
Politecnico di Torino

ScuDo
Scuola di Dottorato ~ Doctoral School
WHAT YOU ARE, TAKES YOU FAR

Doctoral Dissertation
Doctoral Program in Computer and Control Engineering ($35^{th}$cycle)

# Visual Domain Generalization via Self-Supervised Learning

By

**Silvia Bucci**

******

**Supervisor(s):**
Prof. Tatiana Tommasi, Supervisor

**Doctoral Examination Committee:**
Prof. Paolo Favaro, Referee, University of Bern
Prof. Elisa Ricci, Referee, University of Trento
Prof. Fabio Galasso, Sapienza University of Rome
Prof. Paolo Garza, Polytechnic of Turin
Dr. Gabriela Csurka, Naver Labs Europe

Politecnico di Torino
2023

# Declaration

I hereby declare that, the contents and organization of this dissertation constitute my own original work and does not compromise in any way the rights of third parties, including those relating to the security of personal data.

Silvia Bucci
2023

* This dissertation is presented in partial fulfillment of the requirements for **Ph.D. degree** in the Graduate School of Politecnico di Torino (ScuDo).

*I would like to dedicate this thesis to my loving parents*

# Acknowledgements

Firstly, I would like to give a huge thank you to my supervisor, Prof. Tatiana Tommasi who throughout all these years has been an exceptional mentor guiding me in every step and consistently fostering my personal growth. Thank you Tatiana for believing in me and having made these years so meaningful.

I would like to express my sincere gratitude to Prof. Paolo Favaro and Prof. Elisa Ricci for carefully reviewing this thesis. Receiving their positive and valuable feedback has been an honor for me.

Heartfelt thanks are reserved to Prof. Timothy Hospedales and all the members of the IPAB Lab at The University of Edinburgh for having welcomed me into their research group. I had the privilege of meeting many brilliant researchers who gave me valuable guidance and insights. I would especially thank Yinbing, Yongshuo, Fady, and Henry for all the lunches and coffees we had together, I hope we will meet again soon. A special thanks to Sabrina, Francesca, Pratika, and Alfonsi for all the laughs and experiences we shared while discovering the beauty of Scotland.

I would like to thank the Frontier Development Lab (FDL) community for the truly extraordinary and enriching experience I had the privilege of being a part of. During my time with FDL, I had the chance to connect with numerous young researchers, and these connections continue to endure. I want to extend a special thanks to Roelien, Takuya, and Aurelien for the memorable adventures we shared while exploring the big parks in California.

I thank my lifelong friends, from my beloved hometown in Molise, Giulia, Valeria, Grazia, Giulia Z., Ludovica, Arianna, Laura, and Arianna I. who taught me how true friendship can withstand anything.

I would like to thank the entire VANDAL Lab, starting from the first years with Antonio, Fabio, Dario, Tav, Massi, and Mohammad until the last years with Deb,

Paolo, Lidia, Linda, Luca, Alli, Berton, Nik, Tib, Eros, Chiara and all the others, I will never forget the time spent together, especially on Thursday evening, the happy hour day! A special thanks to Trivi, Gole, and Borlino for making me discover new *wild* passions, I cannot wait for the next adventure.

I want to thank my parents and my entire family for their constant support, I couldn't make it without them.

Lastly, I want to thank Mirco, the most important person in my life. You know how significant you have been for me in this journey. I couldn't have asked for more.

# Abstract

In these days the world is wondering about the potentialities and risks of artificial intelligence models trained on a huge amount of data and computational resources. While this debate is certainly important, it is also relevant to put the spotlight on a hallmark of human intelligence which is still far-fetched for machines: visual generalization and adaptability. Several studies in neuroscience have discussed how these skills develop in children from a combination of supervised and self-supervised learning, with the first guided by adults and the second spontaneously rising from playing and freely interacting with the world. Games like jigsaw puzzles and coloring help to learn the invariances and regularities of objects and scenes, and contribute to building robust semantic knowledge that generalizes to novel contexts.

With this thesis, we show how these learning strategies can be exploited to improve artificial model robustness and reliability. Specifically, we show how auxiliary self-supervised tasks can be paired with supervised ones with significant beneficial effects.

The manuscript is divided into three main parts. In the first part, we show how transformation-based self-supervised objectives (jigsaw puzzle, rotation recognition, inter-modal RGB-D translation) promote visual domain adaptation. In the second part, we discuss how the same strategies help when dealing not only with domain shift but also semantic shift due to new object categories. Finally, in the third part, we discuss relation-based self-supervised approaches (contrastive learning and relational reasoning) and how they can easily integrate supervision obtaining a powerful model that can efficiently cope with the open world.

**Keywords:** deep learning, domain adaptation, domain generalization, open world, transfer learning, self-supervised learning, multi-modal learning

# Contents

**List of Figures** **xi**

**List of Tables** **xviii**

**1 Introduction** **1**

1.1 Problem Statement . . . . . 4

1.2 Contributions . . . . . 5

1.3 Outline . . . . . 6

1.4 Publications and Reports . . . . . 8

**2 Related Works** **10**

2.1 Learning across Domains . . . . . 11

2.2 Self-Supervised Learning . . . . . 17

2.3 Datasets . . . . . 20

**3 Auxiliary Self-Supervision for Closed-Set Cross-Domain Learning** **28**

3.1 JiGen: Self-Supervised Learning Across Domains . . . . . 29

3.1.1 Method . . . . . 29

3.1.2 Experiments . . . . . 33

3.1.3 Computational Cost . . . . . 43

3.1.4 Conclusions . . . . . 44

3.2 TranAdapt: Multi-Modal RGB-D Scene Recognition Across Domains 45
3.2.1 Method . . . 47
3.2.2 Experiments . . . 49
3.2.3 Computational Cost . . . 55
3.2.4 Conclusions . . . 56

**4 Auxiliary Self-Supervision for Partial and Open-Set Cross-Domain Learning 58**

4.1 Tackling Partial Domain Adaptation with Self-Supervision . . . 59
4.1.1 Method . . . 59
4.1.2 Experiments . . . 61
4.1.3 Computational Cost . . . 64
4.1.4 Conclusions . . . 66
4.2 ROS: On the Effectiveness of Image Rotation for Open-Set DA . . . 67
4.2.1 Method . . . 68
4.2.2 On reproducibility and Open-Set metrics . . . 73
4.2.3 Experiments . . . 74
4.2.4 Computational Cost . . . 79
4.2.5 Conclusions . . . 79

