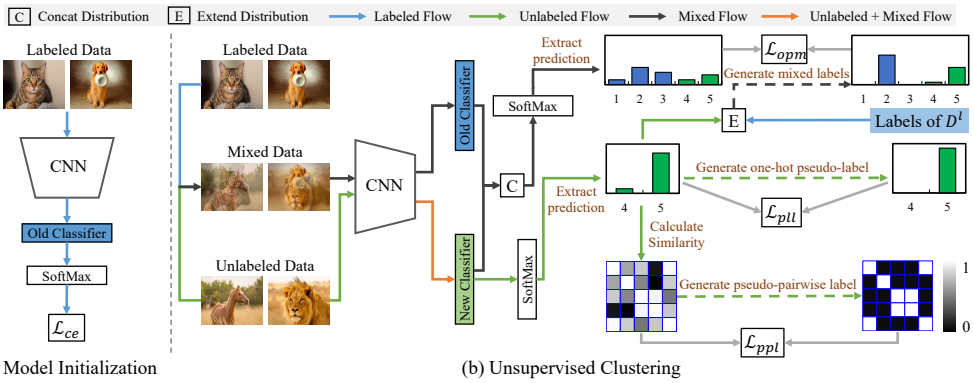


Figure 1. The proposed method: (a) we first initialize the model on the labeled data ($\mathcal{L}_{ce}$). (b) then, we learn the unsupervised clustering model for discovering new classes in unlabeled data, by pseudo-pair learning ($\mathcal{L}_{ppl}$), pseudo-label learning ($\mathcal{L}_{pll}$) and learning with the proposed loss ($\mathcal{L}_{opm}$).

Novel Class Discovery (NCD) is an open set problem where classes of unlabeled data are undefined previously and annotated samples of these novel classes are not available [1, 10, 11, 12]. In this concept, labeled data of known (so-called old) classes and the unlabeled data of novel (so-called new) classes are given. The goal of novel class discovery is to identify new classes in unlabeled data with the support of knowledge of old classes.

3.2.1. Discovering novel visual categories in an open world

Existing methods on NCD [1, 10, 11, 12] typically first pre-train a model with labeled data, and then identify new classes in unlabeled data via unsupervised clustering. However, the labeled data that provide essential knowledge are often under-explored in the second step. Thus, the model can only benefit from the off-the-shell knowledge of the labeled data, but fails to leverage the underlying relationship between the labeled and unlabeled data. The challenge is that the labeled and unlabeled examples are from non-overlapping classes, which makes it difficult to build the learning relationship between them.

We claim that the labeled data provide essential knowledge about underlying object structures and common visual patterns. However, the use of labeled data is much harder than in semi-supervised learning [5, 13] because the labeled and unlabeled samples are from disjoint classes. To this end, we investigate: "effectively exploiting the labeled data to promote the discovery of new classes" and to do so, we propose a simple but effective method, called *OpenMix*.

We are provided with labeled data $D^l = \{X^l, Y^l\}$ and unlabeled data $D^u = \{X^u\}$. The number of samples is N^l in D^l and N^u in D^u, respectively. Each labeled image x_i^l has a label y_i^l, where $y_i^l \in \{1, 2, ..., C^l\}$ and C^l is the number of classes of D^l. Following [1], we assume the number of classes of D^u is prior knowledge, which is defined as C^u. The classes of D^l and D^u are disjoint. We define the classes as old classes and new classes for D^l and D^u, respectively. The goal of novel class discovery is to leverage the knowledge of D^l to identify the classes in D^u.

We try to achieve this goal by learning a model constructed by a Convolutional Neural Network (CNN) and two classifiers. These two classifiers, called old classifier and new classifier, are used to recognize samples from old classes and new classes, respectively. The framework of our method is illustrated in Fig. 1.

We utilize the labeled data to train the CNN and the old classifier, which can provide basic discriminative representations for images and accurately classify samples of old classes. Given the

labeled data $D^l = \{X^l, Y^l\}$, we are able to train the model in a supervised way. Specifically, the model is trained with the cross-entropy loss, as done in the traditional supervised classification [14].

Following this, we learn an unsupervised clustering model on the unlabeled data by *(1) pseudo-pair learning* and *(2) pseudo-label learning*, which allow us to identify samples of new classes. In detail, given the model pre-trained on the labeled data, we additionally add a classifier layer of C^u new classes on the head of CNN. We then focus on unsupervised clustering in unlabeled data by first exploring the relationship between two images for model training. Inspired by DAC [15] and DCCM [16], we first obtain the outputs of the new classifier for input unlabeled samples and compute their cosine similarity matrix. We then estimate the pseudo-pairwise labels by setting a threshold on cosine similarity matrix. By doing so, two images are defined as a positive pair if their cosine similarity is larger than threshold, otherwise they are a negative pair. Given this pairwise supervision, we train the model with a binary cross-entropy loss. For pseudo-label learning, we reformulate the predictions output by the new classifier to one-hot pseudo-labels, which can be used to further improve the model performance. At this stage, we only train the model with the unlabeled samples using cross-entropy loss that are assigned with one-hot pseudo-labels. By jointly considering the pseudo-pair learning and pseudo-label learning, the unsupervised clustering loss is expressed as the sum of pseudo-pair learning and weighted pseudo-label learning.

In the described architecture until here, the labeled data only play the role of model initialization, however, there is no utilization of labeled data in the unsupervised clustering stage. Instead OpenMix effectively leverage the labeled data D^l during the unsupervised clustering in unlabeled data D^u. In a nutshell, during unsupervised clustering, OpenMix additionally compounds examples in two ways:

(a) Mix unlabeled examples with labeled samples: OpenMix mixes the labeled samples with unlabeled samples, as well as their labels with pseudo-labels. Taking the prior knowledge that labeled samples and unlabeled samples belong to completely different classes, we first extend the label distributions of the labeled samples and unlabeled samples to the same size. Specifically, we concatenate $\hat{y}^l$ with a C^u-dim zeros-vector while $\hat{z}^u$ (prediction of new classifier) with a C^l-dim zeros-vector. The extended labels/pseudo-labels are represented by $\bar{y}^l$ for labeled samples and $\bar{y}^u$ for unlabeled samples, respectively. We then generate virtual sample with MixUp [4],

$$\begin{aligned} &\eta \sim \text{Beta}(\epsilon, \epsilon), \quad \eta^* = \text{Max}(\eta, 1-\eta), \\ &m = \eta^* x^l + (1-\eta^*)x^u, \quad v = \eta^* \bar{y}^l + (1-\eta^*)\bar{y}^u, \end{aligned} \tag{1}$$

where ϵ is a hyper-parameter and $\eta \in [0, 1]$. m is the generated sample and v is the pseudo-label of m. The second constraint in Eq. 1 ensures that the generated sample m is closer to x^l than x^u. This can alleviate the negative impact caused by unreliable pseudo-labels of unlabeled samples.

(b) Mix unlabeled examples with reliable anchors: We select the unlabeled samples that have high class-probabilities as reliable anchors. Then, we mix the anchors with unlabeled samples through OpenMix. Specifically, we perform this operation by replacing the labeled sample x^l with a reliable anchor in Eq. 1.

Given mixed samples $\mathcal{M}$ and their pseudo-labels $\mathcal{V}$, we apply the L2-norm loss to train OpenMix, defined as,

$$\mathcal{L}_{opm} = \frac{1}{|\mathcal{M}|} \sum_{i \in \mathcal{M}} \frac{1}{C^l + C^u} \| v_i - \text{SoftMax}(z_i^m) \|_2, \tag{2}$$

where $|\mathcal{M}|$ denotes the number of samples in $\mathcal{M}$. z_i^m indicates the concentrated outputs of the old and new classifiers. Specifically, we forward m_i to the model and extract the outputs of the old and new classifiers, which are represented as z_i^l and z_i^u, respectively. z_i^m is then obtained by

concentrating z_i^l and z_i^u.

Experimental Analysis and Results. We evaluate our method on three image classification benchmarks: CIFAR-10 [17], CIFAR-100 [17] and ImageNet [18]. We employ the clustering ACC and Normalized Mutual Information (NMI) [19] as the metrics to evaluate the clustering performance of new classes. Both metrics range from 0 to 1. Higher scores mean better performance. For CIFAR-10 and CIFAR-100, we show the average results of 10 runs. For ImageNet, results averaged in three different subsets are reported.

Table 1. Comparison with state-of-the-art methods on CIFAR-10, CIFAR-100 and ImageNet for novel class discovery. Note that, RS [1] did not evaluate the NMI metric and did not provide results of 10-class setting on CIFAR-100.

Method	CIFAR-10		CIFAR-100		ImageNet	
	ACC	NMI	ACC	NMI	ACC	NMI
K-means [20]	65.5%	0.422	66.2%	0.555	71.9%	0.713
KCL [11]	66.5%	0.438	27.4%	0.151	73.8%	0.750
MCL [12]	64.2%	0.398	32.7%	0.202	74.4%	0.762
DTC [10]	87.5%	0.735	72.8%	0.634	78.3%	0.791
RS [1]	91.7%	-	-	-	82.5%	-
Ours	**95.3%**	**0.879**	**87.2%**	**0.754**	**85.7%**	**0.827**

In Table 1, we compare the proposed method with the state-of-the-art methods, including: K-means [20], KCL [11], MCL [12], DTC [10] and RS [1]. Our baseline achieves very competitive clustering performance compared with the state-of-the-art. The baseline is higher than DTC [10] on CIFAR-10 and CIFAR-100, and slightly lower than DTC [10] on ImageNet. Moreover, it is clear that our method outperforms the state-of-the-art methods by a large margin. Specifically, our approach achieves 95.3% for CIFAR-10, 87.2% for CIFAR-100 and 85.7% for ImageNet in clustering ACC. Both KCL [11] and MCL [12] use pairwise similarity for clustering learning. However, these two method fail to produce competitive performance. For example, KCL and MCL have similar results to K-means on CIFAR-10 and ImageNet, but are largely inferior to K-means on CIFAR-100. Our method is significantly superior to KCL and MCL. Compared to DTC [10], our method surpasses it in all three datasets, especially in CIFAR-100. RS [1] is the latest method, which also uses the labeled data during unsupervised clustering. However, RS mainly focuses on using labeled data to maintain the ACC in old classes. Compared to RS, our method outperforms it by 3.6% and 3.2% in clustering ACC for CIFAR-10 and ImageNet, respectively.

3.2.2. Contrastive learning for novel class discovery

We also addressed the NCD problem by proposing a holistic learning framework that uses contrastive loss [8, 21] to learn discriminative features from both the labeled and unlabelled data. This framework relies on two key ideas: (1) The local neighborhood of a sample (*query*) in the embedding space will contain samples, which most likely belong to the same semantic category of the query, and can be considered as *pseudo-positives*. (2) Addressing the better selection of *negatives* to further improve the contrastive learning. Peculiar to the NCD task where we are given labeled samples of the known classes (a.k.a *true* negatives of any unlabeled instance), we exploit them, together with the unlabeled samples, to generate synthetic samples in the feature space using a mixing strategy and treat them as *hard* negatives.

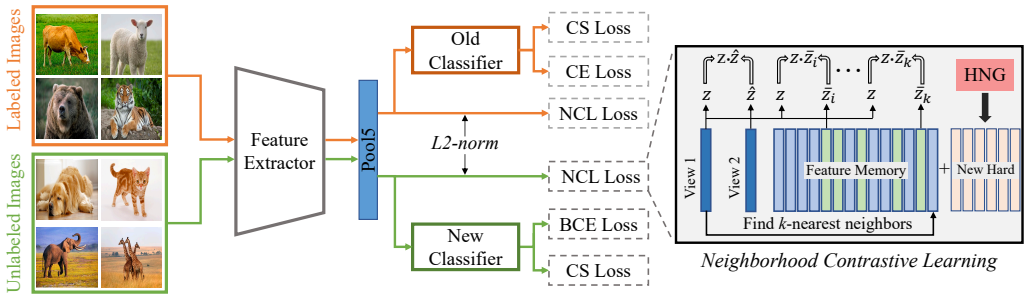


Figure 2. The proposed neighborhood contrastive learning framework for novel class discovery. Given training images sampled from the labeled and the unlabeled data, we forward them into the network to obtain corresponding representations. For the labeled data, the CE loss, CS loss and the proposed NCL loss are calculated with the ground-truth labels. For the unlabeled data, BCE loss and CS loss are computed to optimize the new classifier while the NCL loss is proposed to learn discriminative representation. CE: cross-entropy, BCE: binary cross-entropy, CS: consistency, NCL: neighborhood contrastive learning, HNG: hard negative generation.

Given a labeled dataset D^l and an unlabeled dataset D^u, containing C^l and C^u classes respectively (where the sets of classes in the two datasets are disjoint, but some degree of similarity between the two is necessary), the goal of NCD is to cluster the data in D^u, leveraging the knowledge from D^l. To discover the latent classes in D^u, we learn a shared feature extractor $\Omega : x \mapsto z \in R^H$ and two linear classifiers ϕ^l and ϕ^u, with C^l and C^u output neurons respectively, each followed by a softmax layer. At each training step, a batch of images is sampled from both D^u and D^l. Using data augmentation we generate two correlated views of the same batch and forward them into the feature extractor. The features of the labeled images are fed to the classifier ϕ^l, which is optimized with the cross-entropy loss using the labels. By using the binary cross-entropy loss, the classifier ϕ^u learns to infer the cluster assignments for the unlabeled images. Both classifiers are encouraged to output consistent predictions using the consistency loss. The representations z are refined by the proposed Neighborhood Contrastive Loss (NCL) on both labeled and unlabeled samples. The overall framework is depicted in Fig. 2.

Our framework can be described in detail as follows. We first learn a label-agnostic image representation by self-supervision learning [22] using both labeled and unlabeled datasets. Subsequently, high-level features are learned using supervision on the labeled dataset. Given a sample and its label, we optimize the network using the *cross-entropy* loss. The cluster discovery is performed by using the cosine similarity of the features to estimate pairwise pseudo-labels. Specifically, given a pair of images (x_i^u, x_j^u) sampled from dataset D^u, we extract features (z_i^u, z_j^u) and compute their cosine similarity $\delta\left(z_i^u, z_j^u\right) = z_i^{u\top} z_j^u / \|z_i^u\| \|z_j^u\|$. The pairwise pseudo-label is then assigned as follows:

$$\hat{y}_{i,j} = \mathbb{1}\left[\delta\left(z_i^u, z_j^u\right) \geq \lambda\right], \tag{3}$$

where λ is a threshold that represents the minimum similarity for two samples to be assigned to the same latent class. Then, the pairwise pseudo-label is compared to the inner product of the outputs of the unlabeled head $p_{i,j} = \phi^u\left(z_i^u\right)^\top \phi^u\left(z_j^u\right)$. The network is optimized using the *binary cross-entropy* loss:

$$\ell_{bce} = -\hat{y}_{i,j} \log\left(p_{i,j}\right) - \left(1 - \hat{y}_{i,j}\right) \log(1 - p_{i,j}). \tag{4}$$

The last building block is the consistency loss, which enforces the network produce similar predic-

tions for an image x_i and its correlated view $\hat{x}_i$. We use *mean squared error*:

$$\ell_{mse} = \frac{1}{C^l}\sum_{i=1}^{C^l}\left(\phi_i^l\left(z^l\right) - \phi_i^l\left(\hat{z}^l\right)\right)^2 + \frac{1}{C^u}\sum_{j=1}^{C^u}\left(\phi_j^u\left(z^u\right) - \phi_j^u\left(\hat{z}^u\right)\right)^2. \tag{5}$$

The overall loss for this baseline reads as:

$$\ell_{base} = \ell_{ce} + \ell_{bce} + \omega\left(t\right)\ell_{mse}, \tag{6}$$

where the coefficient $\omega\left(t\right)$ is a ramp-up function.

Neighborhood Contrastive Learning is performed as follows. Given a set of stochastic image transforms, we generate two correlated views $(x^u, \hat{x}^u)$ of a generic unlabeled sample to be used as a positive pair. Subsequently, we apply the network Ω to extract $(z^u, \hat{z}^u)$ from the views. This operation is repeated for all the samples of a batch of length B. We also maintain a queue M^u of features stored from past training steps, which initially are regarded as negatives, denoted with $\bar{z}^u$. The contrastive loss for the positive pair is written as:

$$\ell_{(z^u,\hat{z}^u)} = -\log\frac{e^{\delta(z^u,\hat{z}^u)/\tau}}{e^{\delta(z^u,\hat{z}^u)/\tau} + \sum_{m=1}^{|M^u|} e^{\delta(z^u,\bar{z}_m^u)/\tau}}, \tag{7}$$

where $\delta(\cdot,\cdot)$ is the cosine similarity and τ is a temperature parameter that controls the scale of distribution.

We generate pseudo-positive pairs of samples, *i.e.*, to consider the *neighbors* of the representation z^u as instances of the same class. To do so, we leverage the labeled dataset D^l to bootstrap the representations, and then use them to infer the relationships between unlabeled examples in D^u. We can retrieve the top-k most similar features from the queue for a query z^u:

$$\rho_k = \underset{\bar{z}_i^u}{\text{argtop}_k}\left(\left\{\delta\left(z^u, \bar{z}_i^u\right) \mid \forall i \in \{1,\ldots,|M^u|\}\right\}\right). \tag{8}$$

Assuming the examples in ρ_k are false-negatives (*i.e.*, they belong to the same class as z^u), we can regard them as pseudo-positives and write their contributions in the contrastive loss as follows:

$$\ell_{(z^u,\rho_k)} = -\frac{1}{k}\sum_{\bar{z}_i^u \in \rho_k}\log\frac{e^{\delta(z^u,\bar{z}_i^u)/\tau}}{e^{\delta(z^u,\hat{z}^u)/\tau} + \sum_{m=1}^{|M^u|} e^{\delta(z^u,\bar{z}_m^u)/\tau}}. \tag{9}$$

Finally, we can introduce our *Neighborhood Contrastive loss* as follows:

$$\ell_{ncl} = \alpha\ell_{(z^u,\hat{z}^u)} + (1-\alpha)\,\ell_{(z^u,\rho_k)}, \tag{10}$$

where α controls the weight of the two components. In the case of the labeled dataset D^l, instead of using the network to mine the pseudo-positives, we can directly use the ground-truth labels to retrieve the set of positives from the queue of labeled set M^l for a sample x_i^l with features z_i^l:

$$\rho = \left\{\bar{z}_j^l \in M^l : y_i = y_j\right\} \cup \hat{z}_i^l. \tag{11}$$

Note that ρ contains both the features $\hat{z}_i^l$ of the correlated view $\hat{x}_i^l$ and the other samples belonging to the same class. Using this supervision, our Neighborhood Contrastive loss can be reduced to

the *Supervised Contrastive loss* [23]:

$$\ell_{scl} = -\frac{1}{|\rho|} \sum_{\hat{z}_j^l \in \rho} \log \frac{e^{\delta\left(z_i^l, \hat{z}_j^l\right)/\tau}}{e^{\delta\left(z_i^l, \hat{z}_i^l\right)/\tau} + \sum_{m=1}^{|M^l|} e^{\delta\left(z_i^l, \bar{z}_m^l\right)/\tau}}. \tag{12}$$

Hard Negative Generation is performed as follows. Given a view x^u of an image belonging to the unlabeled set and its representation in the feature space z^u, we can select easy negatives by looking at the features with minimal similarity in the queue M^u:

$$\varepsilon_k = \underset{\bar{z}_i^u}{\text{argtop}_k} \left(\{ -\delta\left(z^u, \bar{z}_i^u\right) | \forall i \in \{1, \ldots, |M^u|\} \} \right). \tag{13}$$

Let us also consider a queue M^l containing labeled samples stored from past training steps. These are by definition true negatives with respect to x^u. For each $\bar{z}^u \in \varepsilon_k$ we randomly sample a feature $\bar{z}^l \in M^l$ and compute the following:

$$\zeta = \mu \cdot \bar{z}^u + (1 - \mu) \cdot \bar{z}^l, \tag{14}$$

where μ is the mixing coefficient. This process of cycling through ε_k is repeated N times such that the resulting set of mixed negatives η will contain $k \times N$ features. Then, the hardest negatives are filtered from η, using the cosine similarity as before:

$$\eta_k = \underset{\zeta_i}{\text{argtop}_k} \left(\{ \delta\left(z^u, \zeta_i\right) | \forall i \in \{1, \ldots, k \times N\} \} \right). \tag{15}$$

This results in a set η_k of hard negatives. Finally the queue for x^u is derived by adding the newly generated mixed negatives into the queue M^u:

$$M^{u'} = M^u \cup \eta_k, \tag{16}$$

and the contrastive loss is computed as in Eq. 7 and Eq. 9, but replacing M^u with $M^{u'}$. This pipeline for hard negative generation is repeated for each unlabeled sample in the current batch, as they will have different sets of easy negatives.