**5 Improved Supervised Models for Cross-Domain Learning 81**

5.1 HyMOS: Distance-based Hyperspherical Classification for Multi-source Open-Set Domain Adaptation . . . 82
5.1.1 Method . . . 84
5.1.2 Experiments . . . 89
5.1.3 Ablation Analysis . . . 91
5.1.4 Extension to Closed-Set and Universal . . . 93
5.1.5 Computational Cost . . . 94

5.1.6 Conclusions . . . 94
5.2 ReSeND: Semantic Novelty Detection via Relational Reasoning . . 96
5.2.1 Method . . . 97
5.2.2 Experimental Setup . . . 100
5.2.3 Experiments . . . 102
5.2.4 Further analysis and discussions . . . 106
5.2.5 Computational Cost . . . 108
5.2.6 Conclusions . . . 109

**6 Conclusions 111**
6.1 Summary . . . 112
6.2 Open issues and Future Works . . . 113

**References 115**

**Appendix A Open-Set Cross-Domain Learning 138**
A.1 ROS . . . 138
A.1.1 Further implementation Details . . . 138
A.1.2 Reproducibility Study . . . 140
A.1.3 Extended Openness Analysis . . . 143
A.1.4 Sensitivity analysis of the hyper-parameters . . . 143
A.1.5 Other Self-Supervised Tasks and Further Ablation . . . 146
A.1.6 Normality Score Pseudo-code . . . 146

**Appendix B Improved Supervised models for Cross-Domain Learning 147**
B.1 HyMOS . . . 147
B.1.1 Qualitative Analysis . . . 147
B.1.2 Further experiments . . . 148

B.1.3 Implementation Details . . . . . 149
B.2 ReSeND . . . . . 151
B.2.1 Implementation details . . . . . 151
B.2.2 Further Analysis . . . . . 152

**Appendix C Project Contributions and Computational Resources Support 154**

# List of Figures

1.1 Examples of domain shift taken from DomainNet [1], SUN RGB-D [2], Cityscapes [3] and OfficeHome [4] datasets. . . . . 2

1.2 The greatest part of the baby's toys is designed to give children the ability to autonomously find patterns that don't require any supervision. 3

2.1 A schematic illustration of the Context Free Network (CFN) proposed in [5] trained to predict the index of the permutation applied to the original image. . . . . 18

2.2 Illustration of the rotation recognition task proposed in [6] where the network is trained to predict the rotation angle applied to the original image. . . . . 18

2.3 Samples from the Office-31 [7] dataset. Each row contains images from a different domain: starting from the first row, examples from Amazon, Webcam and DSLR domains are shown. . . . . 20

2.4 Samples from the VLCS [7] benchmark. Each row contains images from a different domain: starting from the first row, examples from PASCAL VOC 2007 [8], LabelMe [9], Caltech-101 [10] and SUN09 [11] are shown. . . . . 21

2.5 Sample images from the Office-Home [4] dataset. The dataset consists of images of everyday objects organized into 4 domains. Each row contains images from a different domain: starting from the first row, examples from Art, Clipart, Product, and Real-World domains are shown. . . . . 22

2.6 Samples from the PACS [12] dataset. Each column contains images from a different domain: starting from the left, examples from Art Painting, Cartoon, Photo, and Sketch domains are shown. . . . . 23

2.7 Samples from the VisDA2017 [13] dataset which is made by two domains: Synthetic (on the top) and Real images (on the bottom). . 23

2.8 Samples from the DomainNet [1] dataset. Each row contains images from a different domain: starting from the first row, examples from Clipart, Infograph, Painting, Quickdraw, Real, and Sketch domains are shown. . . . . 24

2.9 Samples from the SUN RGB-D [2] dataset. Each column contains images from a different domain: starting from the left, images recorded with Intel RealSense, Asus Xtion, Microsoft Kinect v1, and Microsoft Kinect v2 are shown. . . . . 25

2.10 Physical place overlapping. Some of the scene categories contain images taken in the exact same physical place with different devices, while others are recorded from different locations. We adopted the following color legend: black for a class that contains images recorded in the exact same location by multiple cameras, while blue/green/red/violet indicate classes with specific room images captured only with Kinect v2/Realsense/Xtion/Kinect v1. . . . . 26

2.11 Visualization through t-SNE [14] of the three domains of our multi-modal cross-domain scene classification testbed. Each domain is composed by images of a different camera: Kinect v2 (blue), Realsense (green), Xtion (red). . . . . 27

2.12 Image samples from Textures [15] dataset. . . . . 27

3.1 Recognizing objects across different visual domains is a difficult task that requires strong generalization abilities. Self-supervised learning can help to capture natural invariances and regularities, which can then assist in bridging large style gaps. Our multi-task approach learns to jointly classify objects and solve jigsaw puzzles or recognize image orientation, demonstrating its effectiveness in knowledge generalization. . . . . 30

3.2 The illustration of the proposed multi-task approach when using the jigsaw puzzle as self-supervised task. It starts from images from multiple domains and breaks them down into a $3 \times 3$ grid of patches which are then randomly shuffled and recomposed back into images of the same dimensions as the original ones. By using the maximal Hamming distance algorithm in [5], we establish a set of $P$ patch permutations and assign an index to each of them. Both the original and the shuffled images are fed into a convolutional network that is trained to meet two objectives: object classification on the ordered images and jigsaw classification (i.e. permutation index recognition) on the shuffled images. An analogous approach is used when using rotation recognition as self-supervised task. Note that the notation used to name the different network parts refers to Section 3.1.1. . . . 30

3.3 We perform an ablation study and analyze the effects of different hyperparameters on our approach when using Jigsaw on the Alexnet-PACS domain generalization setting. The reported accuracy is the global average over all target domains, and each run was repeated three times. The red line in the figures represents the average accuracy of our DeepAll model from Table 3.2. . . . . 38

3.4 Ablation analysis on the Alexnet-PACS DG setting when using Rotation. The reported accuracy is the global average over all the target domains, and each run was repeated three times. The red line in the figures represents the average accuracy of our DeepAll model from Table 3.2. . . . . 40

3.5 Analysis of the Jigsaw classifier on Alexnet-PACS DG setting. In the plot on the left, each axis refers to the color matching curve in the graph. . . . . 40

3.6 In CAM activation maps, yellow represents high values and dark blue represents low values. The jigsaw puzzle task is effective in identifying the most informative parts of an image for object class prediction across different visual domains. Similarly, rotation recognition can also be useful, but it tends to be less accurate in terms of localization, particularly for sketches, cartoons, and paintings. 41

3.7 Examples of RGB and Depth HHA [16] images from all the cameras within the SUN RGB-D dataset [2]. The category *classroom* contains images taken in the exact same place with Kinect v2, Realsense, and Xtion cameras, while the physical location captured with Kinect v1 is different although annotated with the same label. For the category *discussion_area* there is no room overlap: despite the shared label, each camera captured images in a different physical location. As can be noticed, the specific camera characteristics contribute to producing significant appearance differences. Best seen in color. . . 47

3.8 Our Translate-to-Adapt method for RGB-D scene recognition across domains consists of several key components: encoders (E), inter-modality decoders (D), a semantic feature extractor (F), classification and similarity evaluation heads. The encoders process both the RGB and depth images separately and then combine their features for the classification task. The decoders are responsible for converting one modality into the other, and both have the same structure but focus on different translation directions. The generated images are compared to their original versions using the semantic feature extractor and the similarity head. It's worth noting that while only the supervised source data is used in the classification task, both source and target data are used in the inter-modality generation self-supervised task. We use the notation presented in Section 3.2.1. . . . . 48

3.9 Visualizations obtained by guided backpropagation [17] that show the most important pixels used by Rel. Rot. [18] and our Tran-Adapt. 54

3.10 Qualitative comparison of real and generated images on the unseen class *corridor*. It's particularly evident the effectiveness of Tran-Adapt and its Aug version on the RGB images considering the uniform regions like walls and floors that appear smoother than in Tran-Rec. . . . . 55

4.1 Schematic representation of our approach. All the parts in gray illustrate the main blocks of the network with the solid line arrows indicating the contribution of each group of training samples to the corresponding final tasks and related optimization objectives according to the assigned blue/green/black colors. The blocks in red describe the domain adversarial classifier with the gradient reversal layer (GRL) and a weighting procedure for source samples (weight $\gamma$), which can be incorporated into our method. . . . . 60

4.2 Histogram showing the values of $\gamma$ vector which correspond to the class weight learned by PADA, SSPDA-$\gamma$ and SSPDA-PADA for the A→W domain shift. . . . . 65

4.3 Schematic illustration of our Rotation-based Open Set (**ROS**). In Stage 1, we use the source dataset $D_s$ to train an encoder $E$, a semantic classifier $C_1$, and a multi-rotation classifier $R_1$, for known/unknown separation. $C_1$ is trained using the features of the original images, and $R_1$ is trained using the concatenated features of the original and rotated images. After training, we use the prediction of $R_1$ on the target dataset $D_t$ to generate a normality score that separates the target samples into a known target dataset $D_t^{knw}$ and an unknown target dataset $D_t^{unk}$. In Stage 2, we train $E$, the semantic+unknown classifier $C_2$, and the rotation classifier $R_2$ to align the source and target distributions and to recognize the known classes while rejecting unknowns. $C_2$ is trained using original images from $D_s$ and $D_t^{unk}$, and $R_2$ is trained using the concatenated features of the original and rotated known target samples. . . . . 67

4.4 Are you able to infer the rotation degree of the rotated images without looking at the respective original one? . . . . . 70

4.5 The objects on the left may be confused. The relative rotation guides the network to focus on discriminative shape information . . . . . 70

4.6 t-SNE visualization of the target features for the W→A domain shift from Office-31. Red and blue points are respectively features of known and unknown classes. . . . . 77

4.7 Accuracy (%) averaged over the three openness configurations. . . . 79

5.1 HyMOS uses supervised contrastive learning to address all the challenges of multi-source open-set domain adaptation. We integrate style transfer into the double path contrastive logic to achieve domain-invariant representation. By balancing the class and source domains in each training batch, we achieve class-wise domain alignment. The embedding space learned by HyMOS inherently separates *unknown* target samples in low-density regions, while the *known* samples are located close to their corresponding class cluster and can be easily used in self-training for further adaptation. . . . . 83

5.2 Schematic illustration of HyMOS (best viewed in color). We use the same notation adopted in Algorithm 1, please refer to it to follow the flow of the method. . . . . 85

5.3 Illustration of the distances used to set the class prediction and the self-training procedure. . . . . 88

5.4 Left: analysis on the dynamic threshold $\alpha$ at different training iterations. Right: performance of HyMOS and ROS [19] at different openness ($\mathbb{O}$) levels. . . . . 90

5.5 Comparison between standard supervised learning and relational reasoning representation learning. Standard supervised learning focuses on recognizing known object classes, while relational reasoning representation learning aims to learn a measure of semantic similarity among image pairs. We show that relational reasoning is particularly well-suited for semantic novelty detection tasks. Our pre-trained large-scale relational model can be applied to these tasks without the need for a fine-tuning phase on the known classes specific to the task. 97

5.6 Schematic illustration of the training phase of ReSeND. The features extracted from a pair of images are provided as input to our relational module. It consists of a transformer encoder that elaborates over a tuple composed of the sample pair and of a learnable label token. The output corresponding to this last token is finally provided as input to a semantic similarity head that predicts the sample resemblance. . . 99

5.7 Open-Set DG setting . . . . . 106

5.8 Loss trend for the probability of the correct class. . . . . 108

5.9 AUROC comparison with ReSeND trained for classification through Cross Entropy Loss or for Regression via MSE. OH stands for Office-Home. SS and MS indicate respectively the Single- and Multi-Source settings. . . . . 108

A.1 Accuracy (%) averaged over the three configurations designed for each degree of openness considered: with 40, 25, 10 and 5 known classes. The table reports in details the values used to prepare the plots 143