Considering the baseline model, neighborhood contrastive learning on unlabeled data, supervised contrastive learning on labeled data, and the hard negative generation on unlabeled data, the overall loss for our framework is:

$$\ell_{all} = \ell_{base} + \ell_{ncl} + \ell_{scl}. \tag{17}$$

Experimental analysis and results. We conduct experiments on three datasets: CIFAR-10 [17], CIFAR-100 [17] and ImageNet [18]. We employ average clustering ACC to evaluate the performance of different methods on unlabelled data. We compare the proposed approach with k-means [20], KCL [11], MCL [12], DTC [10] and RS [1]) in Table 2. Given these results we can conclude that using self-supervised learning generally can improve the results of all methods, except when evaluated k-means [20] on CIFAR-100. For example, when using self-supervised learning, the ACC of KCL [11] is increased from 66.5% to 72.3% and from 14.3% to 42.1% on CIFAR-10 and CIFAR-100, respectively. This indicates the effectiveness of self-supervised learning. Second, two versions of our method outperform the state-of-the-art methods (whether using self-supervised learning or not) by a large margin on all datasets, especially on CIFAR-100 and ImageNet. Specifically, our full method achieves ACC=93.4% on CIFAR-10, ACC=86.6% on CIFAR-100 and ACC=90.7% on ImageNet, respectively. These results are higher than RS [1] by +3% on CIFAR-10, +13.4% on CIFAR-100 and +8.2% on ImageNet, respectively.

Table 2. Comparison with state-of-the-art methods on CIFAR-10, CIFAR-100 and ImageNet for novel class discovery. Clustering ACC is reported on the unlabelled set. "" indicates methods that initialize models with self-supervised learning, except when evaluated on ImageNet. Ours: our method with both neighborhood contrastive learning and hard negative generation, Ours w/o HNG: our method without hard negative generation.*

Method	CIFAR-10	CIFAR-100	ImageNet
Methods without self-supervised learning			
k-means [20]	65.5±0.0%	56.6±1.6%	71.9%
KCL [11]	66.5±3.9%	14.3±1.3%	73.8%
MCL [12]	64.2±0.1%	21.3±3.4%	74.4%
DTC [10]	87.5±0.3%	56.7±1.2%	78.3%
Methods with self-supervised learning			
k-means [20]*	72.5±0.0%	56.3±1.7%	71.9%
KCL [11]*	72.3±0.2%	42.1±1.8%	73.8%
MCL [12]*	70.9±0.1%	21.5±2.3%	74.4%
DTC [10]*	88.7±0.3%	67.3±1.2%	78.3%
RS [1]*	90.4±0.5%	73.2±2.1%	82.5%
Ours* w/o HNG	**93.4±0.2%**	**82.3±2.6%**	**89.5%**
Ours*	**93.4±0.1%**	**86.6±0.4%**	**90.7%**

3.2.3. Relevant publications

- Z. Zhong, L. Zhu, Z. Luo, S. Li, Y. Yang, N. Sebe, OpenMix: Reviving Known Knowledge for Discovering Novel Visual Categories in an Open World, CVPR 2021 [24].
  Zenodo record: https://zenodo.org/record/5014206.
- Z. Zhong, E. Fini, S. Roy, Z. Luo, E. Ricci, N. Sebe, Neighborhood Contrastive Learning for Novel Class Discovery, CVPR 2021 [25].
  Zenodo record: https://zenodo.org/record/5014108.

3.2.4. Relevant software and/or external resources

- The Pytorch implementation of our work "Neighborhood Contrastive Learning for Novel Class Discovery" can be found in https://github.com/zhunzhong07/NCL.

3.2.5. Relevant WP8 Use Cases

Our tools on novel class discovery contribute to use cases (a) 3A3 and 4C1 by providing solutions to analyze visual content, and (b) 7A3 by supporting the organization of image and video collections.

3.3. Comparative Study of Class-Incremental Learning Algorithms (Task 3.1)

Contributing partners: CEA

The ability of artificial agents to augment their capabilities when confronted with new data is an open challenge in artificial intelligence. The main challenge faced in such cases is catastrophic forgetting, i.e., the tendency of neural networks to underfit past data when new ones are ingested. A first group of approaches tackles forgetting by increasing deep model capacity to accommodate new knowledge. A second type of approaches fix the deep model size and introduce a mechanism whose objective is to ensure a good compromise between stability and plasticity of the model. While the first type of algorithms were compared thoroughly, this is not the case for methods which

exploit a fixed size model. Here, we focus on the latter, place them in a common conceptual and experimental framework and discuss the following contributions: (1) define six desirable properties of incremental learning algorithms and analyze them according to these properties, (2) propose a common evaluation framework which is more thorough than existing ones in terms of number of datasets, size of datasets, size of bounded memory and number of incremental states, and (3) provide experimental evidence that it is possible to obtain competitive performance without the use of knowledge distillation to tackle catastrophic forgetting. The main experimental finding is that none of the existing algorithms achieves the best results in all evaluated settings. Important differences arise notably if a bounded memory of past classes is allowed or not. A detailed description of the contributions summarized here is available in [26].

Desirable Properties of Incremental Learning Algorithms. We define a common analysis framework made of six desirable properties of incremental learning algorithms. This set of properties builds on the one proposed in [27], which already includes three of them (marked with * below):

1. **Complexity (C)*** - capacity to integrate new information with a minimal change in terms of the model structure. For a deep neural network, only the size of the classification layer should grow. Otherwise, the total number of parameters of the model is likely to increase strongly, especially at large scale.

2. **Memory (M)*** - ability to work with or without a fixed-size memory of past classes. Naturally, algorithms that do not require past data are preferable, but their performance is likely to be lower, especially if complexity growth is minimized.

3. **Accuracy (A)*** - performance for past and new classes should approach that of a non-incremental learning process that has access to all data at all times.

4. **Timeliness (T)** - delay needed between the occurrence of new data and their integration in the incremental models.

5. **Plasticity (P)** - capacity to deal with new classes that are significantly different from the ones that were learned in the past [28].

6. **Scalability (S)** - the aptitude to learn a large number of classes, typically up to tens of thousands, and ensure usability in complex real-world applications.

A mapping of different groups of existing approaches with respect to the desirable properties described above is provided in Table 3. The main finding here is that each group is characterized by a combination of strengths and limitations. This happens because it is impossible to satisfy all desirable properties jointly. The practical choice of one approach over the others should be done with respect to the application needs and constraints.

Experimental analysis and results. We compare state of the art methods and modified versions of them using four public datasets: **ILSVRC** [48] - fine-grained object recognition, **VGGFACE2** [49] - face recognition, **Google Landmarks** [50] (**LANDMARKS** below) - tourist landmark recognition, and **CIFAR100** [51] - basic-level object recognition. The first three datasets include 1000 classes and the fourth includes 100 classes. We vary the number of incremental states and the available memory for each dataset since these are the two most important parameters of the class IL scenario. Note that the that performance for the setting without memory is analyzed separately since the absence of memory makes a part of existing methods unusable. Results are presented for a series of competitive existing methods, which exploit knowledge distillation, and versions of them in which the knowledge distillation is ablated.

Table 3. Analysis of the main groups of incremental learning algorithms with respect to their desirable properties. A global assessment with recommended use cases is also provided.

	Model-Growth based	Fixed-Representation based	Fine-Tuning based
Complexity	The model evolves by adding parameters and weights to interconnect them [29, 30, 31] or small networks [32] to include new knowledge. The challenge is to optimize the effect of model growth on performance [33, 29].	The model is fixed after the first step. In a basic setting [34, 35], the only parameters added are those needed for new classes weights, but additional parameters can be added to improve stability [36].	This group of IL methods work with a fixed structure of the backbone model. The number of parameters is only marginally affected by the modifications of the classification layer [37, 38, 39].
Memory	The model growth allows for the deployment of these methods without the use of an exemplar memory. Memory is allocated to additional model parameters and weights instead of raw data for past classes, which is a more parsimonious way to store information about past classes [40, 32, 41].	Fixed-representations do not update the model during the incremental learning process and thus have a very low dependency on the memory of past classes [34]. Class weights are learned when they are first encountered and can be used throughout all subsequent incremental states.	Performance is heavily dependent on the size of the past memory. However, storing a large amount of past exemplars is contradictory to IL objectives. Memory needs are reduced by exploiting knowledge distillation [37, 38, 42, 39] or by exploiting statistical properties of past states [43, 44].
Accuracy	Performance is correlated to the amount of model growth allowed. If growth is limited, MG-based methods have lower performance compared to FT-based ones [33]. A significant growth [41] gives performance comes close to that of classical learning, but limits the interest of these methods.	Accuracy is lower compared to FT-based methods because the model is not updated incrementally. High performance can be obtained with fixed-representations if the initial model is learned with a large dataset [34], but the existence of such a dataset is a strong assumption in IL.	Recent approaches report strong performance gain compared to previous work such as [37, 42, 27] through more sophisticated definitions of distillation [38] or through the casting of IL as an imbalanced learning problem [43, 44] or a combination of both [39].
Timeliness	The complexity of model growth is similar to that of FT-based methods since retraining is needed for each incremental update [33].	Only the classifier weights layer needs to be trained and new knowledge is integrated in a timely manner [45].	New classes are not recognizable until retraining is finished, which is problematic for time-sensitive applications.
Plasticity	MG-based methods are specifically designed to cope with different visual tasks [40, 46]. The challenge is to minimize the amount of additional parameters needed to accommodate each new task [33].	Plasticity is limited since the representation is learned in the first state and then fixed. Performance drops significantly if the incremental tasks change a lot and the initial representation is not transferable anymore [47].	The model updates enable adaptation to new data as they are streamed into the system. If no memory is allowed, plasticity is too important and this shift is controlled through knowledge distillation or imbalance handling.
Scalability	These methods scale well to new classes or tasks as long as the systems in which they are deployed have sufficient resources to support the underlying model growth for training and inference phases.	The dependence on the bounded memory is limited and FR-based methods can include many classes. This is possible because class weights are learned in their initial state and reused later.	The size of the bounded memory determines the number of past classes for which exemplars can be stored and which are still recognizable when new ones are integrated.
Global assessment	Approaches in this group cope well with new data, are not or weakly dependent on a memory, and are scalable to many classes. However, complexity is a disadvantage since the model has to grow in order to integrate new knowledge. They also require retraining when new classes are added, and timeliness is not optimal. They are usable when: model complexity can be increased, streamed data vary a lot, no memory is allowed and timeliness is not essential.	Fixed-representation methods inherit the advantages and disadvantages of transfer learning schemes. Model complexity is constant and they can be updated in a timely manner. They do not depend on memory and are scalable. However, they depend on the quality of the initial representation and have a low plasticity. They are usable when: model complexity should stay constant, data variability is low, no memory is available and updates are needed in a timely manner.	FT methods are adequate when trying to optimize the architecture complexity and the plasticity [28] of representations. They are not timely since retraining is required. The use of a fixed memory limits their use at very large scale. They are usable when: model complexity is fixed, streamed data vary a lot, storage is available for past data and timeliness is not essential.

Table 4. Top-5 average incremental accuracy (%) for IL methods using different memory sizes and numbers of incremental states. G_{IL} is an aggregated score over all configurations tested which measures the gap with a classical learning (noted Full). Best results are in bold.

States	$T = 10$												$\|\mathcal{K}\| = 0.5\%$								G_{IL}
Dataset	ILSVRC			VGGFACE2			LANDMARKS			CIFAR100			ILSVRC		VGGFACE2		LANDMARKS		CIFAR100		
$\|\mathcal{K}\|$	2%	1%	0.5%	2%	1%	0.5%	2%	1%	0.5%	2%	1%	0.5%	T=20	T=50	T=20	T=50	T=20	T=50	T=20	T=50	
iCaRL[27]	79.3	76.5	71.0	96.0	95.3	93.9	95.1	94.0	91.8	66.5	56.1	47.9	55.9	45.0	88.5	78.2	86.8	82.4	35.5	35.4	-7.36
$LUCIR^{CNN}$ [38]	79.9	76.4	72.6	97.2	**96.9**	**96.5**	97.2	96.6	96.1	79.8	75.4	69.9	63.9	55.3	93.5	88.3	93.7	90.5	53.5	47.9	-4.13
$LUCIR^{NEM}$ [38]	80.5	80.0	79.4	96.2	96.0	95.7	95.4	94.9	94.4	82.6	80.8	78.8	73.6	66.3	92.7	87.9	91.9	89.8	69.0	63.0	-4.33
FT [43]	79.4	74.4	65.9	96.4	94.5	91.3	96.6	94.7	91.4	82.4	77.9	70.7	69.4	64.3	91.6	89.2	90.9	89.0	64.3	54.8	-5.19
FT^{NEM} [26]	81.4	79.0	75.0	96.4	95.4	94.0	96.1	94.6	92.6	85.1	81.7	76.0	76.5	69.0	94.0	91.1	91.9	89.9	68.8	55.9	-4.28
FT^{BAL} [26]	84.0	80.9	76.5	97.0	95.7	92.4	96.9	95.3	92.2	80.0	74.0	69.0	75.9	67.1	92.3	89.5	91.2	88.9	62.9	54.2	-4.70
BiC [39]	**85.5**	**82.8**	**79.7**	**97.3**	96.6	95.7	**97.9**	**97.3**	**96.6**	**88.8**	**87.6**	**83.5**	74.6	63.9	92.3	85.3	**94.7**	90.5	50.5	19.6	-4.03
ScaIL [44]	82.0	79.8	76.6	96.5	95.8	95.2	97.3	96.0	94.0	85.6	83.2	79.1	76.6	70.9	**95.0**	**92.4**	92.6	90.4	69.8	51.0	-3.70
IL2M [43]	80.9	78.1	73.9	96.7	95.4	93.4	96.5	94.7	92.5	81.8	77.0	71.2	70.9	60.6	92.5	88.4	90.8	88.1	61.5	51.0	-4.95
FT^{th} [26]	84.3	82.1	78.3	97.2	96.3	94.8	97.2	95.8	94.0	86.4	83.9	79.1	**78.6**	**71.2**	94.3	91.6	92.9	**90.7**	**71.4**	**57.9**	**-3.62**
FT^{init}_{L2} [52]	79.2	76.5	73.0	95.9	95.2	94.6	97.0	95.5	92.7	83.4	80.5	75.2	73.6	67.3	94.6	91.4	91.2	88.5	63.6	44.1	-4.43
FR [27]	76.7	76.6	76.4	91.7	91.5	89.7	93.8	93.5	93.5	79.5	79.4	78.7	69.2	58.2	85.8	75.2	89.3	82.8	62.3	33.5	-7.62
DeeSIL [34]	75.5	75.1	74.3	92.7	92.5	92.2	94.0	93.7	93.2	66.9	65.8	64.2	73.0	58.1	87.2	80.0	90.5	85.1	63.9	44.0	-6.92
REMIND [53]	80.9	80.7	78.2	94.7	93.2	93.0	96.3	95.8	94.7	60.7	60.7	60.7	73.9	65.0	87.4	80.1	92.8	88.6	52.8	46.4	-6.02
Full	92.3			99.2			99.1			91.2			92.3		99.2		99.1		91.2		-

The results presented in Table 4 indicate that the best overall performance is obtained with FT^{th}, a method which considers IL with memory as a sub-problem of imbalanced learning [54]. It corrects the bias in favor of new (majority) classes by correcting initial class predictions with a term which boosts past (minority) classes. Interestingly, FT^{th} does not exploit knowledge distillation which is extensively used in class IL to preserve the stability of incremental deep representations. The second best aggregated performance is obtained with *ScaIL* [44], a method which scales classifier weights to make them more comparable between new and past classes. *BiC* [39], which adds a linear layer for bias correction, also has a very interesting behavior. It outperforms other methods for $T = 10$ states but has poorer behavior for larger number of states. This is explained by the fact that *BiC* needs a validation set whose size needs to be sufficient in order to optimize the bias correction layer. This size is clearly not sufficient, especially for small and medium sized datasets such as CIFAR100. Globally, strong progress was achieved in class IL with memory between the proposal of *iCaRL* [27] in 2017 and today. However, the gap with a classical training which has access to the full dataset at all times remains important. IL with memory thus remains and open research topic.

The results for class IL without memory are presented in Table 5. Here, the best overall results are obtained with *DeeSIL* [34], a simple method which is based on the transfer of fixed representations learned in the initial state of the IL process. This result is intriguing insofar it interrogates the progress made in class IL without memory which is based, in a majority of cases, on the use of knowledge distillation [27, 38]. However, we note that fixed representations are strongly dependent on the quality of the fixed initial representations. Their performance drops significantly when the initial representation is learned with a small number of classes (large T value for large datasets or small datasets overall). $Deep - SLDA$ [45], which is also based on a fixed representation but uses a more refined training procedure compared to *DeeSIL*, has the second best global performance. FT^{init}_{siw+mc} [52], which exploits normalized initial classifier weights in later incremental states, has the best performance among the fine-tuning based methods tested. It does not exploit knowledge distillation, which is classically implemented by methods such as *LwF* [27] and $LUCIR^{CNN}$ [38]. The results from Table 5 lead us to conclude that class IL without memory is a very open topic. Research is particularly needed in order to make fine-tuning-based methods competitive with fixed-representation-based methods.

Table 5. Top-5 average incremental accuracy (%) for IL methods without memory for past classes and different numbers of incremental states. G_{IL} is an aggregated score over all configurations tested which measures the gap with a classical learning (noted Full). Best results are in bold.