A.2 Hyper-parameter analysis . . . . . 144

B.1 Qualitative analysis of the Ar, Pr, Rw $\rightarrow$ Cl case in the Office-Home dataset. The red dots represent the source domain, the blue dots represent the known samples of the target domain, and green dots represent the unknown samples of the target domain. HyMOS 20k: source balancing and style transfer already favor a good alignment of most known target classes with their respective source known clusters. HyMOS 40k: self-training further move the target known samples towards the respective source clusters, while the unknown samples remain in the regions among the clusters. The zooms demonstrate how the neighborhood of a known target sample (bike) and an unknown target sample (speaker) changes during training. . . . . 148

B.2 Sensitivity analysis for the temperature value $\tau$ on Office-Home. . . 148

B.3 Performance trend increasing the number of image pairs . . . . . 152

# List of Tables

2.1 Number of images in the considered classes. . . . . 25

3.1 We evaluated the performance of our model on different tasks and architectures using domain generalization (DG) classification accuracy. The column titles indicate the target domain. The best results are highlighted in bold. The top part of the table shows the results of self-supervised pretraining on Imagenet, followed by fine-tuning on the source. Methods that use patch-based networks are indicated by (p) while those that use whole-image networks are indicated by (w). The bottom part of the table shows the results of supervised pretraining on ImageNet followed by the multi-task combination of self-supervised objectives and supervised fine-tuning. The numbers reported correspond to the accuracy (%) averaged over three runs. . 34

3.2 We compare our approach with state-of-the-art DG methods on the PACS dataset. The column titles indicate the target domain. The table shows the hyperparameters used for each experiment, obtained through source cross-validation. The best results are highlighted in bold. The numbers reported correspond to the accuracy (%) averaged over three runs. . . . . 36

3.3 We compare our approach to state-of-the-art DG methods on the VLCS dataset. For more information on the notation used, please refer to Table 3.2. The numbers reported correspond to the accuracy (%) averaged over three runs. . . . . 37

3.4 We compare our approach to state-of-the-art DG methods on the Office-Home dataset. For more information on the notation used, please refer to Table 3.2. The numbers reported correspond to the accuracy (%) averaged over three runs. . . . . 37

3.5 Accuracy on *Office-Home* under single-source DA setting. The top result is highlighted in bold. The numbers reported correspond to the accuracy (%) averaged over three runs. . . . . 42

3.6 Multi-source Domain Adaptation results on PACS. The numbers reported correspond to the accuracy (%) averaged over three runs. . 43

3.7 Cost analysis on PACS with AlexNet in DG setting. Hardware - CPU: Intel(R) Core(TM) i7-9800X CPU @ 3.80GHz, GPU (x1): Nvidia TITAN Xp. . . . . 44

3.8 Accuracy (%) across domains for single modality. The performance drop shows the effect of the domain shift. The confusion matrices for the K→X case also reveal that the behavior across domains varies for the two modalities: classes 2 (classroom) and 7 (kitchen) are the ones most affected by the domain shift for RGB and depth modalities, respectively. . . . . 51

3.9 Accuracy (%) of several methods for RGB-D domain adaptation. Top results in bold. The confusion matrices show the K→X per-class results for the ResNet-18 baseline and Tran-Adapt (Fusion++). . . . 52

3.10 Number of samples in extra classes considered for the missing modality prediction. . . . . 55

3.11 Pixel-to-pixel L2 distance between real and generated images from unseen classes of the target domain. Top results in bold (the lower the better). . . . . 56

3.12 Cost analysis on SUN RGB-D with ResNet-18 in DA setting. Hardware - CPU: Intel(R) Core(TM) i7-9800X CPU @ 3.80GHz, GPU (x1): Nvidia TITAN Xp. . . . . 56

4.1 Accuracy (%) in the PDA setting on Office-31 dataset (source: 31 classes, target: 10 classes). The results are obtained by averaging over three repetitions of each run. With * we indicate ten-crop testing. 64

4.2 Accuracy (%) in the PDA setting on VisDA2017 dataset (source: 12 classes, target: 6 classes). The results are obtained by averaging over three repetitions of each run. . . . . 65

4.3 Cost analysis on Office-31 with ResNet-50 in PDA setting. Hardware - CPU: Intel(R) Core(TM) i7-9800X CPU @ 3.80GHz, GPU (x1): Nvidia TITAN Xp. . . . . 66

4.4 Accuracy (%) averaged over three runs of each method on Office-31 dataset using ResNet-50 and VGGNet as backbones . . . . . 76

4.5 Accuracy (%) averaged over three runs of each method on Office-Home dataset using ResNet-50 as backbone. . . . . 76

4.6 Reported vs reproduced OS accuracy (%) averaged over three runs . 77

4.7 Ablation Analysis on Stage I and Stage II . . . . . 78

4.8 Cost analysis on Office-31 with ResNet-50 in OSDA setting. Hardware - CPU: Intel(R) Core(TM) i7-9800X CPU @ 3.80GHz, GPU (x1): Nvidia TITAN Xp. . . . . 80

5.1 Comparison with existing open-set and universal domain adaptation approaches. HPs indicate the hyperparameters, $|C_s|$ the number of source categories, $|S|$ is the number of source domains. Note that synthesizing new samples is a time-consuming operation and any validation procedure requires at least a dedicated per-dataset tuning. 83

5.2 Results averaged over three runs for each method on the DomainNet, Office31, and Office-Home datasets. . . . . 90

5.3 Average performance (HOS) when changing the train-time multiplier $\alpha_m$ to the self-paced threshold $\alpha$. . . . . 91

5.4 Ablation Study, HOS results. . . . . 91

5.5 Multi-Source Closed-Set (Accuracy) and Universal Domain Adaptation (HOS) performance on DomainNet. . . . . 92

5.6 Cost analysis on Office-31 with ResNet-50 in MOSDA setting. Hardware - CPU: Intel(R) Core(TM) i7-9800X CPU @ 3.80GHz, GPU (x1): Nvidia TITAN Xp. . . . . 94

5.7 Intra-Domain analysis. Best result in bold and second best underlined. 103

5.8 Cross-domain analysis. Top: single-source results, Bottom: multi-source results. We consider the PACS dataset with all the possible combinations of source/target as support/test sets. Best result in bold and second best underlined. . . . . 104

5.9 Comparison with finetuning-based state-of-the-art OOD methods. Best result in bold and second best underlined. . . . . 105