Dataset	ILSVRC			VGGFACE2			LANDMARKS			CIFAR100			G_{IL}
States	T=10	T=20	T=50	T=10	T=20	T=50	T=10	T=20	T=50	T=10	T=20	T=50	
LwF [27]	45.3	37.6	27.1	53.3	42.6	30.8	58.8	49.2	35.2	79.5	65.3	39.0	-34.72
LwF^{init} [52]	47.1	39.9	32.2	58.1	50.8	40.5	55.7	50.2	39.8	79.4	67.9	42.8	-31.97
LwF^{init}_{L2} [52]	24.5	39.7	32.0	57.1	50.7	40.5	52.1	50.5	40.0	79.5	68.1	43.3	-32.60
LwF^{init}_{siw} [52]	54.0	45.8	35.1	70.4	59.3	45.2	61.0	53.8	42.2	**80.0**	**68.8**	44.6	-28.06
$LUCIR^{CNN}$ [38]	57.6	39.4	21.9	91.4	68.2	32.2	87.8	63.7	32.3	57.5	35.3	21.0	-24.75
FT [29]	20.6	13.4	7.1	21.3	13.6	7.1	21.3	13.6	7.1	21.3	13.7	17.4	-54.91
FT^{init} [52]	61.0	44.9	23.8	90.9	64.4	33.1	68.8	49.4	22.2	55.1	40.8	19.9	-28.99
FT^{init}_{L2} [52]	51.6	43.3	34.5	76.8	66.8	55.1	61.4	52.5	39.2	47.5	39.3	22.5	-26.80
FT^{init}_{L2+mc} [52]	53.6	42.7	35.6	86.9	71.4	53.6	66.2	52.6	37.9	52.6	43.1	18.2	-25.02
FT^{init}_{siw+mc} [52]	64.4	54.3	41.4	88.6	84.1	62.6	79.5	64.5	43.2	59.7	44.3	18.4	-19.38
FR [27]	74.0	66.9	49.2	88.7	83.0	54.4	**93.6**	88.1	71.2	73.1	54.8	27.4	-16.30
DeeSIL [34]	**73.9**	**67.5**	53.9	**92.3**	**87.5**	75.1	**93.6**	**91.1**	**82.1**	65.2	63.4	32.3	**-9.22**
REMIND [53]	62.2	56.3	44.4	86.8	81.4	69.2	84.5	79.6	69.0	52.7	40.5	25.7	-22.00
Deep-SLDA [45]	70.3	64.5	**56.0**	90.2	85.4	**78.2**	89.3	86.4	81.3	68.9	64.4	**54.5**	-15.40
Full		92.3			99.2			99.1			91.2		-

3.3.1. Relevant publications

- E. Belouadah, A. Popescu, I. Kanellos, A comprehensive study of class incremental learning algorithms for visual tasks, Neural Networks, 2020 [26].
  Zenodo record: https://zenodo.org/record/4462956.

3.3.2. Relevant software and/or external resources

- The Python implementation of "several class incremental learning methods" can be found in https://github.com/EdenBelouadah/class-incremental-learning

3.3.3. Relevant WP8 Use Cases

This tool can be exploited in the following use cases: (1) UC2 (2A and 2B) because incremental learning allows AI systems to process new topics which appear in news and thus improve tagging and search capabilities, (2) UC3 (3C) because the automatic management of unexpected journalistic events requires a swift update of recognition models for relevant content such as faces or company brands.

3.4. DNN Education (Task 3.1)

Contributing partners: AUTH

AUTH research in T3.1 revolves around the concept of Deep Neural Network (DNN) "education", a proposed dynamic adaptation framework for neural models that combines and unifies Out-of-Distribution (OOD) detection, incremental/continual/lifelong learning and neural distillation. The envisioned framework will allow interacting, pretrained DNNs to on-the-fly and dynamically self-assess their knowledge about current unknown/novel/test data points and, if they deem it necessary, request relevant retraining from dynamically and autonomously discovered teacher models, without forgetting their past knowledge. The main novelty entails on-the-fly, automated knowledge

exchanges within the grid of participating DNN models, with dynamic, ad-hoc and autonomously assigned student-teacher roles. In order to provide a proof-of-concept prototype within the span and context of the AI4Media project, AUTH focuses on Convolutional Neural Networks (CNNs) for image classification. Overall, the proposed DNN education framework is complementary to the rest of T3.1, which focuses specifically on scientific advances in lifelong learning.

During M1-M12 of the project, AUTH research focused on the first ingredient of the DNN education framework, i.e., Out-of-Distribution detection (OOD) for DNNs. To this end, a survey of the relevant state-of-the-art was conducted, resulting in a technical report that has been submitted as a conference paper. The survey presents an overview and a taxonomy of related methods, as well as typical evaluation datasets and metrics. Without loss of generality, the focus is on image classification using Convolutional Neural Networks (CNNs).

Besides this survey, AUTH also performed research on two novel OOD detection algorithms, which is still on-going. They are detailed in the following two subsections.

3.4.1. Adversarial Out-of-Distribution Detection using Regularized Reconstructions

Typical OOD detection methods for DNNs fall under either the *discriminative* or the *generative* family. Both approaches rely on comparing test data points with the training dataset, irrespective of the actual DNN model being employed during inference. A more modern approach is to design model-specific variants of similar discriminative or generative models, where the representations of the input data points constructed internally by the classifier DNN may also be used for distinguishing between in-distribution (ID) and OOD data [55, 56, 57, 58, 59, 60, 61, 62]. Additionally, several different methods have been proposed that rely on directly or indirectly measuring *epistemic uncertainty* (i.e., uncertainty due to lack of relevant knowledge encoded in the model parameters) for each test-time prediction [63, 64, 65, 66].

The first line of AUTH research during M1-M12 of the project was to design a novel model-specific OOD detector for pretrained image classification CNN models, which does not require example OOD datasets during training. Thus, assuming such a classification CNN model has been pretrained on in-distribution (ID) dataset $\mathcal{K}$, the goal is for a separate OOD detection DNN model to be able to discriminate, during inference, whether each test/unknown data point (image) $\mathbf{X}_i$ has been sampled from the data distribution of $\mathcal{K}$ or from that of a different, unknown, OOD dataset $\mathcal{L}$.

A DenseNet [67] with depth $L = 100$ and growth rate k= 12 (Dense-BC) was employed as the base image classification CNN, with the output of its first Transition Layer during the inference stage forming the input of the proposed OOD detector. The proposed method is evidently model-specific, since it processes early-layer convolutional features computed by the pretrained (on $\mathcal{K}$) DenseNet-based classifier, instead of the raw RGB test-stage images.

The proposed OOD detection architecture consists of a Convolutional Autoencoder (CA) and a Generative Adversarial Network (GAN) [68], composed of a Generator G Discriminator D. The CA and the the GAN are trained separately and consecutively, while during the inference stage of the OOD detector G is not needed; it can be discarded after training the GAN. The CA follows the Encoder-Decoder paradigm and is trained using ID data from $\mathcal{K}$. The Encoder gradually shrinks (w.r.t. both the spatial dimensions and the number of channels) its tensorial input, i.e., the output $\mathbf{Y}_i \in \mathbb{R}^{M \times N \times C}$ of the first Dense-BC Transition Layer, while the Decoder gradually recovers the original input. The respective reconstructed outputs of the CA, i.e., $\tilde{\mathbf{Y}}_i \in \mathbb{R}^{M \times N \times C}$, $\forall i \in \mathcal{K}$, are subtracted from the corresponding inputs in order to form *reconstruction error maps* $\mathbf{E}_i \in \mathbb{R}^{M \times N \times C}, \forall i \in \mathcal{K}$, where:

$$\mathbf{E}_i = \tilde{\mathbf{Y}}_i - \mathbf{Y}_i. \tag{18}$$

The GAN is trained so as to approximate the data distribution of the computed $\mathbf{E}_i, \forall i \in \mathcal{K}$. Thus, D is a binary classifier trained so as to discriminate reconstruction error maps computed by the CA on ID data and on OOD data. The underlying idea is that the CA will most likely reconstruct significantly better any ID data rather than OOD data during the inference stage, thus facilitating D in its job. The naive alternatives would be:

- to directly train a GAN on all $\mathbf{Y}_i$ derived from $\mathcal{K}$, so as to implicitly capture the ID data distribution in the feature space, without a CA, or
- directly compute the quality of the reconstruction (e.g., by reducing each reconstruction error map into a scalar reconstruction error) and threshold it, so as to discriminate between ID and OOD data without a Discriminator.

However, the former naive scenario would ignore discriminant information captured by the quality of the CA reconstruction, while the latter one would require manual and brittle threshold tuning.

According to work performed up to now, a key component of the process is the degree of restraining the ability of CA to reconstruct its input, i.e., how to keep the generalization ability of the CA under check, in order to retain discrimination between ID and OOD data. To this end, different regularizers applied on the intermediate, latent representation (the output of the Encoder) during training the CA are currently being compared.

This is still on-going research, expected to be completed and submitted in paper form until the end of 2021.

3.4.2. Out-Of-Distribution Detection for Self-Supervised Monocular Depth Map Estimation

A second line of AUTH research regarding OOD detection that was pursued during M1-M12 concerned DNN-based dense image prediction tasks, instead of simple classification. This is a much less explored area, with relatively sparse existing literature. For instance, [69] focused on semantic image segmentation and derived inference-stage uncertainty for different image regions, by estimating segmentation quality for each predicted segment using statistical properties of the output segmentation map. In a different approach for OOD detection in semantic segmentation [70], spatial entropy heatmaps and related entropy thresholds first determine whether each pixel is OOD. Subsequently, the process is further improved at the segment level, using aggregated statistical properties of the segmentor's output and geometrical characteristics of the derived segments. A different, non-supervised dense image prediction task is self-supervised monocular depth map estimation from subsequent video frames. OOD detection literature for this task is even more limited, despite the importance of robust visual depth map estimators in fields such as robotics. For example, in [71], visual odometry data are exploited to train an autoencoder alongside a depth estimator, with the former one enforcing consistency in the losses during training, reducing the number of OOD samples. In contrast, in [72] a self-teaching method is proposed, where two architecturally identical instances of the depth estimation DNN Monodepth2 [73] are consecutively in a teacher-student fashion relying essentially on neural knowledge distillation: the teacher is first optimized in the typical self-supervised manner, while the student exploits the teacher outputs as direct supervision targets. Then, the differences between student and teacher inference-stage predictions on the training set are exploited in order to learn an uncertainty prediction model. Finally, in [74], a very simple method is proposed: each test-stage video frame pair is forwarded along the pretrained depth estimation DNN twice: once in its regular form and once in its horizontally flipped form; the absolute difference between the two respective predicted depth maps is exploited as an uncertainty estimator.

Partly inspired from [72] and [74], but following a different approach, AUTH is investigating an OOD detection method relying on two entirely different DNN architectures that are independently and consecutively trained for self-supervised monocular depth map estimation on the same training set. The first one is a typical depth map estimator, such as Monodepth2 [73], while the second one is an Image-to-Image Translator (I2I) based on Conditional GANs [75]. Thus, not only the neural architectures differ between them, but also the learning paradigms and the corresponding loss functions (self-supervised versus adversarial learning). The I2I Translator is optimized using the depth maps derived by the trained Monodepth2 DNN as targets; therefore, a self-teaching component is evident in the proposed method, as in [72]. During the inference stage, each test image is fed to both models and both predict a depth map. As in [74], the difference between these two depth maps is employed as a prediction uncertainty indicator and, by thresholding the uncertainty, as an OOD detector. The underlying intuition is that the two neural architectures will most likely give rise to similar outputs when fed ID data, but their predictions will tend to vary widely for OOD data due to their very different architectural properties and learning paradigms.

This is still on-going research, expected to be completed and submitted in paper form until the end of 2021.

3.4.3. Relevant publications

- I. Mademlis, I. Pitas, "Out-Of-Distribution Detection for Deep Convolutional Neural Networks: An Overview", technical report, submitted as conference paper [76].

3.5. Towards Compatible Lifelong Learning Representations (Task 3.1)

Contributing partners: UNIFI

We are investigating novel methods to learn internal *feature representations* models that are *compatible* with previously learned ones. Compatible features allow "new" learned features to be compared directly to "old" features, so they can be used interchangeably in time. This enables visual search systems to avoid extracting new features for all previously seen images (i.e., the gallery-set) when updating the representation model. Extracting new features is typically quite expensive or infeasible in the case of very large gallery-sets (i.e., face-recognition systems, social networks, life-long learning systems, autonomous robotics). Our method trains representation models exploiting pre-allocated fixed classifier based on regular polytopes [77]. Recently, these classifiers have been shown to enable learning of *stationary* feature representation without affecting classification performance [78]. Our objective is to use these two "parts" to obtain a lifelong compatible learning system. Concisely *stationarity* $\Rightarrow$ *compatibility* (i.e. if features "move" they cannot be compatible with old features, therefore we want to keep them as *stationary* as possible while learning). Our method is based on the fact that *RePoNets [77],[78]* $\Rightarrow$ *stationarity*). Our ongoing investigation (under review) is the chaining of the two implications to obtain: *RePoNets* $\Rightarrow$ *stationarity* $\Rightarrow$ *compatibility*.

3.5.1. Regular Polytope Networks

Deep Convolutional Neural Networks (DCNNs) have achieved state-of-the-art performance on a variety of tasks [79, 80] and have revolutionized Computer Vision in both classification [81, 82] and representation [83, 84]. In DCNNs, both representation and classification are typically jointly learned in a single network. The classification layer placed at the end of such models transforms the d-dimension of the network internal feature representation to the K-dimension of the output

class probabilities. Despite the large number of trainable parameters that this layer adds to the model (i.e. $d \times K$), it has been verified that its removal only causes a slight increase in error [85]. Moreover, the most recent architectures tend to avoid the use of fully connected layers [86] [87] [14]. It is also well known that DCNNs can be trained to perform metric learning without the explicit use of a classification layer [88] [89] [90]. In particular, it has been shown that excluding from learning the parameters of the classification layer causes little or no decline in performance while allowing a reduction in the number of trainable parameters [91]. Fixed classifiers also have an important role in the theoretical convergence analysis of training models with batch-norm [92]. Very recently it has been shown that DCNNs with a fixed classifier and batch-norm in each layer establish a principle of equivalence between different learning rate schedules [93].

All these works seem to suggest that the final fully connected layer used for classification is somewhat redundant and does not have a primary role in learning and generalization. In this paper we show that a special set of fixed classification layers *has a key role in modeling the internal feature representation* of DCNNs, while ensuring little or no loss in classification accuracy and a significant reduction in memory usage.

In DCNNs the internal feature representation for an input sample is the feature vector $\mathbf{f}$ generated by the penultimate layer, while the last layer (i.e. the classifier) outputs score values according to the inner product as:

$$z_i = \mathbf{w}_i^\top \cdot \mathbf{f} \tag{19}$$

for each class i, where $\mathbf{w}_i$ is the weight vector of the classifier for the class i. To evaluate the loss, the scores are further normalized into probabilities via the softmax function [94].

Since the values of z_i can be also expressed as $z_i = \mathbf{w}_i^\top \cdot \mathbf{f} = ||\mathbf{w}_i||\,||\mathbf{f}||\cos(\theta)$, where θ is the angle between $\mathbf{w}_i$ and $\mathbf{f}$, the score for the correct label with respect to the other labels is obtained by optimizing the length of the vectors $||\mathbf{w}_i||$, $||\mathbf{f}||$ and the angle θ they are forming. This simple formulation of the final classifier provides the intuitive explanation of how feature vector directions and weight vector directions align simultaneously with each other at training time so that their average angle is made as small as possible. If the parameters $\mathbf{w}_i$ of the classifier in Eq. 19 are fixed (i.e. set as non-trainable), *only the feature vector directions* can align toward the classifier weight vector directions and not the opposite. Therefore, weights can be regarded as fixed angular references to which features align.

According to this, we obtain a precise result on the spatio-temporal statistical properties of the generated features during the learning phase. Supported by the empirical evidence in [91] we show that not only the final classifier of a DCNN can be set as non-trainable with no loss of accuracy and

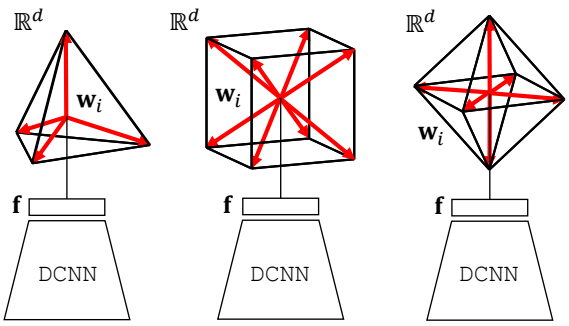


Figure 3. RePoNet. The fixed classifiers derived from the three regular polytopes available in $\mathbb{R}^d$ with $d \geq 5$ are shown. From left: the d-Simplex, the d-Cube and the d-Orthoplex fixed classifier. The trainable parameters $\mathbf{w}_i$ of the classifier are replaced with fixed values taken from the coordinate vertices of a regular polytope (shown in red).

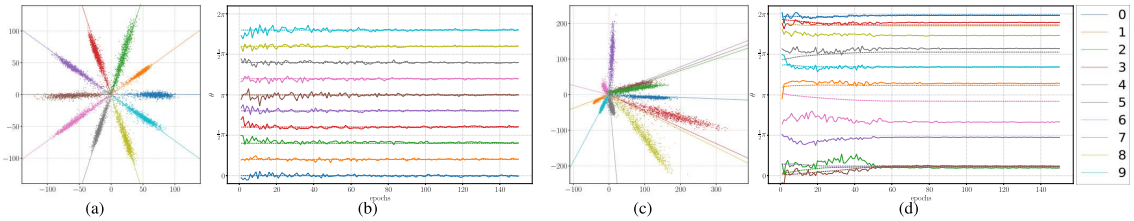


Figure 4. Feature learning on the MNIST dataset in a 2D embedding space. Fig. (a) and Fig. (c) show the 2D features learned by RePoNet and by a standard trainable classifier respectively. Fig. (b) and Fig. (d) show the training evolution of the classifier weights (dashed) and their corresponding class feature means (solid) respectively. Both are expressed according to their angles. Although the two methods achieve the same classification accuracy, features in the proposed method are both stationary and maximally separated.

with a significant reduction in memory usage, but that an appropriate set of values assigned to its weights allows learning a maximally separated and strictly stationary embedding while training. That is, the features generated by the Stochastic Gradient Descent (SGD) optimization have constant mean and are angularly centered around their corresponding fixed class weights. Constant known mean implies that features cannot have non-constant trends while learning. Maximally separated features and their stationarity are obtained by setting the classifier weights according to values following a highly symmetrical configuration in the embedding space.

DCNN models with trainable classifiers are typically convergent and therefore, after a sufficient learning time has elapsed, some form of stationarity in the learned features can still be achieved. However, until that time, it is not possible to know where the features will be projected by the learned model in the embedding space. An advantage of the approach proposed in this paper is that it allows to define (and therefore to know in advance) where the features will be projected before starting the learning process.

Our result can be understood by looking at the basic functionality of the final classifier in a DCNN. The main role of a trainable classifier is to dynamically adjust the decision boundaries to learn class feature representations. When the classifier is set as non-trainable this dynamic adjustment capability is no longer available and it is automatically demanded to all the previous layers. Specifically, the work [91] reports empirical evidence that the expressive power of DCNN models is large enough to account for the missing dynamic adjustment capability of the classifier. We provide more systematic empirical evidence confirming and broadening the general validity of DCNNs with fixed classifiers (see [78]).

We show that our approach can be theoretically justified and easily implemented by setting the classifier weights to values taken from the coordinate vertices of a regular polytope in the embedding space. Regular polytopes are the generalization in any number of dimensions of regular polygons and regular polyhedra (i.e. Platonic Solids). Although there are infinite regular polygons in $\mathbb{R}^2$ and 5 regular polyhedra in $\mathbb{R}^3$, there are only three regular polytopes in $\mathbb{R}^d$ with $d \geq 5$, namely the d-Simplex, the d-Cube and the d-Orthoplex. Having different symmetry, geometry and topology, each regular polytope will reflect its properties into the classifier and the embedding space which it defines. Fig. 3 illustrates the three basic architectures defined by the proposed approach termed Regular Polytope Networks (RePoNet). Fig. 4 provides a first glance at our main result in a 2D embedding space. Specifically, the main evidence from Fig. 4(a) and 4(b) is that the features learned by RePoNet remain aligned with their corresponding fixed weights and maximally exploit the available representation space directly from the beginning of the training phase.

We apply our method to multiple vision datasets showing that it is possible to generate stationary and maximally separated features without affecting the generalization performance of DCNN

models and with a significant reduction in GPU memory usage at training time. A preliminary exploration of this work was presented in [95, 96].

3.5.2. Class-incremental Learning with Pre-allocated Fixed Classifiers

Natural intelligent systems learn incrementally by continuously receiving information over time. They learn new concepts adapting to changes in the environment by leveraging past experiences. A remarkable capability of these systems is that learning of new concepts is achieved while *not* forgetting previous ones. In contrast, current Deep Learning models, when updated with novel incoming data, suffer from *catastrophic forgetting*: the tendency of Neural Networks to completely and abruptly forget previously learned information [97, 98, 99]. This problem is related to the plasticity/stability dilemma in incremental learning [100]. Too much "plasticity" leads to catastrophic forgetting, too much "stability" leads to an inability to adapt to novel information. Continual Learning [101, 3] specifically addresses this problem, bringing machine learning closer to natural learning. In this learning scenario, the agent is presented with a stream of tasks and each new task is learned by reusing and combining the knowledge acquired while learning previous tasks. As the learning agent is processing a stream, it cannot store all examples seen in the past.

Continual learning has recently received increasing attention and several methods have been developed [102, 103, 104, 105, 106, 107, 108, 109]. However, despite the intense research efforts, the gap in performance with respect to offline multi-task learning makes continual learning an open problem. Most of the techniques have focused on a sequence of tasks in which both the identity of the task (task label) and boundaries between tasks are provided [110, 111, 112, 113]. Thus, many of these methods fail to capture real-world continual learning, with unknown task labels [114] [115]. A typical example that illustrates the difference between using or not the task labels is the Split MNIST experiment, in which the ten digits of the well known handwritten dataset are split into five classification tasks of two-class each. The model has five different final classification layers, one for each task. Those classifiers (i.e. output heads) are indexed by the task identity (1 to 5) that is given at testing time. This scenario is shown to be easier than class-incremental learning

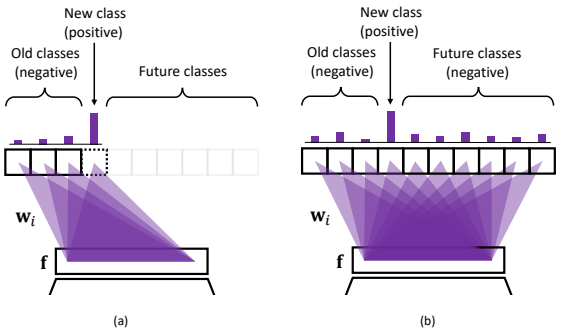


Figure 5. Class-incremental classifiers. (a)*: Expanding classifier.* (b)*: Pre-allocated classifier. The latter can exploit unseen future classes as negative examples.*

(CIL) since the selection of the output head is given by the task identity [114]. CIL is typically addressed with single-headed variants that do not require task identity, where the model always performs prediction over all classes (i.e. all digits 0 to 9) [116, 3, 117, 114, 115, 118].

In CIL the single head final layer of a Neural Network is *expanded* with an output node when a new class arrives (multiple new classes are expanded with multiple nodes); thus, in general, during

learning, an output node sees, according to the samples in the current random batch, positive and negative samples in the *newly arrived* class and in the *old seen classes* (i.e. the remaining), respectively (Fig. 5(a)).

In this paper, we address CIL using a novel classifier in which a number of pre-allocated output nodes are subject to the classification loss right from the beginning. This allows the output nodes of yet *unseen classes* to firstly see negative samples since the beginning of learning together with the positive samples that incrementally arrive (Fig. 5(b)). Contrarily to the *expanding* classifier, in our formulation, the output nodes can learn from the beginning of the learning phase. This is achieved by *pre-allocating* a special classifier with a large number of output nodes in which the weights are *fixed* (i.e. not undergoing learning) and set to values taken from the coordinate vertices of regular polytopes [119]. The classification layer so defined has two intriguing properties. The first is that the features do not change their geometric configuration as novel classes are incorporated in the learning model. The second is that a very large number of classes can be pre-allocated with no loss of accuracy. This allows the method to meet the underlying assumption of lifelong learning as for the case of the expanding classifier.
Further technical contents and evaluations are available in [77] .

3.5.3. Relevant publications

- Federico Pernici, Matteo Bruni, Claudio Baecchi, and Alberto Del Bimbo. Regular Polytope Networks, IEEE Transactions on Neural Networks and Learning Systems (2021) [78] https://zenodo.org/record/5045051
- Niccolò Biondi Federico Pernici, Matteo Bruni, and Alberto Del Bimbo. CoReS: learning Compatible Representations via Stationarity (Under Review).

3.5.4. Relevant software and/or external resources

- The Python implementation of our work "class-incremental learning with pre-allocated fixed classifiers: polytope networks" can be found in:

  https://github.com/DigiTurk84/class-incremental-polytope

3.5.5. Relevant WP8 Use Cases

Our tools, described in Sec. 3.5.1 and Sec. 3.5.2, contribute to use case 2 (2A and 2B) by providing the possibility to an AI search system to avoid extracting new features for all the previously seen images (i.e., the gallery-set) when updating the representation model.

4. Transfer learning (Task 3.3)

Transfer Learning is an emerging field among Deep Learning practitioners, that seeks to reuse and exploit previously generated models for different purposes. Considering the huge amount of data, human effort and computational power needed to train these models, being able to reuse them is of paramount importance.

Beyond practical reasons, Transfer Learning poses a scientific challenge of relevance, as it forces researchers to question the internal knowledge representation of deep models. Indeed, to understand how to reuse deep representations, one must first understand how are these representations learned, and how are they internally structured. Advances in this field has potential relevance for key aspects of Deep Learning, such as explainability and interpretability, efficiency and foot-print reduction, and real world deployment of AI powered systems.

4.1. Overview of our transfer learning contributions (Task 3.3)

Within this task partners are contributing in fundamental aspects of Transfer Learning, coordinately so that their advances can contribute to one another, and with the use cases of the project in mind. To further detail this collaboration, let us first summarize the contribution of partners.

In Subsection 4.2, UNITN studies how to train models on a set of data (the source) so that the internal representations generated can be more appropriate to solving not one, but a variety of additional tasks (the targets). This setting, named Multi-target Domain Adaptation can significantly increase the reusability of trained models, extending their transferability to more than one purpose. The complementary approach, looking for the combination of several source domains to solve a given task, is addressed by UNITN in 4.3.

In Subsection 4.4, CNR proposes Generalized Funnelling (GFun), a novel Heterogeneous transfer learning (HTL) method, and applies it to the problem of cross-lingual text classification. GFun generalizes Funnelling (Fun), a previously proposed HTL method based on hierarchical ensemble learning, whereby 1st-tier language-dependent classifiers return vectors of calibrated posterior probabilities, and where these vectors, being aligned across languages, are input to a language-agnostic metaclassifier that returns the final classification decision. In GFun, the 1st-tier classifiers are just special cases of view-generating functions (VGFs), that provide different views (e.g., in the form of different types of document embeddings that capture different types of correlation in the data) of the same document as input to the metaclassifier.

Finally, BSC contribution to this task is also directly related with the previously mentioned works. Although it is not reported in this deliverable due to its non-mature status, BSC's work will try to assess to which degree is it beneficial to use pre-trained representations as they are (also known as feature extraction approach within the Transfer Learning field), compared to their performance when adapting thoroughly for a task through deep net optimization procedures (also known as fine tuning within Transfer Learning). That is, a sort of Transfer Learning benchmark to find how much effort is worth putting into improving representations instead of simply reusing them. Since this approach allows us to compare the quality of deep net representations, the works of UNITN (Subsections 4.2 and 4.3) will qualify for assessment under BSC's Transfer Learning benchmark.

4.2. Multi-target domain adaptation (Task 3.3)

Contributing partners: UNITN

Deep learning models do not generalize well when deployed in the real-world. The gap in performance arises due to the difference in the distributions of the training (a.k.a source) and the test (a.k.a target) data, which is popularly referred to as *domain-shift* [120]. Since, collecting labeled data for every new operating environment is prohibitive, a rich line of research, called *Unsupervised Domain Adaptation (UDA)*, has evolved to tackle the task of leveraging the source data to learn a robust predictor on a desired target domain.

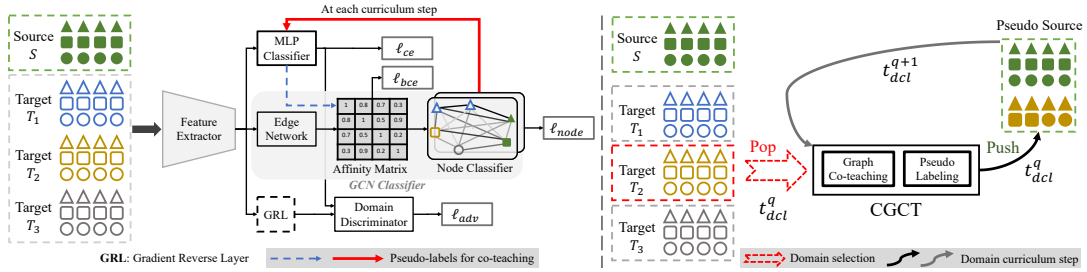
In the literature, UDA methods have predominantly been designed to adapt from a single source domain to a Single Target Domain (STDA), e.g., [121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131, 132, 133, 134], to name a few. However, given the proliferation in unlabeled data acquisition, the need to adapt to just a single target domain has lost traction in the real-world scenarios. As the number of target domains grows, the number of models that need to be trained also scales linearly. For this reason, the research focus has very recently been steered to address a more practical scenario of adapting simultaneously to multiple target domains from a single source domain. This adaptation setting is formally termed as *Multi-target Domain Adaptation (MTDA)*. The goal of the MTDA is to learn more compact representations with a single predictor that can perform well in all the target domains.

4.2.1. Curriculum graph co-teaching for multi-target domain adaptation

We build a framework for the MTDA pivoted around two key concepts: *feature aggregation* and *curriculum learning*. Firstly, as learning robust features in a unified space is a prerequisite for attaining minimum risk across multiple target domains, we propose to represent the source and the target samples as a graph and then leverage Graph Convolutional Networks (GCN) [135] to aggregate semantic information from similar samples in a *neighbourhood* across different domains. For the GCN to be operative, partial relationships among the samples (nodes) in the graph must at least be known a priori in the form of class labels. However, this information is absent for the target samples. To this end, we design a *co-teaching* framework where we train two classifiers: a MLP classifier and a GCN classifier that provide target pseudo-labels to each other. The MLP classifier is utilized to make the GCN learn the pairwise similarity between two nodes in the graphwhile the GCN classifier, due to its feature aggregation property, provides better pseudo-labels to assist the training of the MLP classifier. Given that co-teaching works on the assumption that different networks capture different aspects of learning [136], it is beneficial for suppressing noisy pseudo-labels. Secondly, during training as the network tries to adapt to multiple domain-shifts of varying degree, pseudo-labels obtained on-the-fly from the network for the target samples are very noisy. Self-training the network with unreliable pseudo-labeled target data further deteriorates the performance. To handle this, we propose to obtain pseudo-labels in an episodic fashion, and advocate the use of *curriculum learning* in the context of MTDA. In particular, when the domain labels of the target are latent, each episode or *curriculum step* consists of a fixed number of training iterations. Fairly consistent and reliable pseudo-labels are obtained from the GCN classifier at the end of each curriculum step. We call this proposed framework as Curriculum Graph Co-Teaching (CGCT) (see Fig. 6(a)).

Furthermore, when the domain labels of the target are available, we propose an Easy-To-Hard Domain Selection (EHDS) strategy where the feature alignment process begins with the target domain that is closest to the source and then gradually progresses towards the hardest one. This makes adaptation to multiple targets smoother. In this case, each curriculum step involves adaptation with a single new target domain. The CGCT when combined with this proposed Domain-aware Curriculum Learning (DCL) (see Fig. 6(b)) is referred to as D-CGCT.

Experimental analysis and results. We conduct experiments on five standard UDA benchmarks: Digits-five [137], Office-31 [138], PACS [139], Office-Home [140] and the very large scale



(a) Curriculum Graph Co-Teaching (b) Domain-aware Curriculum Learning

Figure 6. The pipeline of the proposed framework: a) CGCT: Curriculum Graph Co-Teaching and b) DCL: Domain-aware curriculum learning. (a) In the CGCT, the MLP Classifier provides pseudo-labels (PL) for the target samples to guide the Edge Network to learn the Affinity Matrix, whereas the Node Classifier of the GCN provides PL to the MLP Classifier at the end of each curriculum step, realizing the co-teaching. (b) In the DCL, the target domains are selected for adaptation, one at a time per domain curriculum step t^q_{dcl}, with the "easier" domains selected first and then the "harder" ones. After PL are obtained, the pseudo-labeled target dataset is added to the Pseudo Source dataset, which is then used in the next adaptation step.

DomainNet [141] (0.6 million images). In this document, we only report our results on DomainNet dataset (Table 6) and compare them with the state-of-the-art methods. The results on other datasets, implementation details and an extensive ablation study can be found in our paper given in Section 4.2.2. We use the classification accuracy to evaluate the performance. The classification accuracy is computed for every possible combination of one source domain and the rest of the target domains. The performance for a given direction, i.e., *source→rest*, is given by averaging the accuracy on all the target domains, where *source* signifies the source domain and *rest* indicates all the unlabeled domains except the *source*. Importantly, in all our experiments we always report the final classification accuracy obtained with the *MLP* because the *GCN* always requires a mini-batch at inference, an assumption which is easily violated when deployed in the real world.

Table 6. Comparison with the state-of-the-art methods on DomainNet. All methods use the ResNet-101 as the backbone. The classification accuracies are reported for each source→rest direction, with each source domain being indicated in the columns. All the reported numbers are evaluated on the multi-target setting.

	DomainNet						
Model	Cli.	Inf.	Pai.	Qui.	Rea.	Ske.	**Avg**(%)
Source train	25.6	16.8	25.8	9.2	20.6	22.3	20.1
SE [142]	21.3	8.5	14.5	13.8	16.0	19.7	15.6
MCD [143]	25.1	19.1	27.0	10.4	20.2	22.5	20.7
DADA [144]	26.1	20.0	26.5	12.9	20.7	22.8	21.5
CDAN [131]	31.6	27.1	31.8	12.5	33.2	35.8	28.7
MCC [145]	33.6	30.0	32.4	13.5	28.0	35.3	28.8
CDAN + DCL	35.1	31.4	37.0	**20.5**	35.4	**41.0**	33.4
CGCT	36.1	**33.3**	35.0	10.0	39.6	39.7	32.3
D-CGCT	**37.0**	32.2	**37.3**	19.3	**39.8**	40.8	**34.4**

As can be seen in the Table 6, the D-CGCT advances the state-of-the-art results for the very challenging DomainNet dataset by a non-trivial margin of 5.6%. This further verifies the effectiveness of our proposed methods for addressing the MTDA.

4.2.2. Relevant publication

- S. Roy, E. Krivosheev, Z. Zhong, N. Sebe, and E. Ricci, Curriculum Graph Co-Teaching for Multi-Target Domain Adaptation, CVPR 2021 [146]. https://zenodo.org/record/5014029

4.2.3. Relevant software and/or external resources

- The Pytorch implementation of our work "curriculum graph co-teaching for multi-target domain adaptation" can be found in https://github.com/Evgeneus/Graph-Domain-Adaptaion.

4.2.4. Relevant WP8 Use Cases

Our multi-target domain adaptation tool contribute to use cases (a) 2-2B by providing solutions to analyze/adapt the visual content, and (b) 2-2A and 2-2B by providing the discovery of new visual content and adapt accordingly. These can help to improve tagging and search capabilities.

4.3. Multi-source domain generalization (Task 3.3)

Contributing partners: UNITN

Domain Generalization (DG) is a promising solution that aims to learn generalizable models with one or several labeled source domains. DG does not require the access to target domains. Generally, DG can be divided into two categories, single-source DG [147, 148, 149] and multi-source DG [150, 151], according to the number of source domains. Recent works mainly focus on single-source DG where only one labeled source domain is available. However, a single domain provides limited training samples and scene information, restricting the improvement of single-source DG methods. In contrast, multi-source DG utilizes multiple datasets of different distributions, providing more training data that contain numerous variations and environmental factors. However, due to the strong compatibility of deep networks, directly aggregating all source domains together might lead the model to overfit on the domain bias, hampering the generalization ability of the model. Although we can sample balanced training data from all source domains during training to reduce the impact of domain bias, the above issue still remains.

4.3.1. Learning to generalize unseen domains

We address the multi-source DG by aiming to enforce the model to learn discriminative features without domain bias so that the model can be generalized to unseen domains. To achieve this goal, we introduce a meta-learning strategy for multi-source DG, which simulates the train-test process of DG during model optimization. We dynamically divide the source domains into meta-train and meta-test sets at each iteration. The meta-train is regarded as source data, and the meta-test is regarded as "unseen" data. During training, we encourage the loss of meta-train samples to optimize the model towards a direction that can simultaneously improve the accuracy of meta-test samples. To overcome the unstable meta-optimization caused by the parametric classifier, we propose a memory-based identification loss that is non-parametric and harmonizes with meta-learning. We also present a meta batch normalization layer (MetaBN) to diversify meta-test features, further establishing the advantage of meta-learning. The proposed framework is tested on person re-identification (ReID) problem. The illustration of the proposed method is given in Fig. 7 and the method is described in detail as follows.

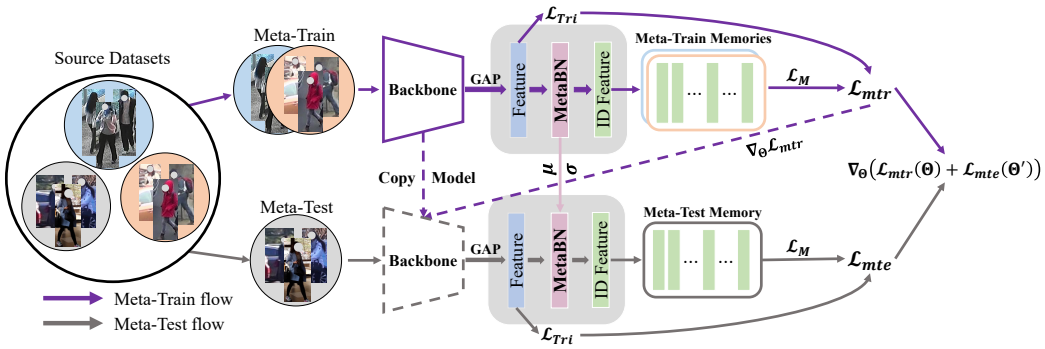


Figure 7. During training, we are given several (three in this example) source domains. At each iteration, source domains are divided into one meta-test and two meta-train domains. In the meta-train stage, memory-based identification loss and triplet loss are calculated from meta-train data as the meta-train loss. In the meta-test stage, the original model is copied and then the copied model is updated with meta-train loss. We compute the meta-test loss on the updated model. In this stage, MetaBN is used to diversify the meta-test features. Finally, the combination of meta-train and meta-test losses is used to optimize the original model.

For multi-source DG in person ReID, we are provided with N_S source domains $\mathcal{D}_S = \{\mathcal{D}_S^1, ..., \mathcal{D}_S^{N_S}\}$ in the training stage. The label spaces of the source domains are disjointed. The goal is to train a generalizable model with the source data. In the testing stage, the model is evaluated directly on a given unseen domain $\mathcal{D}_T$.

At each training iteration, we randomly divide N_S source domains into $N_S - 1$ domains as *meta-train* and the rest *one* domain as *meta-test*. The process of computing the meta-learning loss includes the meta-train and the meta-test stages. *In the meta-train stage*, we calculate the meta-train loss $\mathcal{L}_{mtr}$ on the meta-train samples to optimize the model. *In the meta-test stage*, the optimized model is used to calculate the meta-test loss $\mathcal{L}_{mte}$ with the meta-test samples. Finally, the network is optimized by the combination of meta-train and meta-test losses:

$$\underset{\Theta}{\operatorname{argmin}}\, \mathcal{L}_{mtr}(\Theta) + \mathcal{L}_{mte}(\Theta^{'}), \tag{20}$$

where Θ denotes the parameters of the network, and $\Theta^{'}$ denotes the parameters of the model optimized by the $\mathcal{L}_{mtr}$.