5.10 Open-Set DG experiments. . . . . 106

5.11 Results obtained with different configurations of the relational module. We compare ReSeND with handcrafted feature aggregation strategies for sample pairs. . . . . 107

5.12 Cost analysis on PACS Single-Source experiment. Hardware: i) Inference: CPU: Intel(R) Core(TM) i7-9800X CPU @ 3.80GHz, GPU (x1): Nvidia TITAN Xp ii) Training 16 units: CPU: Intel(R) Xeon(R) Silver 4210 CPU @ 2.20GHz, GPU (x1): Tesla V100 SXM2 16GB . . . . . 108

A.1 The OS accuracy (%) reported in the papers compared to the one obtained in our reproducibility study. The results show the average over three runs and across all sub-domains of Office-31 and Office-Home with the indicated backbones. . . . . 141

A.2 Analysis on the use of self-supervised tasks for the two stages of the method and further ablation. . . . . 144

# Chapter 1

# Introduction

One of the biggest challenges in the current technological revolution is building reliable autonomous machines able to deal with the huge complexity and variability that characterizes our world and that requires an effort for continual adaptation [20]. This effort has been largely internalized by living beings throughout the whole evolutionary process. Indeed, neuroscientists and psychologists agree that one of the key traits of *intelligence* is the ability to *adapt* in response to different situations while *autonomously* learning knowledge patterns [21, 22]. However, current *Artificial Intelligence* (AI) models still struggle in dealing with discrepancies between training and deployment conditions.

We know from daily experience that when exploring the world it is natural to encounter unknown object categories. Still, for a vision-based object recognition model this *category shift* would lead to dramatic failures. Similarly, a change in visual appearance simply caused by a difference in illumination condition, point of view, or style (e.g. photograph vs sketch) defines a *domain shift* (examples in Figure 1.1) that largely reduces the original abilities of the recognition model. Even powerful Deep Neural Networks (DNNs) show performance reduction in these scenarios and it is possible to identify two main causes. On one side their prediction tends to be overconfident, thus new samples are assigned to one of the known object categories with low uncertainty [23]. On the other, the models lack generalization and incur in negative transfer by focusing too much on local and texture appearance, rather than on global and shape-related semantic information which is shared across domains [24].

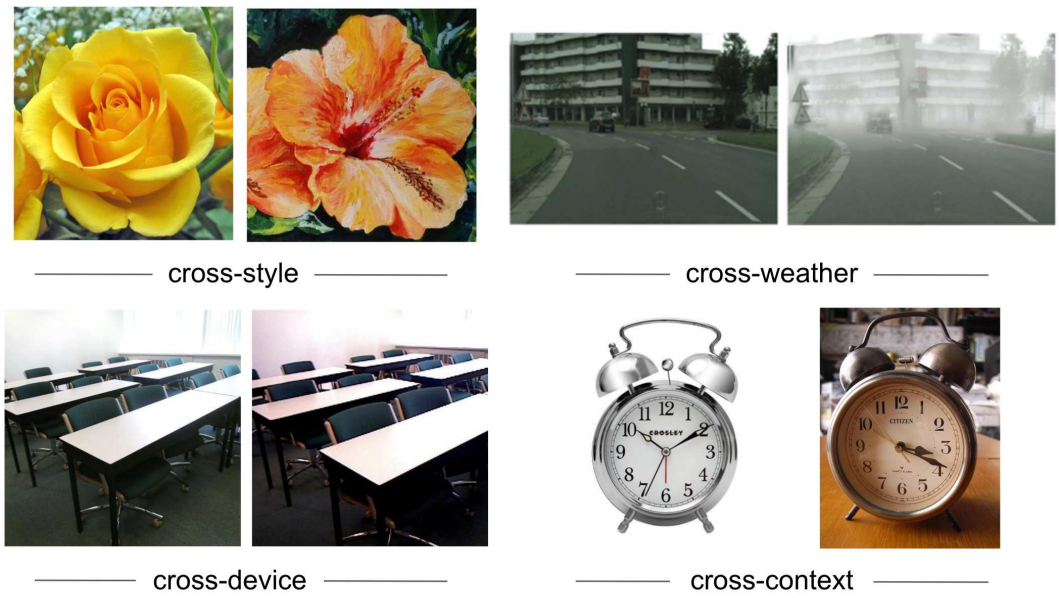



Fig. 1.1 Examples of domain shift taken from DomainNet [1], SUN RGB-D [2], Cityscapes [3] and OfficeHome [4] datasets.

With the aim of getting inspiration from the intelligence of living beings and improving the current limitations of artificial learning systems, it is helpful to resort to developmental science. Different studies highlight how children gain an understanding of the world from both *Supervised* and *Self-Supervised* learning: while parents or teachers may provide guidance on a few important concepts, children are able to fill the gaps on their own by discovering patterns that may not have been covered through traditional education. Kids mainly play to experience how the world reacts to certain stimuli, inventing new activities with apparently random rules. Moreover, many children's games are fully self-supervised tasks based on discovering object regularities as jigsaw, shape, and matching puzzles (see Figure 1.2).

Self-supervised learning has been used in *Computer Vision* (CV) for the first time in [25–27]: it was proposed as an alternative to supervision with the advantage of avoiding costly data annotation and leveraging large amounts of freely available unlabeled data. Several other works followed the first seminal ones, showing how self-supervised pre-trained models were able to capture general-purpose feature embeddings that could be inherited for a variety of downstream tasks [5, 28–32, 27]. However, none of them put the spotlight on how self-supervision could complement supervised learning in improving domain generalization or reducing model over-

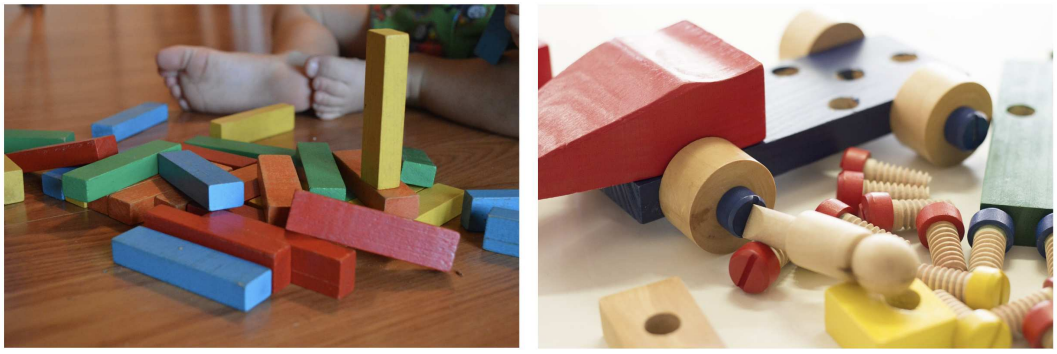


Fig. 1.2 The greatest part of the baby's toys is designed to give children the ability to autonomously find patterns that don't require any supervision.

confidence. This thesis summarizes the work done by the author toward these goals.