In multi-source DG of ReID, we typically have two kinds of parametric classifier selections, one global fully-connected layer (FC) classifier or N_S parallel FC classifiers for each domain, both of which will lead to problems during meta-learning. Herein, we propose a memory-based identification loss for multi-source DG, which is non-parametric and suitable for both meta-learning and person ReID. We maintain an individual memory for each source domain. For a source domain $\mathcal{D}_S^i$ with n_i identities, the memory $\mathcal{M}^i$ has n_i slots, where each slot saves the feature centroid of the corresponding identity. In initialization, we use the model to extract features for all samples of $\mathcal{D}_S^i$. Then, we initialize the centroid of each identity with a feature, which is averaged on the features of the corresponding identity. At each training iteration, we update the memory with the features in the current mini-batch. A centroid in the memory is updated through:

$$\mathcal{M}[k] \leftarrow m \cdot \mathcal{M}[k] + (1 - m) \cdot \frac{1}{|\mathcal{B}_k|} \sum_{x_i \in \mathcal{B}_k} f(x_i), \tag{21}$$

where $\mathcal{B}_k$ denotes the samples belonging to the kth identity and $|\mathcal{B}_k|$ denotes the number of samples for the kth identity in current mini-batch. $m \in [0, 1]$ controls the updating rate.

Given an embedding feature $f(x_i)$ from the forward propagation, we calculate the similarities between $f(x_i)$ and each centroid in the memory. The memory-based identification loss aims to classify $f(x_i)$ into its own identity, which is calculated by:

$$\mathcal{L}_M = -\log \frac{\exp\left(\mathcal{M}[i]^T f(x_i)/\tau\right)}{\sum_{k=1}^{n_i} \exp\left(\mathcal{M}[k]^T f(x_i)/\tau\right)}, \tag{22}$$

where τ is the temperature factor that controls the scale of distribution. We also use triplet loss [152] to train the model, which is formulated as,

$$\mathcal{L}_{Tri} = [d_p - d_n + \delta]_+, \tag{23}$$

where d_p is the Euclidean distance between an anchor feature and a hard positive feature, and d_n is the Euclidean distance between an anchor feature and a hard negative feature. δ is the margin of triplet loss and $[\cdot]_+$ refers to $max(\cdot, 0)$.

In our meta-learning strategy, the meta-test loss is important for learning generalizable representations, since the meta-test plays the role of the "unseen" domain. Intuitively, if the meta-test examples are sampled from more diverse distributions, the model will be optimized to be more robust to variations and thus be more generalizable to unseen domains. To achieve this goal, we introduce MetaBN to generate more diverse meta-test features at the feature-level. We replace the last Batch Normalization Layer (BN) [153] in the network with MetaBN. During training, MetaBN utilizes the domain information from meta-train domains to inject domain-specific information into meta-test features. This process can diversify meta-test features, enabling the model to simulate more feature variations.

In the meta-train stage, for the ith meta-train domain, MetaBN normalizes the meta-train features as the traditional BN, and saves the mini-batch mean μ_i and mini-batch variance σ_i, which are used in the following meta-test stage. In the meta-test stage, MetaBN uses the saved mean and variance to form $N_S - 1$ Gaussian Distributions. We sample features from these distributions and inject these domain-specific features into meta-test features such that for the ith distribution, we sample one feature z_j^i for each meta-test feature:

$$z_j^i \sim \mathcal{N}\left(\mu_i, \sigma_i\right), \tag{24}$$

where $\mathcal{N}$ denotes a Gaussian Distribution. By doing so, we obtain B (the batch size of meta-test features) sampled features, which are mixed with the original meta-test features for generating new features F_T^i,

$$F_T^i = \lambda F_T + (1-\lambda) Z^i, \tag{25}$$

where F_T denotes the original meta-test features. $Z^i = [z_0^i, z_1^i, \cdots, z_B^i]$ denotes B sampled features from the ith Gaussian Distribution. λ is the mixing coefficient, which is sampled from Beta Distribution, i.e.,, $\lambda \sim \text{Beta}(1, 1)$. Finally, the mixed features are normalized by batch normalization,

$$f_T^i = \gamma \frac{F_T^i - \mu_T^i}{\sqrt{{\sigma_T^i}^2 + \epsilon}} + \beta, \tag{26}$$

where μ_T^i and σ_T^i denote mini-batch mean and variance of F_T^i. γ and β denote the learnable parameters that scale and shift the normalized value.

During training, N_S source domains are separated into $N_S - 1$ meta-train domains and *one* meta-test domain at each iteration. The model is optimized by the losses calculated in the meta-train and meta-test stages. For each meta-train domain, the meta-train loss is a combination of memory-based identification (Eq. 22) and triplet losses (Eq. 23), i.e.,

$$\mathcal{L}_{mtr}^i = \mathcal{L}_{Tri}(X_S^i; \Theta) + \mathcal{L}_M(X_S^i, \mathcal{M}_S^i; \Theta), \tag{27}$$

where Θ denotes the parameters of the network. X_S^i and $\mathcal{M}_S^i$ denote the training samples and memory of the ith meta-train domain, respectively. The total loss for meta-train is averaged over $N_S - 1$ meta-train domains, formulated as,

$$\mathcal{L}_{mtr} = \frac{1}{N_S - 1} \sum_{i=0}^{N_S - 1} \mathcal{L}_{mtr}^i. \tag{28}$$

In the meta-test stage, the meta-test domain is performed on the new parameters $\Theta^{'}$, which is obtained by optimizing Θ with $\mathcal{L}_{mtr}$. With the MetaBN, we can obtain $N_S - 1$ mixed features for each meta-test sample. The average memory-based identification loss over these features is considered as the meta-test memory-based identification loss. The meta-test loss is:

$$\mathcal{L}_{mte} = \mathcal{L}_{Tri}(X_T; \Theta^{'}) + \frac{1}{N_S - 1} \sum_{k=0}^{N_S - 1} \mathcal{L}_M(f_T^k, \mathcal{M}_T; \Theta^{'}), \tag{29}$$

where X_T denotes the meta-test samples and f_T^k denotes the kth mixed features generated by the MetaBN. Finally, the model is optimized by the objective in Eq. 20.

Experimental Analysis and Results. We conduct experiments on four large-scale person re-identification benchmarks: Market-1501 [154], DukeMTMC-reID [155, 156], CUHK03 [157, 158] and MSMT17 [159]. We divide these four datasets into two parts: three domains as source domains for training and the other one as target domain for testing. The cumulative matching characteristic (CMC) at Rank-1 and mean average precision (mAP) are used to evaluate performance on the target testing set.

As seen in Table 7, when using Combined MSMT17 as the source data, OSNet-AIN [149] and QAConv [148] achieve the best results on both Market-1501 and DukeMTMC-reID. Compared to single-source DG methods that use more training data (Combined MSMT17), our method outperforms them by a large margin on Market-1501 and achieves comparable results with them on DukeMTMC-reID. Specifically, when testing on Market-1501, with the same backbone, our method surpasses SNR [147] by 6.7% in mAP and 4.4% in Rank-1 accuracy. When training with multiple source domains, with the same backbone, our method produces significantly higher results than QAConv$_{50}$. Specifically, proposed method is higher than QAConv$_{50}$ by 12.5% in mAP for Market-1501 and by 3.4% in mAP for DukeMTMC-reID. This demonstrates the superiority of our method over the method that considers all the source domains as one domain. When using the IBN-Net50 as the backbone, our method can achieve better mAP than using ResNet-50. There is only one method (QAConv [148]) evaluated on CUHK03 and MSMT17. When testing on MSMT17, QAConv [148] uses DukeMTMC-reID as the source data. Clearly, our method achieves higher results than QAConv [148] on both datasets, no matter how many source domains QAConv is trained with. We also find that both our method and QAConv produce poor results on CUHK03 and MSMT17, indicating there is still a large room for generalizable models in DG.

4.3.2. Relevant publications

- Y. Zhao, Z. Zhong, F. Yang, Z. Luo, Y. Lin, S. Li, and N. Sebe, Learning to Generalize Unseen Domains via Memory-based Multi-Source Meta-Learning for Person Re-Identification, CVPR 2021 [161]. https://zenodo.org/record/5014450

4.3.3. Relevant software and/or external resources

- The Python implementations of our work "learning to generalize unseen domains via memory-based multi-source meta-learning for person re-identification" can be found in https://

Table 7. Comparison with State-of-the-Arts domain generalization methods on four large-scale person ReID benchmarks — Market-1501 (M), DukeMTMC-reID (D), CUHK03 (C) and MSMT17 (MS). The performance is evaluated quantitatively by mean average precision (mAP) and cumulative matching characteristic (CMC) at Rank-1 (R1).

Method	Source	IDs	Images	Market-1501 mAP	Market-1501 R1	Source	IDs	Images	DukeMTMC mAP	DukeMTMC R1
OSNet-IBN [160]	Com-MS	4,101	126,441	37.2	66.5	Com-MS	4,101	126,441	45.6	67.4
OSNet-AIN [149]				43.3	70.1				52.7	71.1
SNR [147]				41.4	70.1				50.0	69.2
QAConv$_{50}$ [148]				43.1	72.6				52.6	69.4
QAConv$_{50}$ [148]	MS+D+C	3,110	75,406	35.6	65.7	MS+M+C	3,159	71,820	47.1	66.1
Ours (ResNet-50)				48.1	74.5				50.5	**69.4**
Ours (IBN-Net50)				**50.2**	**75.9**				**51.1**	69.2
QAConv$_{50}$ [148]	MS+D +C-NP	2,510	56,508	39.5	68.6	MS+M +C-NP	2,559	52,922	43.4	64.9
Ours (ResNet-50)				51.1	76.5				48.2	67.1
Ours (IBN-Net50)				**52.5**	**78.3**				**48.8**	**67.2**

Method	Source	IDs	Images	CUHK-NP mAP	CUHK-NP R1	Source	IDs	Images	MSMT17 mAP	MSMT17 R1
QAConv$_{50}$ [148]	Com-MS	4,101	126,441	22.6	25.3	D	702	16,522	8.9	29.0
QAConv$_{50}$ [148]	MS+D+M	2,494	62,079	21.0	23.5	D+M+C	2,820	55,748	7.5	24.3
Ours (ResNet-50)				29.9	30.7				12.9	33.0
Ours (IBN-Net50)				**32.1**	**33.1**				**14.7**	**36.9**
QAConv$_{50}$ [148]	MS+D+M	2,494	62,079	19.2	22.9	D+M +C-NP	2,220	36,823	10.0	29.9
Ours (ResNet-50)				30.9	**31.9**				13.1	32.0
Ours (IBN-Net50)				**31.4**	31.6				**15.4**	**37.1**

github.com/HeliosZhao/M3L.

4.3.4. Relevant WP8 Use Cases

Our multi-source domain generalization tool contribute to use cases (a) 2-2B by providing solutions to analyze the visual content thanks to being able to generalize under domain-gap.

4.4. Heterogeneous Document Embeddings for Cross-Lingual Text Classification (Task 3.3)

Contributing partners: CNR

In transfer learning (TL), given a training set $\mathrm{Tr}_{\mathcal{S}}^{L}$ of labelled data items from a "source" domain $\mathcal{S}$, we must issue predictions for unlabelled data items from a "target" domain $\mathcal{T}$, related to $\mathcal{S}$ but different from it. *Heterogeneous* TL (HTL) denotes the set of TL tasks in which the feature spaces $F_{\mathcal{S}}$ and $F_{\mathcal{T}}$ of the two domains are different (and, in general, non-overlapping). An example HTL task is *cross-lingual text classification (CLC)*, the task of classifying documents, each written in one of a finite set $\mathcal{L} = \{\lambda_1, ..., \lambda_{|\mathcal{L}|}\}$ of languages, according to a common "codeframe" (or: classification scheme) $\mathcal{Y} = \{y_1, ..., y_{|\mathcal{Y}|}\}$. CLC can be tackled as a TL task, with the goal of improving on the naïve "monolingual baseline" (consisting of $|\mathcal{L}|$ independently generated monolingual classifiers) by exploiting synergies among training sets from different languages. Here, each language-specific domain of documents is at the same time a source domain and a target domain, according to an "all languages help each other" metaphor.

Esuli et al. [162] proposed *Funnelling* (FUN), a two-tier ensemble method for HTL, and tested it on a CLC setting. In FUN, a set of $|\mathcal{L}|$ 1st-tier, language-specific classifiers return, for each unlabelled document d, a vector of $|\mathcal{Y}|$ calibrated posterior probabilities; each such vector is fed to a 2nd-tier "metaclassifier" which returns the final classification decisions. Vectors of $|\mathcal{Y}|$ calibrated posterior probabilities thus form an "interlingua" among the $|\mathcal{L}|$ languages, since all such vectors are in the same vector space, irrespectively of the language of the documents they correspond to.

One of the reasons FUN outperforms the naïve monolingual baseline is that the metaclassifier leverages *class-class correlations*, i.e., stochastic dependencies among the different classes in $\mathcal{Y}$. In this work we propose *Generalized Funnelling* (GFUN), an extension of FUN capable of leveraging additional types of correlations (e.g., word-class correlations, word-word correlations). This is obtained by aggregating the vector of calibrated posterior probabilities and document embeddings that encode these additional types of correlations. We here present CLC experiments in which we extend the vectors of calibrated posterior probabilities by using *Word-Class Embeddings (WCEs)* [163] and *Multilingual Unsupervised or Supervised Embeddings* (MUSEs) [164], which encode word-class correlations and word-word correlations for multiple languages, respectively.

4.4.1. Generalized Funnelling

Funnelling, as described in [162], comes in two variants, called FUN(KFCV) and FUN(TAT); we here disregard FUN(KFCV) and only use FUN(TAT), since in all the experiments reported in [162], FUN(TAT) clearly outperformed FUN(KFCV). Both FUN and GFUN can tackle single-label and multi-label text classification alike; for reasons of space, we here deal only with the latter.

In FUN(TAT), in order to train a classifier ensemble, we first train language-specific, 1st-tier classifiers $h_1^1, ..., h_{|\mathcal{L}|}^1$ (with superscript s indicating the s-th tier) from the language-specific training sets $\text{Tr}_1, ..., \text{Tr}_{|\mathcal{L}|}$. Training documents $d \in \text{Tr}_i$ may be represented by means of any desired vectorial representation $\phi^1(d)$, such as, e.g., *tfidf*-weighted bag-of-words, and classifiers may be trained by any learner, provided the resulting classifier returns, for each document d to classify and for each class y_j, a confidence score $h_i^1(d, y_j) \in \mathbb{R}$, where λ_i is the language document d is written in.

We then apply each 1st-tier classifier h_i^1 to all training documents $d \in \text{Tr}_i$, thus obtaining a vector

$$S(d) = (h_i^1(d, y_1), ..., h_i^1(d, y_{|\mathcal{Y}|})) \tag{30}$$

of confidence scores for each $d \in \text{Tr}_i$.

The next step consists of computing (via a chosen probability calibration method) language- and class-specific calibration functions f_{ij} that map confidence scores $h_i^1(d, y_j)$ into calibrated posterior probabilities. We can then apply f_{ij} to each document $d \in \text{Tr}_i$ and obtain a vector of calibrated posterior probabilities

$$\begin{aligned} \phi^2(d) &= (f_{i1}(h_i^1(d, y_1)), ..., f_{i|\mathcal{Y}|}(h_i^1(d, y_{|\mathcal{Y}|}))) \\ &= (\Pr(y_1|d), ..., \Pr(y_{|\mathcal{Y}|}|d)) \end{aligned} \tag{31}$$

At this point, we train a language-independent, 2nd-tier "meta"-classifier h^2 from all training documents $d \in \bigcup_{i=1}^{|\mathcal{L}|} \text{Tr}_i$, where document d is represented by its $\phi^2(d)$ vector. This concludes the training phase.

In order to apply the trained ensemble to a test document $d \in \text{Te}_i$ we apply classifier h_i^1 to d and convert the resulting vector $S(d)$ of confidence scores into a vector $\phi^2(d)$ of calibrated posterior probabilities. We then feed this latter into the metaclassifier h^2, which returns a vector of confidence scores $(h^2(d, y_1), ..., h^2(d, y_{|\mathcal{Y}|}))$ from which the final decisions are obtained in the usual way.

As explained in [162], the reasons of the good performance of FUN are essentially two. The first is that FUN learns from heterogeneous data; i.e., while in the naïve monolingual baseline each classifier is trained on just $|\text{Tr}_i|$ labelled examples, in FUN we have a metaclassifier trained on $\bigcup_{i=1}^{|\mathcal{L}|} |\text{Tr}_i|$ labelled examples, which means that all training examples contribute to classifying all unlabelled examples, irrespectively of the languages of the former and of the latter. The second is that the metaclassifier leverages *class-class correlations*, i.e., it learns to exploit the stochastic dependencies between classes typical of multilabel settings.

The goal of GFUN is that of allowing additional types of stochastic dependencies (e.g., word-class correlations, word-word correlations) to contribute to the classification process.

The key step in allowing FUN's metaclassifier to leverage the different language-specific training sets consists of representing their documents in a space that is common to all languages. In FUN, this is made possible by the fact that the 1st-tier classifiers all return vectors of calibrated posterior probabilities. In GFUN this process is generalized by introducing a set Ψ of *view generators*, i.e., language-dependent functions mapping documents into language-independent vectorial representations *aligned across languages*, i.e., such that both the dimensionality of the vectors and the meaning of each vector dimension are the same for all languages.
The view generators $\psi_k \in \Psi$ might require parameter optimization during the training phase; this is undertaken independently for each language and view generator. GFUN also implements an aggregation function (*aggfunc*) that brings together the different representations produced by the view generators, and shapes the document representation for use in the metaclassifier. In this case, as *aggfunc* we simply adopt concatenation.

Note that the original formulation of FUN thus reduces to an instantiation of GFUN in which there is a single view generator (a calibrated classifier) and the aggregation function is the identity function. In this case, fitting this single view generator comes down to training the 1st-tier classifier h_i^1 and choosing the calibration functions f_{ik}. During the test phase, invoking the view generator amounts to computing the $\phi^2(d)$ representations (Eq. 31) of the test documents.

The FUN metaclassifier has access to vectors of $|\mathcal{Y}|$ posterior probabilities, and can thus leverage class-class correlations. In what follows we instead describe new view generators that we have investigated in order to introduce additional information into GFUN. In particular, we describe view generators that mine word-class correlations and word-word correlations. We also discuss a few additional modifications concerning data normalization that we have introduced into GFUN and that, although subtle, bring about a substantial improvement in the effectiveness of the method.

For encoding **word-class correlations** we derive document embeddings from *Word-Class Embeddings* (WCEs) [163]. WCEs are supervised embeddings meant to extend (e.g., by concatenation) other unsupervised pre-trained word embeddings (e.g., those produced by `word2vec` or `GloVe`) in order to inject task-specific word meaning in multiclass text classification. The WCE for word w is defined as

$$E(w) = \varphi(\eta(w, y_1), ..., \eta(w, y_{|\mathcal{Y}|})) \tag{32}$$

where η is a real-valued function that quantifies the correlation between word w and class y_j as observed in the training set, and where φ is any dimensionality reduction function. Here, as the η function we adopt the normalized dot product, as proposed in [163], whose computation is very efficient; as φ we use the identity function, and our WCEs are thus $|\mathcal{Y}|$-dimensional vectors.

So far, WCEs have been tested exclusively in monolingual settings. However, WCEs are *naturally aligned across languages*, since WCEs have one dimension for each $y \in \mathcal{Y}$, which is the same for all $\lambda_i \in \mathcal{L}$. Document embeddings relying on WCEs thus display similar characteristics irrespective of the language in which the document is written in. This is, to the best of our knowledge, the first application of WCEs to a multilingual setting.

The view generator for WCEs consists of first computing, for each language $\lambda_i \in \mathcal{L}$, the language-specific WCE matrix $\mathbf{W}_i$, and then projecting the *tfidf* matrix $\mathbf{X}_i$ of Tr_i (during training) or Te_i (during test) as $\mathbf{X}_i \cdot \mathbf{W}_i$.

For encoding **word-word correlations** we derive document embeddings from *Multilingual Unsupervised or Supervised Embeddings* (MUSEs) [164]. MUSEs are generated via a method for aligning in a common vector space unsupervised (monolingual) word embeddings. The alignment is obtained via a linear mapping (i.e., a rotation matrix) W learned by an adversarial training process in which a *generator* (in charge of mapping the source embeddings onto the target space) is trained to fool a *discriminator* from distinguishing the language of provenance of the embeddings, that is, from discerning if the embeddings it receives as input originate from the target language or are instead the product of a transformation of embeddings originated from the source language. Mapping W is then further refined using Procrustes alignment. The name "Unsupervised or Supervised" refers to the fact that the method can operate with or without a dictionary of parallel seed words.

We used the MUSEs that the authors of [164] make publicly available[1], and that consist of 300-dimensional multilingual word embeddings trained on Wikipedia using fastText. The embeddings have been aligned for 30 languages with the aid of a bilingual dictionary.

The view generator for MUSEs is similar to that for WCEs, with the sole exception that fitting the generator comes down to just allocating in memory the pre-trained MUSE matrices $\mathbf{M}_i$ for each language λ_i involved; the projection of training and test documents is as before, and is computed as $\mathbf{X}_i \cdot \mathbf{M}_i$.

We have found that applying some routine **normalization** techniques consistently increases the performance of gFun. This normalization consists of imposing unit L2-norm to the vectors computed by the view generators, removing (following [165]) the first principal component of the document embeddings obtained via WCEs or MUSEs, and standardizing the columns of the shared space before passing the vectors to the metaclassifier.[2]

The intuition behind normalization, when dealing with heterogeneous representations, is straightforward, and is that of allowing all sources of information to equally contribute to the classification process. What instead might come as a surprise is the fact that normalization helps improve gFun even when relying exclusively on the class-class correlations (i.e., as Fun does [162]), and that this improvement is statistically significant. We quantify this variation in performance in the following experiments.

4.4.2. Experiments

We here summarize the results of the experiments that we have carried out (see [166] for details). For different variants of gFun we indicate in parentheses the document representations that the variant uses, with the vectors of calibrated posterior probabilities denoted by X, and with document embeddings obtained via MUSEs and WCEs denoted by M and W, resp. gFun(X), the variant that uses the same document representation as Fun, outperforms Fun, which indicates that the normalization steps are beneficial. The results of gFun(XM) and gFun(XW) show that the M and W representations contribute differently, depending on the nature of the dataset: on RCV1/RCV2, adding M delivers better results than adding W, while W is more useful than M on JRC-Acquis. This can be ascribed to the higher number of classes that JRC-Acquis (300) has with respect to RCV1/RCV2 (73): the 300 classes of JRC-Acquis enable WCEs (that encode word-class

[1] https://github.com/facebookresearch/MUSE

[2] Standardizing (a.k.a. "z-scoring", or "z-transforming") consists of having a random variable x, with mean μ and standard deviation σ, translated and scaled as $z = \frac{x-\mu}{\sigma}$, so that the new random variable z has zero mean and unit variance. The statistics μ and σ are unknown, and are thus estimated on the training set.

correlations) to bring in a higher amount of information, thus making WCEs more discriminative for JRC-Acquis than for RCV1/RCV2. Using all three representations, as in gFun(XMW), yields the best result in 3 out of 4 (measure, dataset) combinations, and in the 4th combination yields a result not different, in a statistically significant sense, from the best one; this confirms the value of the intuitions that underlie gFun.

4.4.3. Relevant publications

- Alejandro Moreo, Andrea Pedrotti, and Fabrizio Sebastiani. Heterogeneous Document Embeddings for Cross-Lingual Text Classification. Proceedings of the 36th ACM Symposium On Applied Computing (SAC 2021), Gwangju, KR, pp. 685–688. (Best short paper award) [166].
  Zenodo record: https://zenodo.org/record/4467989

4.4.4. Relevant software and/or external resources

- The Python implementation of the work "Heterogeneous document embeddings for cross-lingual text classification" can be found at https://github.com/andreapdr/gFun.

5. Learning to count (Task 3.7)

Contributing partners: CNR

"Learning to Count" is a task having to do with machine learning approaches for training estimators of quantities. There are two classes of problems that are being addresses in this task, and that may be usefully viewed as forming two different subtasks, i.e.,

- "Learning to quantify" (a.k.a. *quantification*). This subtask is concerned with training unbiased estimators of class prevalence via supervised learning, i.e., learning to estimate, given a sample of objects, the percentage of items that belong to a given class. This task originates with the observation that "Classify and Count (CC)", the trivial method of obtaining class prevalence estimates, is often a biased estimator, and thus delivers suboptimal quantification accuracy. This bias is particularly strong when the data exhibits *dataset shift*, i.e., when the joint distribution of the independent and the dependent variables is not the same in the training data and in the unlabelled data for which predictions must be issued. Quantification is important for several applications, e.g., gauging the collective satisfaction for a certain product from textual comments, establishing the popularity of a given political candidate from blog posts, predicting the amount of consensus for a given governmental policy from tweets, or predicting the amount of readers who will find a product review helpful.
- "Learning to count objects". This subtask has to do with using machine learning approaches in order to train estimators of the number of objects (which may be inanimate objects, such as cars, but may also be animate objects, such as people or animals) in visual media, such as still images or video frames. Example applications of these techniques are e.g., counting the number of cars in a video frame (in order to estimate traffic volume or car park occupancy) or counting the number of people in a still image (say, in order to estimate the amount of people taking part in a rally).

5.1. Overview of our learning to count contributions (Task 3.7)

We here present two main contributions.

In Section 5.2 CNR casts a critical eye on much of the experiments that have been carried out in the literature on learning to quantify, and that have contributed to assess the relative strengths of different methods, including the "Classify and Count" trivial baseline. In this research, the authors argue that much of this experimentation has been inadequate, in that parameter optimisation has often been carried out by minimizing classification-oriented loss functions and by using simplistic classification-oriented evaluation protocols. The authors of this research argue that this parameter optimisation should instead have been carried out by minimizing truly quantification-oriented loss functions, and propose a much more complex quantification-oriented evaluation protocol that reflects the parameter optimisation needs of learning to quantify. The authors of this research go on to reassess a number of previously proposed quantification methods by using the parameter optimisation standards they propose, and conclude that many well-known methods now look different, in terms of performance, than they previously appeared.

In Section 5.3 CNR presents the activity it has carried out on investigating techniques for counting objects in images. Specifically, a technique relying on density map estimation and unsupervised domain adaptation is introduced. This activity also has relationships with T3.3 (Transfer Learning) and T5.3 (Learning with scarce data). Its details are given in Deliverable D5.1 ("Initial report on Multimedia Summarisation and Analysis") where the activity carried out in T5.3 is presented as well.

5.2. Re-Assessing Quantification Methods with Quantification-Oriented Parameter Optimisation (Task 3.7)

Learning to quantify (a.k.a. quantification) consists of training a predictor that returns estimates of the relative frequency (a.k.a. *prevalence*, or *prior probability*) of the classes of interest in a set of unlabelled data items, where the predictor has been trained on a set of labelled data items [167].

The rationale of this task is that many real-life applications of classification suffer from *distribution shift* [168], the phenomenon according to which the distribution $p_y(U)$ of the labels in the set of unlabelled test documents U is different from the distribution $p_y(L)$ that the labels have in the set of labelled training documents L. It has been shown that, in the presence of distribution shift, the trivial strategy of using a standard classifier to classify all the unlabelled documents in U and counting the documents that have been assigned to each class (the "Classify and Count" (CC) method), delivers poor class prevalence estimates. The reason is that most supervised learning methods are based on the IID assumption, which implies that the distribution of the labels is the same in L and U. "Classify and Count" is considered a *biased estimator* of class prevalence, since the goal of standard classifiers is to minimise (assuming for simplicity a binary setting) *classification* error measures such as $(\text{FP} + \text{FN})$, while the goal of a quantifier is to minimise *quantification* error measures such as $|\text{FP} - \text{FN}|$. (In this work we tackle binary quantification, so FP and FN denote the numbers of false positives and false negatives, resp., from a binary contingency table.) Following this observation, several quantification methods have been proposed, and have been experimentally shown to outperform CC.

In this work we contend that previous works, when testing advanced quantification methods, have used as baselines versions of CC that had not been properly optimised. This means that published results on the relative merits of CC and other supposedly more advanced methods are still unreliable. We thus reassess the real merits of CC by running extensive experiments (on three publicly available sentiment classification datasets) in which we compare properly optimised versions of CC and its three main variants (PCC, ACC, PACC) with a number of more advanced quantification methods. In these experiments we properly optimise all quantification methods, i.e., (a) we optimise their hyperparameters, and (b) we conduct this optimisation via a truly quantification-oriented evaluation protocol, which also involves minimising a quantification loss rather than a classification loss. Our results indicate that, while still inferior to some cutting-edge quantification methods, CC and its variants deliver near-state-of-the-art quantification accuracy once hyperparameter optimisation is performed properly.

We here assume a binary setting, with the two classes $\mathcal{Y} = \{\oplus, \ominus\}$ standing for Positive and Negative. By $\mathbf{x}$ we denote a document drawn from a domain $\mathcal{X}$ of documents; by $L \subset \mathcal{X}$ we denote a set of labelled documents, that we typically use as a training set, while by U we denote a sample of unlabelled documents, that we typically use as the sample to quantify on. By $p_y(\sigma)$ we indicate the true prevalence of class y in sample σ, by $\hat{p}_y(\sigma)$ we indicate an estimate of this prevalence, and by $\hat{p}_y^M(\sigma)$ we indicate the estimate of this prevalence as obtained via quantification method M. Of course, for any method M it holds that $\hat{p}_\ominus^M(U) = (1 - \hat{p}_\oplus^M(U))$.

Unsuitable parameter optimisation and weak baselines The reason why we here reassess CC and its variants we have described above, is that we believe that, in previous papers where these methods have been used as baselines, their full potential has not been realised because of *missing or unsuitable optimisation* of the hyperparameters of the classifier on which the method is based.

Specifically, both CC and its variants rely on the output of a previously trained classifier, and this output usually depends on some hyperparameters. Not only the quality of this output heavily depends on whether these hyperparameters have been optimised or not (on some held-out data

or via k-fold cross-validation), but *it also depends on what evaluation measure this optimisation has used as a criterion for model selection*. In other words, given that hyperparameter optimisation chooses the value of the parameter that minimises error, it would make sense that, for a classifier to be used for quantification purposes, "error" is measured via a function that evaluates *quantification* error, and not classification error. Unfortunately, in most previous quantification papers, researchers either do not specify whether hyperparameter optimisation was performed at all [169, 170, 171, 172, 173, 174, 175, 176], or leave the hyperparameters at their default values [177, 178, 179, 180, 181], or do not specify which evaluation measure they use in hyperparameter optimisation [182, 183], or use, for this optimisation, a classification-based loss [184, 185]. In retrospect, we too plead guilty, since some of the papers quoted here are our own.

All this means that CC and their variants, when used as baselines, have been turned into *weak* baselines, and this means that the merits of more modern methods relative to them have possibly been exaggerated, and are thus yet to be assessed reliably. In this work we thus engage in a reproducibility study, and present results from text quantification experiments in which, contrary to the situations described in the paragraph above, we compare *carefully optimised* versions of CC and its variants with a number of (*carefully optimised* versions of) more modern quantification methods, in an attempt to assess the relative value of each in a robust way.

Quantification-oriented parameter optimisation In order to perform quantification-oriented parameter optimisation we need to be aware that there may exist two types of parameters that require estimation and/or optimisation, i.e., (a) the hyperparameters of the classifier on which the quantification method is based, and (b) the parameters of the quantification method itself.

The way we perform hyperparameter optimisation is the following. We assume that the dataset comes with a predefined split between a training set L and a test set U.

We first partition L into a part L_{Tr} that will be used for training purposes and a part L_{Va} that will be used as a held-out validation set for optimising the hyperparameters of the quantifier. We then extract, from the validation set L_{Va}, several random validation samples, each characterised by a predefined prevalence of the $\oplus$ class; here, our goal is allowing the validation to be conducted on a variety of scenarios characterised by widely different values of class prevalence, and, as a consequence, by widely different amounts of distribution shift.[3] In order to do this, we extract each validation sample σ by randomly undersampling one or both classes in L_{Va}, in order to obtain a sample with prespecified class prevalence values. We draw samples with a desired prevalence value and a fixed amount q of documents; in order to achieve this, in some cases only one class needs to be undersampled while in some other cases this needs to happen for both classes. We use random sampling without replacement if the number of available examples of $\oplus$ (resp. $\ominus$) is greater or equal to the number of required ones, and with replacement otherwise. We extract samples with a prevalence of the $\oplus$ class in the set $\{\pi_1, ..., \pi_n\}$; for each of these n values we generate m random samples consisting of q validation documents each. Let Θ be the set of hyperparameters that we are going to optimise. Given the established grid of value combinations $\theta_1, ..., \theta_n$ that we are going to test for Θ, for each θ_i we do the following, depending on whether the quantification method has its own parameters (Case 1 below) or not (Case 2 below):

1. If the quantification method M we are going to optimise requires some parameters λ_i to be estimated, we first split L_{Tr} into a part $L_{\text{Tr}}^{\text{Tr}}$ and a part $L_{\text{Tr}}^{\text{Va}}$, training the classifier on $L_{\text{Tr}}^{\text{Tr}}$

[3] Note that this is similar to what we do, say, in classification, where the different hyperparameter values are tested on many validation documents; here we test these hyperparameter values on many validation *samples*, since the objects of study of text quantification are document samples inasmuch as the objects of study of text classification are individual documents.

using the chosen learner parameterised with θ_i, and estimate parameters λ_i on $L_{\text{Tr}}^{\text{Va}}$.[4] Among the variants of CC, this applies to methods ACC and PACC, which require the estimation of (the hard or soft version of) TPR and FPR. Other methods used in the experiments of Section 5.2 and that also require some parameter to be estimated are HDy and QuaNet.

2. If the quantification method M we are going to optimise does not have any parameter that requires estimation, then we train our classifier on L_{Tr}, using the chosen learner parameterised with θ_i, and use quantification method M on all the samples extracted from L_{Va}.

In both cases, we measure the quantification error via an evaluation measure for quantification that combines (e.g., averages) the results across all the validation samples. As our final value combination for hyperparameter set Θ we choose the θ_i for which quantification error is minimum.

Note that, in the above discussion, each time we split a labelled set into a training set and a validation set for parameter estimation / optimisation purposes, we could instead perform a k-fold cross-validation; the parameter estimation/optimisation would be more robust, but the computational cost of the entire process would be k times higher. While the latter method is also, from a methodological standpoint, an option, in this work we stick to the former method, since the entire parameter optimisation process is, from a computational point of view, already very expensive.

Experiments The experiments we have carried out (see [186] for details) reveal a number of patterns. One of these is that SVM and LR (the two best-performing classifiers overall) tend to benefit from optimised hyperparameters, and tend to do so to a greater extent when the loss used in the optimisation is quantification-oriented. Somehow surprisingly, not all methods improve after model selection in every case. However, there tends to be such an improvement especially for ACC and PACC. A likely reason for this is the possible existence of a complex tradeoff between obtaining a more accurate classifier and obtaining more reliable estimates for the TPR and FPR quantities.

Regarding the different datasets, it seems that there is no clear improvement from performing model selection when the training set is balanced (see IMDB), neither by using a classification-oriented measures nor by using a quantification-oriented one. A possible reason is that any classifier (with or without hyperparameter optimisation) becomes a reasonable quantifier if it learns to pay equal importance to positive and negative examples, i.e., if the errors it produces are unbiased towards either $\oplus$ or $\ominus$. In this respect, RF and MNB prove strongly biased towards the majority class, and only when corrected via an adjustment (ACC or PACC) they deliver results comparable to those obtained for other learners.

Interestingly, although some advanced quantification methods (specifically: SLD and HDy) stand as the top performers, many among the (supposedly more sophisticated) quantification methods fail to improve over CC's performance. At a glance, most quantification methods tend to obtain lower ranks when compared with properly optimised CC variants. Remarkable examples of rank variation include CC and ACC with SVM and LR: when evaluated on KINDLE and HP, they climb several positions (up to 25), often entering the group of the 10 top-performing methods. In the most extreme case, $\text{ACC}_{\text{LR}}^{\text{AE}}$ moves from position 28 (out of 29) to position 3 once properly optimised for quantification.

See [186] for a much expanded discussion of this work and of the experiments we have carried out.

[4] Note that we do *not* retrain the classifier on the entire L_{Tr}. While this might seem beneficial, since L_{Tr} contains more training data than $L_{\text{Tr}}^{\text{Tr}}$, we need to consider that the estimates $\hat{\text{TPR}}_h$ and $\hat{\text{FPR}}_h$ have been computed on L_{Tr} and not on $L_{\text{Tr}}^{\text{Tr}}$.

5.2.1. Relevant publications

- Alejandro Moreo and Fabrizio Sebastiani. Re-Assessing the "Classify and Count" Quantification Method. Proceedings of the 43rd European Conference on Information Retrieval (ECIR 2021) pp. 75–91, vol. 2 [186].
  Zenodo record: https://zenodo.org/record/4468277

5.2.2. Relevant software and/or external resources

- The Python implementation of the work "Re-assessing the "classify and count" quantification method" can be found at https://github.com/AlexMoreo/CC.

5.3. Counting objects by leveraging domain adaptation techniques (Task 3.7)

Contributing partners: CNR

We developed an end-to-end CNN-based Unsupervised Domain Adaptation (UDA) algorithm for traffic density estimation and counting, based on adversarial learning in the output space. The density estimation is one of those tasks requiring per-pixel annotated labels and, therefore, needs a lot of human effort.

In many real-world applications, there is indeed a large domain shift between the distributions of the train source and test target domains, leading to a significant drop in performance at inference time. UDA is a class of techniques that aims to mitigate this drawback without the need for labelled data in the target domain. This makes it particularly useful for the tasks in which acquiring new labelled data is very expensive, such as for semantic and instance segmentation.

We conducted experiments considering different types of domain shifts, and we made publicly available two new datasets for the vehicle counting task that were also used for our tests. Experiments show a significant improvement using our UDA algorithm compared to the model's performance without domain adaptation.

Our method relies on a CNN model trained end-to-end with adversarial learning in the output space (i.e., the density maps), which contains rich information such as scene layout and context. The peculiarity of our adversarial learning scheme is that it forces the predicted density maps in the target domain to have local similarities with the ones in the source domain.

The proposed framework consisting of two modules: 1) a CNN that predicts traffic density maps, from which we estimate the number of vehicles in the scene, and 2) a discriminator that identifies whether a density map (received by the density map estimator) was generated from an image of the source domain or the target domain.

In the training phase, the density map predictor learns to map images to densities based on annotated data from the source domain. At the same time, it learns to predict realistic density maps for the target domain by trying to fool the discriminator with an adversarial loss. The discriminator's output is a pixel-wise classification of a low-resolution map, where each pixel corresponds to a small region in the density map. Consequently, the output space is forced to be locally similar for both the source and target domains. In the inference phase, the discriminator is discarded, and only the density map predictor is used for the target images. We describe each module and how it is trained in the following subsections.