Specifically, the *first part* of the manuscript presents multi-task solutions optimized for object and scene-supervised recognition together with different self-supervised objectives. Here the goal is cross-domain learning both in single and multi-modal scenarios, but in a simplified closed-set setting so without novel categories at test time. In the *second part* of this thesis, we investigate the more challenging partial and open-set frameworks where category shift appears together with domain shift. We discuss how self-supervision is able to provide insights on the novelty of a test sample and significantly reduce prediction over-confidence. Finally, in the *third part*, we argue how the *analogical reasoning*, which is at the core of several self-supervised tasks, can be directly formalized in supervised learning models to get improved performance. Specifically, we consider contrastive and relation-based learning approaches presenting their effect on the open-set task and semantic novelty detection in multi-domain challenging conditions.

Before providing more details on the thesis contributions we present a brief problem statement to be considered as a glossary of the main terms that will be used throughout the whole manuscript.

## 1.1 Problem Statement

The goal of cross-domain learning algorithms is to transfer knowledge from a labeled dataset (*source* domain) to an unlabeled one (*target* domain) generally characterized by differences in the appearance (*domain shift*) and/or non-overlapping label sets (*category shift*).

If the target domain is available during training but fully unsupervised, the setting is called *Unsupervised Domain Adaptation* (UDA). Formally, in UDA, the source domain $S = \{x_j^s, y_j^s\}_{j=1}^{N^s} \sim p_s$ (with $N^s$ source samples) and the unlabeled target domain $T = \{x_j^t\}_{j=1}^{N^t} \sim p_t$ (with $N^t$ target samples) are drawn from different data distributions $p_s \neq p_t$ . If the possibility to access target samples, even unlabeled, is denied, the setting is called *Domain Generalization* (DG). In both cases, the objective is to build a source model with high recognition performances on the target domain. Motivated by the unrealistic assumption that all the available labeled samples come from the same distribution, both UDA and DG can be extended to the *Multi-Source* scenario where more labeled source domains $S = \{S_1, S_2, \ldots, S_L\}$, all drawn from different data distributions $p_{i=1,\ldots,L}$, can be used during training.

The way in which the source(s) label set $Y_s$ and the target label set $Y_t$ overlap defines the specific scenario we are dealing with. The simplest situation is the *Closed-Set Domain Adaptation* (CSDA) scenario with perfect overlap between the source and target label sets ($Y_s = Y_t$). The *Partial Domain Adaptation* (PDA) setting is characterized by a target domain with a number of categories lower than the source domain ($Y_t \subset Y_s$). Finally, when $Y_s \subset Y_t$, the target covers additional classes which are considered *unknown*. In that case, there can be two different settings: *Semantic Novelty Detection* (SeND) if we are only interested in discriminating between known and unknown samples and *Open-Set Domain Adaptation* (OSDA) when instead we also want to classify the target samples belonging to the known categories. In PDA, SeND and OSDA it's important to address not only the domain shift, as in CSDA but also the category shift, which makes the problem even more challenging. Indeed, ignoring this aspect and trying to force the source and target data to match, inevitably produces negative transfer causing worse performance for any adaptive method compared to its non-adaptive version.

## 1.2 Contributions

In this thesis, we deal with the problem of generalization over *domain shifts* and/or *category shifts* between the training and test sets in the context of visual recognition. The leitmotif of all the proposed solutions is the joint use of supervised and self-supervised learning. In particular, the contributions of this thesis are the following:

- **the first end-to-end multi-task architecture that learns simultaneously to recognize categories and to solve self-supervised tasks [33–35] enhancing its generalization ability**. We focused on the task of recovering an original image from its shuffled parts [5], predicting the object orientation [6] and inter-modal RGB-D translation [36]. We show how these tasks can be re-purposed as side objectives to be optimized jointly with object classification helping the model to focus on object regularities shared across domains. We proved the effectiveness of the proposed solution in the CSDA and DG settings. In [35] we also proposed a benchmark testbed for scene recognition to analyze the problem of domain shift across devices: although often overlooked, the decrease in performance due to data acquired with a different instrument is proved to be a critical issue in real-world applications.

- **self-supervised-based solutions to tackle open-world scenarios where also category shift holds [34, 19]**. Motivated by the remarkable performance gain obtained in CSDA and DG, we employ self-supervision for both PDA and OSDA confirming its great effectiveness across domains. In particular, for OSDA we propose the first approach that exploits self-supervision to tackle both unknown detection and domain alignment. A rotation predictor helps in distribution alignment and guides the model to focus on domain-agnostic properties of the known objects useful to identify unknown samples in the target. Together with a benchmark and reproducibility study of existing OSDA approaches, in [19] we also propose a new metric that evaluates, at the same time, the ability to recognize the known classes and to reject the unknown ones [19]. Indeed, all the existing OSDA metrics only consider, separately, the closed set and open set performance not allowing to properly assess the overall model.

- **supervised solutions based on self-supervised logic [37, 38]**. Despite self-supervision could be decisive in dealing with open-world scenarios preventing the use of category labels, its learning paradigm can be extremely helpful and

sometimes preferred, also in the supervised version [39]. Specifically, we consider contrastive learning [40, 41] and relational reasoning tasks [42]. In [37] we propose to tackle all the challenges of the Multi-Source OSDA setting through a single supervised contrastive loss function, also overcoming the main limitations of the existing approaches. In [38] we propose a representation learning approach based on relational reasoning to obtain a semantic similarity score that indicates the probability that two samples belong to the same class or to different ones (SeND task).