This activity is also related to T3.3 (Transfer Learning), and T5.3 (Learning with scarce data). A more detailed description of it is given in Deliverable D5.1 ("Initial report on Multimedia Summarisation and Analysis").

This activity was developed in synergy with the AI4EU project and it has already been uploaded in the AI on Demand Platform. In AI4Media we particularly took care of the learning with scarce data issue, which was addressed leveraging on solutions of domain adaptation.

5.3.1. Relevant Publications

Relevant Publications:

- Ciampi, L., Santiago, C., Costeira, J.P., Gennaro, C., Amato, G., Domain adaptation for traffic density estimation, VISIGRAPP 2021 - Proceedings of the 16th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, Volume 5, Pages 185-195, 2021 [187].
  Zenodo record: https://zenodo.org/record/5078270.

5.3.2. Relevant software and/or external resources

- The code of our work "AI for visual vehicles counting" can be found at https://www.ai4europe.eu/research/ai-catalog/ai-visual-vehicles-counting.

5.3.3. Relevant WP8 Use Cases

This tool contributes to use cases 2B1, 3C2-6, 3C2-7, 4C2, by providing solutions to analyze visual content and to count objects contained in it.

6. Ongoing Work and Conclusions

6.1. Ongoing work

Below, we briefly summarize the ongoing work associated to each task.

6.1.1. Lifelong and on-line learning (Task 3.1)

UNITN will focus on the problem of Memory-Constrained Online Continual Learning which imposes strict constraints on the memory overhead that a possible algorithm can use to avoid catastrophic forgetting. As most, if not all, previous continual learning methods violate these constraints, we will investigate approaches that effectively balances stability and plasticity in order to learn from data streams, while preserving the ability to solve old tasks through distillation.

CEA's ongoing work is focused on class-incremental learning without memory of past data. This is a very challenging scenario since the effect of catastrophic forgetting is very strong without past data replay. Two topics are of particular interest: (1) transferring knowledge between datasets to reduce the effect of catastrophic forgetting and (2) ensuring a better balance between stability and plasticity by mixing fixed and evolving feature representations.

6.1.2. Transfer learning (Task 3.3)

UNITN will focus on the problem of Unsupervised domain adaptation (UDA) on videos. Inspired by recent literature on contrastive learning we are currently consider a novel UDA approach for action recognition from videos which uses a two-headed deep architecture which simultaneously adopts cross-entropy and contrastive losses from different network branches to robustly learn a target classifier. Moreover, we will consider the domain transfer from synthetically generated action videos to real-life ones.

CNR's ongoing work includes further extensions of the gFun architecture discussed in Section 4.4. These extensions include:

- the addition of a further VGFs that allows the use of document embeddings generated via multilingual BERT (which encode context-dependent word-word correlations);
- the addition of a new function (based on vector averaging, and made possible by a further layer of classifiers) for combining the outputs of different VGFs into a single representation to be fed to the metaclassifier.

A journal paper that reports on these findings is in preparation. Planned work also includes the use of the gFun architecture for performing *cross-media* (instead of *cross-lingual*) classification, i.e., classification in which training items for, say, the "text" domain, contribute to the classification of unlabelled items for, say, the "image" domain.

CEA's ongoing work is focused on the proposal of a stable deep model pretraining pipeline which exploits semantically diversified datasets. Particular attention is given to the class diversification step, which is crucial for obtaining models which are transferable toward a large number of target tasks.

6.1.3. Learning to count (Task 3.7)

CNR's ongoing work includes the following:

- A. Esuli, A. Moreo and F. Sebastiani (all CNR) have developed (and made publicly available) QuaPy, an open-source, Python-based framework that implements several learning methods, evaluation measures, parameter optimisation routines, and evaluation protocols, for learning to quantify (https://github.com/HLT-ISTI/QuaPy); a paper that presents it [188] has also been submitted to an international conference;
- A. Moreo and F. Sebastiani (both CNR) have submitted a paper [189] on tweet sentiment quantification to an international journal;
- A. Esuli, A. Moreo and F. Sebastiani (all CNR) are working on two variants of SLD, a state-of-the-art quantification method; the first variant combines aspects of SLD and aspects of binning-based probability calibration methods, while the second variant brings into SLD intuitions from PACC, another state-of-the-art quantification method;
- A. Esuli, A. Moreo and F. Sebastiani (all CNR), with A. Fabris (University of Padova), are working on an application of learning to quantify to monitoring and mitigating the bias of classifiers; a journal paper is in preparation;
- A. Moreo and F. Sebastiani (both CNR), with J. Pickens (OpenText Inc.), are working on an application of learning to quantify to estimating recall in technology-assisted review;
- A. Esuli, A. Moreo and F. Sebastiani (all CNR), with A. Fabris (University of Padova), are in the process of finishing a book on learning to quantify, to appear in Springer Nature's "Information Retrieval Series";
- A. Esuli, A. Moreo and F. Sebastiani (all CNR) have submitted a proposal for organizing a "lab" (i.e., shared task) on learning to quantify at the CLEF 2022 conference (https://clef2022.clef-initiative.eu/), to be held in September 2022;
- A. Moreo and F. Sebastiani (both CNR) are co-organizers (with J.J. Del Coz and P. González, both University of Oviedo) of a workshop on "Learning to quantify: Methods and Applications" (LQ 2021 – https://cikmlq2021.github.io/), co-located with the CIKM 2021 conference and to take place in November 2021.

6.2. Conclusions

In this deliverable, we presented the current research results of WP3 regarding the new learning paradigms, specifically on the tasks: 3.1 (lifelong and on-line learning), 3.3 (transfer learning) and 3.7 (learning to count).

Several new methodologies bringing novel solutions and state of the art results are presented. These include new approaches for novel class discovery, class-incremental learning, dynamic adaptation of DNNs and compatible feature learning that fall under the category of lifelong and on-line learning (Task 3.1). The presented methodologies in Task 3.1 have been tested mainly on image classification tasks including but not limited to fine-grained object recognition, face recognition, landmark recognition and so forth, and represent generic approaches that could be modified to be applied on other types of media data e.g., videos and audio.

Important contributions, showing improved state of the art results, have been demonstrated for transfer learning (Task 3.3) as well. A diverse set of challenging tasks of transfer learning: multi-target domain adaptation, multi-source domain adaptation, and heterogeneous document embeddings for cross-lingual text classification have been addressed. The approaches that are tested on image classification (i.e., multi-target domain adaptation and multi-source domain adaptation) can be adapted to process other media, e.g., video and audio while heterogeneous document embedding learning targets text classification. All methods rely on essential modalities of multimedia, those considered are indeed covered by the use cases of AI4Media.

On the other hand, novel approaches developed for re-assessing quantification methods and counting objects by leveraging domain adaptation (Task 3.7) provide solutions to analyze visual content and therefore a service for various use of AI4Media.

In summary, the activity so far has been very intense and successful allowing us to have a large number of already published articles (7 conference articles and 2 journal articles) and several submitted contributions. The work reported in the deliverable is of very good quality and reflects the active involvement of all the partners towards fulfilling the goals of the workpackage. While this document covers only the initial results, the current trend is such that it is able to convincingly assure the good continuation of the work according to the original planning. The deliverable D3.3 (M36) will include the outcomes of the ongoing and new work regarding the tasks covered in this deliverable.

The only action point to be considered is the limited amount of reported joint activities. In practice this is not actually critical considering that WP3 is mostly meant to be a "service" workpackage used for developing tools for other WPs and the use cases. Nevertheless, the goal in the near future is to foster joint research that can be mutually beneficial to the cooperating partners. For instance, the WP3 partners will collaborate with the WP5 partners (recall that WP5 focuses in developing novel approaches for content-centered AI, addressing AI issues in content production and processing, specifically targeting challenges in textual / visual / audio media, multimedia production and enhancement, and summarisation) to address the limitations of deep learning related to training data scarcity (e.g., through transfer and lifelong learning) extending the potential applicability of AI to a wider set of media, and to apply CNN/DNN to improve tools for analyzing content provenance and reuse (e.g., through lifelong and online learning and learn to count). Similarly, the developed algorithms in WP3 would allow putting AI technologies to the service of citizens and societies together with WP6 and use cases developed in WP8, resulting in a potential collaboration between WP3 partners and WP6 and WP8.

References

[1] K. Han, S.-A. Rebuffi, S. Ehrhardt, A. Vedaldi, and A. Zisserman, "Automatically discovering and learning new visual categories with ranking statistics," in *Proc. ICLR*, 2020.

[2] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.

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

[4] H. Zhang, M. Cissé, Y. N. Dauphin, and D. Lopez-Paz, "MIXUP: Beyond empirical risk minimization," in *Proc. ICLR*, 2018.

[5] D. Berthelot, N. Carlini, I. Goodfellow, N. Papernot, A. Oliver, and C. A. Raffel, "Mixmatch: A holistic approach to semi-supervised learning," in *Proc. NeurIPS*, 2019.

[6] M. A. Bautista, A. Sanakoyeu, E. Sutter, and B. Ommer, "Cliquecnn: deep unsupervised exemplar learning," in *Proc. NeurIPS*, 2016.

[7] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. ICML*, 2020.

[8] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proc. CVPR*, 2020.

[9] J. Xie, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *Proc. ICML*, 2016.

[10] K. Han, A. Vedaldi, and A. Zisserman, "Learning to discover novel visual categories via deep transfer clustering," in *Proc. ICCV*, 2019.

[11] Y.-C. Hsu, Z. Lv, and Z. Kira, "Learning to cluster in order to transfer across domains and tasks," in *Proc. ICLR*, 2018.

[12] Y.-C. Hsu, Z. Lv, J. Schlosser, P. Odom, and Z. Kira, "Multi-class classification without multi-class labels," in *Proc. ICLR*, 2019.

[13] A. Oliver, A. Odena, C. A. Raffel, E. D. Cubuk, and I. Goodfellow, "Realistic evaluation of deep semi-supervised learning algorithms," in *Proc. NeurIPS*, 2018.

[14] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. CVPR*, 2016.

[15] J. Chang, L. Wang, G. Meng, S. Xiang, and C. Pan, "Deep adaptive image clustering," in *Proc. ICCV*, 2017.

[16] J. Wu, K. Long, F. Wang, C. Qian, C. Li, Z. Lin, and H. Zha, "Deep comprehensive correlation mining for image clustering," in *Proc. ICCV*, 2019.

[17] A. Krizhevsky, G. Hinton, *et al.*, "Learning multiple layers of features from tiny images," 2009.

[18] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Proc. CVPR*, 2009.

[19] A. Strehl and J. Ghosh, "Cluster ensembles—a knowledge reuse framework for combining multiple partitions," *JMLR*, 2002.

[20] J. MacQueen *et al.*, "Some methods for classification and analysis of multivariate observations," in *Proc. BSMSP*, 1967.

[21] Y. Tian, D. Krishnan, and P. Isola, "Contrastive multiview coding," in *Proc. ECCV*, 2020.

[22] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," in *Proc. CVPR*, 2018.

[23] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, and D. Krishnan, "Supervised contrastive learning," in *Proc. NeurIPS*, 2020.

[24] Z. Zhong, L. Zhu, Z. Luo, S. Li, Y. Yang, and N. Sebe, "Openmix: Reviving known knowledge for discovering novel visual categories in an open world," in *CVPR*, 2021.

[25] Z. Zhong, E. Fini, S. Roy, Z. Luo, E. Ricci, and N. Sebe, "Neighborhood contrastive learning for novel class discover," in *CVPR*, 2021.

[26] E. Belouadah, A. Popescu, and I. Kanellos, "A comprehensive study of class incremental learning algorithms for visual tasks," *Neural Networks*, 2020.

[27] S. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, "icarl: Incremental classifier and representation learning," in *Conference on Computer Vision and Pattern Recognition*, CVPR, 2017.

[28] Y. Tamaazousti, H. L. Borgne, C. Hudelot, M. E. A. Seddik, and M. Tamaazousti, "Learning more universal representations for transfer-learning," *CoRR*, vol. abs/1712.09708, 2017.

[29] S. Rebuffi, H. Bilen, and A. Vedaldi, "Efficient parametrization of multi-domain deep neural networks," in *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pp. 8119–8127, 2018.

[30] A. A. Rusu, N. C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick, K. Kavukcuoglu, R. Pascanu, and R. Hadsell, "Progressive neural networks," *CoRR*, vol. abs/1606.04671, 2016.

[31] Y. Wang, D. Ramanan, and M. Hebert, "Growing a brain: Fine-tuning by increasing model capacity," in *Conference on Computer Vision and Pattern Recognition*, CVPR, 2017.

[32] R. Aljundi, P. Chakravarty, and T. Tuytelaars, "Expert gate: Lifelong learning with a network of experts," in *Conference on Computer Vision and Pattern Recognition*, CVPR, 2017.

[33] M. D. Lange, R. Aljundi, M. Masana, S. Parisot, X. Jia, A. Leonardis, G. G. Slabaugh, and T. Tuytelaars, "Continual learning: A comparative study on how to defy forgetting in classification tasks," *CoRR*, vol. abs/1909.08383, 2019.

[34] E. Belouadah and A. Popescu, "Deesil: Deep-shallow incremental learning," *TaskCV Workshop @ ECCV 2018.*, 2018.

[35] S. Rebuffi, H. Bilen, and A. Vedaldi, "Learning multiple visual domains with residual adapters," in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pp. 506–516, 2017.

[36] R. Kemker and C. Kanan, "Fearnet: Brain-inspired model for incremental learning," in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, 2018.

[37] F. M. Castro, M. J. Marín-Jiménez, N. Guil, C. Schmid, and K. Alahari, "End-to-end incremental learning," in *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XII*, pp. 241–257, 2018.

[38] S. Hou, X. Pan, C. C. Loy, Z. Wang, and D. Lin, "Learning a unified classifier incrementally via rebalancing," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pp. 831–839, 2019.

[39] Y. Wu, Y. Chen, L. Wang, Y. Ye, Z. Liu, Y. Guo, and Y. Fu, "Large scale incremental learning," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pp. 374–382, 2019.

[40] R. Aljundi, F. Babiloni, M. Elhoseiny, M. Rohrbach, and T. Tuytelaars, "Memory aware synapses: Learning what (not) to forget," in *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part III* (V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, eds.), vol. 11207 of *Lecture Notes in Computer Science*, pp. 144–161, Springer, 2018.

[41] D. Roy, P. Panda, and K. Roy, "Tree-cnn: A hierarchical deep convolutional neural network for incremental learning," *Neural Networks*, vol. 121, pp. 148–160, 2020.

[42] K. Javed and F. Shafait, "Revisiting distillation and incremental classifier learning," *CoRR*, vol. abs/1807.02802, 2018.

[43] E. Belouadah and A. Popescu, "Il2m: Class incremental learning with dual memory," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 583–592, 2019.

[44] E. Belouadah and A. Popescu, "Scail: Classifier weights scaling for class incremental learning," in *The IEEE Winter Conference on Applications of Computer Vision (WACV)*, March 2020.

[45] T. L. Hayes and C. Kanan, "Lifelong machine learning with deep streaming linear discriminant analysis," *CoRR*, vol. abs/1909.01520, 2019.

[46] A. Mallya, D. Davis, and S. Lazebnik, "Piggyback: Adapting a single network to multiple tasks by learning to mask weights," in *ECCV (4)*, vol. 11208 of *Lecture Notes in Computer Science*, pp. 72–88, Springer, 2018.

[47] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features off-the-shelf: An astounding baseline for recognition," in *Conference on Computer Vision and Pattern Recognition Workshop*, CVPR-W, 2014.

[48] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. S. Bernstein, A. C. Berg, and F. Li, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[49] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and A. Zisserman, "Vggface2: A dataset for recognising faces across pose and age," in *13th IEEE International Conference on Automatic Face & Gesture Recognition, FG 2018, Xi'an, China, May 15-19, 2018*, pp. 67–74, 2018.

[50] H. Noh, A. Araujo, J. Sim, T. Weyand, and B. Han, "Large-scale image retrieval with attentive deep local features," in *ICCV*, pp. 3476–3485, IEEE Computer Society, 2017.

[51] A. Krizhevsky, "Learning multiple layers of features from tiny images," tech. rep., University of Toronto, 2009.

[52] E. Belouadah, A. Popescu, and I. Kanellos, "Initial classifier weights replay for memoryless class incremental learning," in *British Machine Vision Conference (BMVC)*, 2020.

[53] T. L. Hayes, K. Kafle, R. Shrestha, M. Acharya, and C. Kanan, "REMIND your neural network to prevent catastrophic forgetting," *CoRR*, vol. abs/1910.02509, 2019.

[54] M. Buda, A. Maki, and M. A. Mazurowski, "A systematic study of the class imbalance problem in convolutional neural networks," *Neural Networks*, vol. 106, pp. 249–259, 2018.

[55] D. Hendrycks and K. Gimpel, "A baseline for detecting misclassified and out-of-distribution examples in neural networks," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.

[56] S. Liang, Y. Li, and R. Srikant, "Enhancing the reliability of out-of-distribution image detection in neural networks," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

[57] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," in *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*, 2017.

[58] H. Choi, E. Jang, and A. A. Alemi, "WAIC, but why? Generative ensembles for robust anomaly detection," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

[59] E. Nalisnick, A. Matsukawa, Y. W. Teh, D. Gorur, and B. Lakshminarayanan, "Do deep generative models know what they don't know?," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

[60] J. Ren, P. J. Liu, E. Fertig, J. Snoek, R. Poplin, M. A. DePristo, J. V. Dillon, and B. Lakshminarayanan, "Likelihood ratios for out-of-distribution detection," in *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*, 2019.

[61] V. Abdelzad, K. Czarnecki, R. Salay, T. Denounden, S. Vernekar, and B. Phan, "Detecting out-of-distribution inputs in Deep Neural Networks using an early-layer output," *arXiv preprint arXiv:1910.10307*, 2019.

[62] K. Lee, K. Lee, H. Lee, and J. Shin, "A simple unified framework for detecting out-of-distribution samples and adversarial attacks," in *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*, 2018.

[63] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian approximation: Representing model uncertainty in deep learning," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2016.

[64] A. A. Alemi, I. Fischer, and J. V. Dillon, "Uncertainty in the variational information bottleneck," in *Proceedings of the Conference on Uncertainty in Artificial Intelligence Workshops (UAIW)*, 2018.

[65] K. Lee, H. Lee, K. Lee, and J. Shin, "Training confidence-calibrated classifiers for detecting out-of-distribution samples," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

[66] T. DeVries and G. W. Taylor, "Learning confidence for out-of-distribution detection in neural networks," *arXiv preprint arXiv:1802.04865*, 2018.

[67] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4700–4708, 2017.

[68] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. NeurIPS*, 2014.

[69] P. Oberdiek, M. Rottmann, and G. A. Fink, "Detection and retrieval of out-of-distribution objects in semantic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 328–329, 2020.

[70] R. Chan, M. Rottmann, and H. Gottschalk, "Entropy maximization and meta classification for out-of-distribution detection in semantic segmentation," *arXiv preprint arXiv:2012.06575*, 2020.

[71] L. Andraghetti, P. Myriokefalitakis, P. L. Dovesi, B. Luque, M. Poggi, A. Pieropan, and S. Mattoccia, "Enhancing self-supervised monocular depth estimation with traditional visual odometry," in *Proceedings of the IEEE International Conference on 3D Vision (3DV)*, pp. 424–433, 2019.

[72] M. Poggi, F. Aleotti, F. Tosi, and S. Mattoccia, "On the uncertainty of self-supervised monocular depth estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3227–3237, 2020.