## 1.3 Outline

Chapter 2 presents an overview of the current literature and introduces the datasets used for the experimental evaluation. In particular, Section 2.1 presents the relevant works related to the problem of domain shift and category shift between the training and test set. Section 2.2 focuses on self-supervised learning considering the oldest and the most recently developed techniques. Finally Section 2.3 introduces the datasets used.

Chapter 3 shows in detail the two approaches proposed as solutions to the domain shift problem under the closed set assumption. Section 3.1 introduces **JiGen**, the first multi-task approach that exploits self-supervision to deal with the problem of domain shift. In particular, the section presents an extensive analysis of the effectiveness of jigsaw puzzle and rotation recognition as auxiliary self-supervised tasks for visual recognition. Section 3.2 describes **TranAdapt** that builds over the same intuition of JiGen but proposes the inter-modal RGB-D translation as self-supervised task to promote adaptation.

Chapter 4 presents the techniques proposed to tackle the problem of category shift in combination with domain shift. Section 4.1 describes how to extend JiGen to deal also with the PDA setting. Both jigsaw and rotation recognition are tested as self-supervised tasks showing their effectiveness in dealing with the problem of the partial overlap between the source and target label sets. Section 4.2 introduces **ROS** where rotation recognition is exploited for both domain shift and target unknown detection in the OSDA setting.

Chapter 5 moves from multi-task solutions to representation-based approaches to cope with the problem of category shift and domain shift, leveraging two popular self-supervised learning tasks in their supervised variants. Section 5.1 presents **HyMOS** where a supervised contrastive learning loss is optimized to solve all the challenges of the MOSDA setting at once. In Section 5.2, is presented **ReSeND**, a relational reasoning-based approach to deal with the SeND task.

Finally, the thesis ends with a summary of the main conclusions, followed by a discussion on open issues and future perspectives.

## 1.4 Publications and Reports

*The list below presents the author's publications in chronological order. Please note that some of the papers marked with an asterisk (*) are not included in this thesis.*

- Carlucci, F. M., D'Innocente, A., Bucci, S., Caputo, B., & Tommasi, T.
  *Domain generalization by solving jigsaw puzzles.*
  IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 2229-2238) (**CVPR 2019**, **Oral**).

- Bucci, S., D'Innocente, A., & Tommasi, T.
  *Tackling partial domain adaptation with self-supervision.*
  International Conference on Image Analysis and Processing (pp. 70-81). Springer, Cham (**ICIAP 2019**, **Best Paper Award**).

- Bucci, S., Loghmani, M. R., & Tommasi, T.
  *On the effectiveness of image rotation for open set domain adaptation.*
  European Conference on Computer Vision (pp. 422-438). Springer, Cham (**ECCV 2020**).

- (*) D'Innocente, A., Borlino, F. C., Bucci, S., Caputo, B., & Tommasi, T.
  *One-shot unsupervised cross-domain detection.*
  European Conference on Computer Vision (pp. 732-748) 2020. Springer, Cham (**ECCV 2020**).

- Ferreri, A., Bucci, S., & Tommasi, T.
  *Multi-Modal RGB-D Scene Recognition Across Domains.*
  IEEE/CVF International Conference on Computer Vision Workshops (pp. 2199-2208) (**ICCVW 2021**).

- Bucci, S., D'Innocente, A., Liao, Y., Carlucci, F. M., Caputo, B., & Tommasi,T.
  *Self-supervised learning across domains.*
  IEEE Transactions on Pattern Analysis and Machine Intelligence, 44(9), 5516-5528 (**TPAMI 2021**).

- Bucci, S., Borlino, F. C., Caputo, B., & Tommasi, T.
  *Distance-based Hyperspherical Classification for Multi-source Open-Set Domain Adaptation.*

IEEE/CVF Winter Conference on Applications of Computer Vision (pp. 1119-1128) (**WACV 2022**).

- (*) Borlino, F. C., Bucci, S., & Tommasi, T.
  *Contrastive Learning for Cross-Domain Open World Recognition.*
  IEEE/RSJ International Conference on Intelligent Robots and Systems (**IROS 2022**).

- Borlino, F. C., Bucci, S., & Tommasi, T.
  *Semantic Novelty Detection via Relational Reasoning.*
  European Conference on Computer Vision (pp. 183-200). Springer, Cham (**ECCV 2022**).

- (*) Bucci, S., Centeno, J. I. D., Gaffinet, B., Liang, Z., Bickel, V., Moseley, B., & Olivares-Mendez, M.
  *Thermophysical Change Detection on the Moon with the Lunar Reconnaissance Orbiter Diviner sensor.*
  Machine Learning and the Physical Sciences Workshop at Neural Information Processing Systems (**NeurIPSW 2022**)

*The following work is currently under submission in a peer-reviewed conference:*

- Iurada, L., Bucci, S., Hospedales, T. M., & Tommasi, T.
  *Fairness meets Cross-Domain Learning: a new perspective on Models and Metrics.*

# Chapter 2

# Related Works

*The first Section of this Chapter reviews the current literature on Learning Across Domains. The second Section is dedicated to Self-Supervised Learning which is the main solution used in this thesis to address the domain shift and category shift problems. The Chapter concludes with the introduction of all the datasets used in this thesis.*