[73] C. Godard, O. Mac Aodha, M. Firman, and G. J. Brostow, "Digging into self-supervised monocular depth estimation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 3828–3838, 2019.

[74] C. Godard, O. Mac Aodha, and G. J. Brostow, "Unsupervised monocular depth estimation with left-right consistency," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 270–279, 2017.

[75] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1125–1134, 2017.

[76] I. Mademlis and I. Pitas, "Out-of-distribution detection for deep convolutional neural networks: An overview," in *Submitted*, 2021.

[77] F. Pernici, M. Bruni, C. Baecchi, F. Turchini, and A. D. Bimbo, "Class-incremental learning with pre-allocated fixed classifiers," in *25th International Conference on Pattern Recognition, ICPR 2020, Milan, Italy, January 10-15, 2021*, IEEE Computer Society, 2020.

[78] F. Pernici, M. Bruni, C. Baecchi, and A. D. Bimbo, "Regular polytope networks," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–15, 2021.

[79] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[80] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural networks*, vol. 61, pp. 85–117, 2015.

[81] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[82] H. Touvron, A. Vedaldi, M. Douze, and H. Jégou, "Fixing the train-test resolution discrepancy," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[83] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and A. Zisserman, "Vggface2: A dataset for recognising faces across pose and age," in *13th IEEE International Conference on Automatic Face & Gesture Recognition, FG 2018, Xi'an, China, May 15-19, 2018*, pp. 67–74, 2018.

[84] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4690–4699, 2019.

[85] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European conference on computer vision*, pp. 818–833, Springer, 2014.

[86] M. Lin, Q. Chen, and S. Yan, "Network in network," *International Conference on Learning Representations (ICLR)*, 2014.

[87] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.

[88] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005), 20-26 June 2005, San Diego, CA, USA*, pp. 539–546, 2005.

[89] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 815–823, 2015.

[90] E. Hoffer and N. Ailon, "Deep metric learning using triplet network," in *International Workshop on Similarity-Based Pattern Recognition*, pp. 84–92, Springer, 2015.

[91] E. Hoffer, I. Hubara, and D. Soudry, "Fix your classifier: the marginal value of training the last weight layer," in *International Conference on Learning Representations (ICLR)*, 2018.

[92] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32nd International Conference on Machine Learning* (F. Bach and D. Blei, eds.), vol. 37 of *Proceedings of Machine Learning Research*, (Lille, France), pp. 448–456, PMLR, 07–09 Jul 2015.

[93] Z. Li and S. Arora, "An exponential learning rate schedule for deep learning," in *International Conference on Learning Representations*, 2019.

[94] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning*, vol. 1. MIT Press, 2016.

[95] F. Pernici, M. Bruni, C. Baecchi, and A. Del Bimbo, "Maximally compact and separated features with regular polytope networks," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2019.

[96] F. Pernici, M. Bruni, C. Baecchi, F. Turchini, and A. D. Bimbo, "Class-incremental learning with pre-allocated fixed classifiers," in *25th International Conference on Pattern Recognition, ICPR 2020, Milan, Italy, January 10-15, 2021*, IEEE Computer Society, 2020.

[97] M. McCloskey and N. J. Cohen, "Catastrophic interference in connectionist networks: The sequential learning problem," in *Psychology of learning and motivation*, vol. 24, pp. 109–165, Elsevier, 1989.

[98] R. Ratcliff, "Connectionist models of recognition memory: constraints imposed by learning and forgetting functions.," *Psychological review*, vol. 97, no. 2, p. 285, 1990.

[99] I. J. Goodfellow, M. Mirza, D. Xiao, A. Courville, and Y. Bengio, "An empirical investigation of catastrophic forgetting in gradient-based neural networks," *arXiv preprint arXiv:1312.6211*, 2013.

[100] S. Grossberg, "How does a brain build a cognitive code?," in *Studies of mind and brain*, pp. 1–52, Springer, 1982.

[101] Z. Chen and B. Liu, "Lifelong machine learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 12, no. 3, pp. 1–207, 2018.

[102] R. Aljundi, E. Belilovsky, T. Tuytelaars, L. Charlin, M. Caccia, M. Lin, and L. Page-Caccia, "Online continual learning with maximal interfered retrieval," in *Advances in Neural Information Processing Systems*, pp. 11849–11860, 2019.

[103] D. Rao, F. Visin, A. Rusu, R. Pascanu, Y. W. Teh, and R. Hadsell, "Continual unsupervised representation learning," in *Advances in Neural Information Processing Systems*, pp. 7647–7657, 2019.

[104] S. Hou, X. Pan, C. C. Loy, Z. Wang, and D. Lin, "Learning a unified classifier incrementally via rebalancing," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[105] Y. Liu, Y. Su, A. Liu, B. Schiele, and Q. Sun, "Mnemonics training: Multi-class incremental learning without forgetting," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[106] M. Wortsman, V. Ramanujan, R. Liu, A. Kembhavi, M. Rastegari, J. Yosinski, and A. Farhadi, "Supermasks in superposition," *arXiv*, pp. arXiv–2006, 2020.

[107] Q. Liu, O. Majumder, A. Ravichandran, R. Bhotika, and S. Soatto, "Incremental learning for metric-based meta-learners," *ECCV*, 2020.

[108] R. Kurle, B. Cseke, A. Klushyn, P. van der Smagt, and S. Günnemann, "Continual learning with bayesian neural networks for non-stationary data," in *International Conference on Learning Representations*, 2019.

[109] J. von Oswald, C. Henning, J. Sacramento, and B. F. Grewe, "Continual learning with hypernetworks," in *International Conference on Learning Representations*, 2020.

[110] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, *et al.*, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the national academy of sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.

[111] C. V. Nguyen, Y. Li, T. D. Bui, and R. E. Turner, "Variational continual learning," *arXiv preprint arXiv:1710.10628*, 2017.

[112] F. Zenke, B. Poole, and S. Ganguli, "Continual learning through synaptic intelligence," in *Proceedings of machine learning research*, vol. 70, p. 3987, Europe PMC Funders, 2017.

[113] H. Shin, J. K. Lee, J. Kim, and J. Kim, "Continual learning with deep generative replay," in *Advances in Neural Information Processing Systems*, pp. 2990–2999, 2017.

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

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

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

[117] D. Maltoni and V. Lomonaco, "Continuous learning in single-incremental-task scenarios," *Neural Networks*, vol. 116, pp. 56–73, 2019.

[118] G. M. van de Ven and A. S. Tolias, "Three scenarios for continual learning," *arXiv preprint arXiv:1904.07734*, 2019.

[119] F. Pernici, M. Bruni, C. Baecchi, and A. Del Bimbo, "Fix your features: Stationary and maximally discriminative embeddings using regular polytope (fixed classifier) networks," *arXiv preprint arXiv:1902.10441*, 2019.

[120] A. Torralba and A. A. Efros, "Unbiased look at dataset bias," in *Proc. CVPR*, 2011.

[121] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell, "Deep domain confusion: Maximizing for domain invariance," *arXiv*, 2014.

[122] M. Long and J. Wang, "Learning transferable features with deep adaptation networks," in *Proc. ICML*, 2015.

[123] B. Sun and K. Saenko, "Deep coral: Correlation alignment for deep domain adaptation," in *Proc. ECCV*, 2016.

[124] X. Peng and K. Saenko, "Synthetic to real adaptation with generative correlation alignment networks," in *Proc. WACV*, 2018.

[125] F. M. Carlucci, L. Porzi, B. Caputo, E. Ricci, and S. R. Bulò, "Autodial: Automatic domain alignment layers.," in *Proc. ICCV*, 2017.

[126] F. M. Carlucci, L. Porzi, B. Caputo, E. Ricci, and S. R. Bulo, "Just dial: Domain alignment layers for unsupervised domain adaptation," in *Proc. ICIAP*, 2017.

[127] W.-G. Chang, T. You, S. Seo, S. Kwak, and B. Han, "Domain-specific batch normalization for unsupervised domain adaptation," in *Proc. CVPR*, 2019.

[128] S. Roy, A. Siarohin, E. Sangineto, S. R. Bulo, N. Sebe, and E. Ricci, "Unsupervised domain adaptation using feature-whitening and consensus loss," *Proc. CVPR*, 2019.

[129] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, "Domain-adversarial training of neural networks," *JMLR*, 2016.

[130] E. Tzeng, J. Hoffman, T. Darrell, and K. Saenko, "Adversarial discriminative domain adaptation," in *Proc. CVPR*, 2017.

[131] M. Long, Z. Cao, J. Wang, and M. I. Jordan, "Conditional adversarial domain adaptation," in *Proc. NeurIPS*, 2018.

[132] P. Russo, F. M. Carlucci, T. Tommasi, and B. Caputo, "From source to target and back: symmetric bi-directional adaptive gan," in *Proc. CVPR*, 2018.

[133] J. Hoffman, E. Tzeng, T. Park, J.-Y. Zhu, P. Isola, K. Saenko, A. A. Efros, and T. Darrell, "Cycada: Cycle-consistent adversarial domain adaptation," in *Proc. ICML*, 2017.

[134] M.-Y. Liu and O. Tuzel, "Coupled generative adversarial networks," in *Proc. NeurIPS*, 2016.

[135] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. ICLR*, 2017.

[136] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training," in *Proc. CLT*, 1998.

[137] R. Xu, Z. Chen, W. Zuo, J. Yan, and L. Lin, "Deep cocktail network: Multi-source unsupervised domain adaptation with category shift," in *Proc. CVPR*, 2018.

[138] K. Saenko, B. Kulis, M. Fritz, and T. Darrell, "Adapting visual category models to new domains," in *Proc. ECCV*, 2010.

[139] D. Li, Y. Yang, Y.-Z. Song, and T. M. Hospedales, "Deeper, broader and artier domain generalization," in *Proc. ICCV*, 2017.

[140] H. Venkateswara, J. Eusebio, S. Chakraborty, and S. Panchanathan, "Deep hashing network for unsupervised domain adaptation," in *Proc. CVPR*, 2017.

[141] X. Peng, Q. Bai, X. Xia, Z. Huang, K. Saenko, and B. Wang, "Moment matching for multi-source domain adaptation," in *Proc. ICCV*, 2019.

[142] G. French, M. Mackiewicz, and M. Fisher, "Self-ensembling for visual domain adaptation," in *Proc. ICLR*, 2018.

[143] K. Saito, K. Watanabe, Y. Ushiku, and T. Harada, "Maximum classifier discrepancy for unsupervised domain adaptation," in *Proc. CVPR*, 2018.

[144] X. Peng, Z. Huang, X. Sun, and K. Saenko, "Domain agnostic learning with disentangled representations," *arXiv*, 2019.

[145] Y. Jin, X. Wang, M. Long, and J. Wang, "Minimum class confusion for versatile domain adaptation," in *Proc. ECCV*, 2020.

[146] S. Roy, E. Krivosheev, Z. Zhong, N. Sebe, and E. Ricci, "Curriculum graph co-teaching formulti-target domain adaptation," in *CVPR*, 2021.

[147] X. Jin, C. Lan, W. Zeng, Z. Chen, and L. Zhang, "Style normalization and restitution for generalizable person re-identification," in *CVPR*, 2020.

[148] S. Liao and L. Shao, "Interpretable and generalizable person re-identification with query-adaptive convolution and temporal lifting," in *ECCV*, 2020.

[149] K. Zhou, X. Zhu, Y. Yang, A. Cavallaro, and T. Xiang, "Learning generalisable omni-scale representations for person re-identification," *arXiv:1910.06827*, 2019.

[150] D. Kumar, P. Siva, P. Marchwica, and A. Wong, "Fairest of them all: Establishing a strong baseline for cross-domain person reid," *arXiv:1907.12016*, 2019.

[151] J. Song, Y. Yang, Y.-Z. Song, T. Xiang, and T. M. Hospedales, "Generalizable person re-identification by domain-invariant mapping network," in *CVPR*, 2019.

[152] A. Hermans, L. Beyer, and B. Leibe, "In defense of the triplet loss for person re-identification," *arXiv:1703.07737*, 2017.

[153] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. ICML*, 2015.

[154] L. Zheng, L. Shen, L. Tian, S. Wang, J. Wang, and Q. Tian, "Scalable person re-identification: A benchmark," in *CVPR*, 2015.

[155] E. Ristani, F. Solera, R. Zou, R. Cucchiara, and C. Tomasi, "Performance measures and a data set for multi-target, multi-camera tracking," in *ECCVW*, 2016.

[156] Z. Zheng, L. Zheng, and Y. Yang, "Unlabeled samples generated by gan improve the person re-identification baseline in vitro," in *ICCV*, 2017.

[157] W. Li, R. Zhao, T. Xiao, and X. Wang, "Deepreid: Deep filter pairing neural network for person re-identification," in *CVPR*, 2014.

[158] Z. Zhong, L. Zheng, D. Cao, and S. Li, "Re-ranking person re-identification with k-reciprocal encoding," in *CVPR*, 2017.

[159] L. Wei, S. Zhang, W. Gao, and Q. Tian, "Person transfer gan to bridge domain gap for person re-identification," in *CVPR*, 2018.

[160] K. Zhou, Y. Yang, A. Cavallaro, and T. Xiang, "Omni-scale feature learning for person re-identification," in *ICCV*, 2019.

[161] Y. Zhao, Z. Zhong, F. Yang, Z. Luo, Y. Lin, S. Li, and N. Sebe, "Learning to generalize unseendomains via memory-based multi-source meta-learning for person re-identification," in *CVPR*, 2021.

[162] A. Esuli, A. Moreo, and F. Sebastiani, "Funnelling: A new ensemble method for heterogeneous transfer learning and its application to cross-lingual text classification," *ACM Transactions on Information Systems*, vol. 37, no. 3, p. Article 37, 2019.

[163] A. Moreo, A. Esuli, and F. Sebastiani, "Word-class embeddings for multiclass text classification," *Data Mining and Knowledge Discovery*, vol. 353, no. 3, pp. 911–963, 2021.

[164] A. Conneau, G. Lample, M. Ranzato, L. Denoyer, and H. Jégou, "Word translation without parallel data," in *Proceedings of the 6th International Conference on Learning Representations (ICLR 2018)*, (Vancouver, CA), 2018.

[165] S. Arora, Y. Liang, and T. Ma, "A simple but tough-to-beat baseline for sentence embeddings," in *Proceedings of the 5th International Conference on Learning Representations (ICLR 2017)*, (Toulon, FR), 2017.

[166] A. Moreo, A. Pedrotti, and F. Sebastiani, "Heterogeneous document embeddings for cross-lingual text classification," in *Proceedings of the 36th ACM Symposium on Applied Computing (SAC 2021)*, (Gwangju, KR), pp. 685–688, 2021.

[167] P. González, A. Castaño, N. V. Chawla, and J. J. del Coz, "A review on quantification learning," *ACM Computing Surveys*, vol. 50, no. 5, pp. 74:1–74:40, 2017.

[168] J. G. Moreno-Torres, T. Raeder, R. Alaíz-Rodríguez, N. V. Chawla, and F. Herrera, "A unifying view on dataset shift in classification," *Pattern Recognition*, vol. 45, no. 1, pp. 521–530, 2012.

[169] A. Esuli and F. Sebastiani, "Explicit loss minimization in quantification applications (preliminary draft)," in *Proceedings of the 8th International Workshop on Information Filtering and Retrieval (DART 2014)*, (Pisa, IT), pp. 1–11, 2014.

[170] G. Forman, "Quantifying counts and costs via classification," *Data Mining and Knowledge Discovery*, vol. 17, no. 2, pp. 164–206, 2008.

[171] P. González, J. Díez, N. Chawla, and J. J. del Coz, "Why is quantification an interesting learning problem?," *Progress in Artificial Intelligence*, vol. 6, no. 1, pp. 53–58, 2017.

[172] V. González-Castro, R. Alaiz-Rodríguez, and E. Alegre, "Class distribution estimation based on the Hellinger distance," *Information Sciences*, vol. 218, pp. 146–164, 2013.

[173] D. J. Hopkins and G. King, "A method of automated nonparametric content analysis for social science," *American Journal of Political Science*, vol. 54, no. 1, pp. 229–247, 2010.

[174] R. Levin and H. Roitman, "Enhanced probabilistic classify and count methods for multi-label text quantification," in *Proceedings of the 7th ACM International Conference on the Theory of Information Retrieval (ICTIR 2017)*, (Amsterdam, NL), pp. 229–232, 2017.

[175] P. Pérez-Gállego, J. R. Quevedo, and J. J. del Coz, "Using ensembles for problems with characterizable changes in data distribution: A case study on quantification," *Information Fusion*, vol. 34, pp. 87–100, 2017.

[176] M. Saerens, P. Latinne, and C. Decaestecker, "Adjusting the outputs of a classifier to new a priori probabilities: A simple procedure," *Neural Computation*, vol. 14, no. 1, pp. 21–41, 2002.

[177] J. Barranquero, J. Díez, and J. J. del Coz, "Quantification-oriented learning based on reliable classifiers," *Pattern Recognition*, vol. 48, no. 2, pp. 591–604, 2015.

[178] A. Bella, C. Ferri, J. Hernández-Orallo, and M. J. Ramírez-Quintana, "Quantification via probability estimators," in *Proceedings of the 11th IEEE International Conference on Data Mining (ICDM 2010)*, (Sydney, AU), pp. 737–742, 2010.

[179] A. Esuli and F. Sebastiani, "Optimizing text quantifiers for multivariate loss functions," *ACM Transactions on Knowledge Discovery and Data*, vol. 9, no. 4, p. Article 27, 2015.

[180] W. Hassan, A. Maletzke, and G. Batista, "Accurately quantifying a billion instances per second," in *Proceedings of the 7th IEEE International Conference on Data Science and Advanced Analytics (DSAA 2020)*, (Sydney, AU), 2020.

[181] L. Milli, A. Monreale, G. Rossetti, F. Giannotti, D. Pedreschi, and F. Sebastiani, "Quantification trees," in *Proceedings of the 13th IEEE International Conference on Data Mining (ICDM 2013)*, (Dallas, US), pp. 528–536, 2013.

[182] A. Esuli, A. Moreo, and F. Sebastiani, "Cross-lingual sentiment quantification," *IEEE Intelligent Systems*, vol. 35, no. 3, pp. 106–114, 2020.

[183] W. Gao and F. Sebastiani, "From classification to quantification in tweet sentiment analysis," *Social Network Analysis and Mining*, vol. 6, no. 19, pp. 1–22, 2016.

[184] J. Barranquero, P. González, J. Díez, and J. J. del Coz, "On the study of nearest neighbor algorithms for prevalence estimation in binary problems," *Pattern Recognition*, vol. 46, no. 2, pp. 472–482, 2013.

[185] P. Pérez-Gállego, A. Castaño, J. R. Quevedo, and J. J. del Coz, "Dynamic ensemble selection for quantification tasks," *Information Fusion*, vol. 45, pp. 1–15, 2019.

[186] A. Moreo and F. Sebastiani, "Re-assessing the "classify and count" quantification method," in *Proceedings of the 43rd European Conference on Information Retrieval (ECIR 2021)*, vol. II, (Lucca, IT), pp. 75–91, 2021.

[187] L. Ciampi, C. Santiago, J. Costeira, C. Gennaro, and G. Amato, "Domain adaptation for traffic density estimation," in *VISIGRAPP 2021 - Proceedings of the 16th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, pp. 185–195, 2021.

[188] A. Moreo, A. Esuli, and F. Sebastiani, "QuaPy: A Python-based framework for quantification." arXiv:2106.11057 [cs.LG], 2021.

[189] A. Moreo, A. Esuli, and F. Sebastiani, "Tweet sentiment quantification: An experimental re-evaluation." arXiv:2011.08091 [cs.CL], 2021.