## 2.1 Learning across Domains

**Usupervised Domain Adaptation** [20] accounts for the difference between training and test data by considering them as drawn from two different marginal distributions. In the UDA literature, we can identify a few main research directions. *Discrepancy-based* methods define and minimize a metric that measures the distance between source and target data in the feature space: many approaches minimize the Maximum Mean Discrepancy (MMD) [43–47], the Wasserstein distance [48, 49], or other statistical moment matching constraints [50–52]. To minimize the domain shift other strategies also exploit feature normalization [53] or even dedicated batch normalization layers [54, 55]. A considerable number of UDA approaches are based instead on *Adversarial learning* techniques [56–60, 57]. In that case, the objective is to adversarially train a domain discriminator and a backbone so that it converges to a solution that causes the domain discriminator to be unable to distinguish between the source and target data. Those strategies exploit the idea at the basis of the Generative Adversarial Network (GAN) [61] that can be also directly applied to match domains at pixel level transferring the style of the source to the target and vice-versa [58, 62–65]. Given that simply encouraging domain invariance does not guarantee the extraction of discriminative features from the target domain [66], recent approaches have proposed class-aware adversarially-based adaptation techniques to better align the class-conditional distributions between the source and target domains [67–71]. *Feature disentanglement* based approaches aim at separating the deep features into domain-specific and category-specific to extract only relevant factors of data variation [72–74]. Alternatively, other effective and popular strategies consist in measuring the uncertainty of the source classifier on the target data through the *entropy loss* which is minimized during training [75–78], or *pseudo labeling* (also known as self-training) using the output of the source model as coarse annotation on the target [79, 80]. *Self-supervised learning* based techniques [33, 34, 81, 82] tackle UDA by adding auxiliary self-supervised tasks to improve the generalization ability of the network. Given the unsupervised nature of self-supervised learning, the unlabeled target data can be directly used to solve this task helping the network to produce a robust representation for the main supervised task. *Our work pioneered this research direction. We published the first work proposing the multi-task supervised and self-supervised approach that will be described in detail in Section 3.1.*

Out of the variety of UDA solutions developed in the last years, the vast majority deal with a single visual modality. However, in many practical scenarios, more signals can be jointly collected and may support each other to improve model generalization as they are not all equally affected by domain shift. Few works have considered RGB and Depth modalities together in different settings. In [83, 84] RGB and Depth are respectively source and target, while [85] deals with a multi-modal source (RGBD) and a single modal target (RGB). In [86, 62] the depth information is used as an additional input channel for source and target, extending standard RGB domain adaptation methods to the RGB-D case. The importance of exploring the inter-modal relation for adaptive learning has been highlighted in [18]. *With our work we showed how self-supervision is able to enhance domain robustness also in multi-modal settings through cross-modality reconstruction as discussed in Section 3.2.*

In realistic conditions, source data may also originate from multiple distributions (**Multi-Source** DA) [87, 88]. One of the main challenges when working with multiple sources is to create a feature space that is both highly discriminative and capable of aligning all the domains. Based on the assumption that the target distribution can be approximated as a mixture of the source distributions [89, 90], many MDA approaches combine the predictions of source classifiers via re-weighting [91, 49, 92–94]. In particular, in [91] the combination of the distributions is treated as a Difference of Convex (DC) programming deriving a domain generalization bound. Other *Discrepancy-based* approaches aim to reduce the domain gap among the sources by adding minimization constraints on some measures of discrepancy as MMD [95], L2 distance [96], and moment distance [1] or focusing on prototype-based alignment strategies [97, 98].

In **Domain Generalization**, as opposed to UDA, it's not allowed to use the target data during training. When only one source domain is available, regularization strategies are usually applied. They include label smoothing [99], strategic dropout based on the gradient observation [100], tailored model selection [101, 102] or data-augmentation solutions to increase the variability of the training set [103, 104]. In the **Multi-Source** setting the main objective is to distill the most useful and transferrable general knowledge across multiple sources in the hypothesis that it would be transferable also at deployment time with any unseen target domain. Many previous studies have focused on *model-based* approaches to neglect domain-specific features. They include both shallow and deep models learned through multi-task

learning [105], low-rank network parameter decomposition [12, 106, 107], domain-specific aggregation layers [108] or with source model weighting [109]. Some alternative approaches focus on finding a *feature-level* representation that can capture shared information across multiple domains. Proposed strategies include the use of autoencoders [110, 111], learning objectives to project images of the same classes nearby regardless of the source domain [112, 113]. Methods that operate at the *data-level* aim to increase the variability of the source samples, with the goal of expanding the range of data represented and potentially improving the model's performance on the target domain. One of the earliest examples of data-augmentation-based approaches for DG is Domain Randomization [114] where random rendering is added to the samples generated by several simulated environments. In [115] the authors propose a method for augmenting the source domain by applying domain-specific perturbations to the existing data points. Style-transfer techniques are also highly effective in producing strong data augmentation for DG as discussed in [116]. *Meta-Learning* solutions [117–120] follow the learning paradigm proposed in [121] where the sources are split into meta-train and meta-test: the model is learned on the meta-train but the parameter update is also guided by the loss with respect to the meta-test that simulated the deployment of the model in a novel target scenario. Analogous to UDA, *Self-Supervised Learning* can also be applied in DG. The source/s can be used to train the auxiliary self-supervised task to obtain a robust model that focuses only on the class-specific features disregarding the domains. *As in UDA, the use of Self-Supervised learning in DG is one of the novel contributions of this thesis presented in Section 3.1.*

The basic UDA and DG frameworks assume that all the domains share exactly the same label set. However, this **Closed-Set** setting is too optimistic: given that the target is unlabeled (or completely unseen). We cannot be sure that its semantic content perfectly matches that of the source.

**Partial Domain Adaptation** relaxes this assumption and allows the target to cover only a subset of the source categories: it becomes important to drive the learning by considering only the source samples with shared labels during the adaptation process. A popular technique for PDA consists in *source samples re-weighting* based on the probability that each sample belongs to one of the categories shared with the target domain [122–125]. SAN [122] and PADA [123] exploit a standard domain-adversarial UDA technique [56] while the contribution of each source sample is estimated through the evaluation of the target domain on the

source classifier producing statistics on the classes distributions. Differently, in IWAN [124], where each domain has its own feature extractor, the source sample weights are obtained from the domain recognition model rather than from the source classifier. Lastly, TWIN [125] exploits two different classifiers to reduce the domain shift enforcing a minimal disagreement between their predictions on the target. It computes the source weights following PADA [123] but averaging over the output of both the source classifiers. Besides the approaches based on weighting schemes, an effective solution in PDA, as in UDA, consists in *aligning the feature norm* of source and target [53]: without any heuristic weighting mechanism, the model has been proven to be robust to negative transfer. Moreover a *reinforcement learning*-based solution is proposed in [126], while *Liang et al.* [127] propose to augment the small target domain to match the large source domain through an uncertainty-aware approach. In [128] the distributional difference between outlier and shared classes is maximized with the goal of improving alignment between the shared classes across domains. The strategy proposed in [129] involves finding implicit semantic concepts that are shared between source and target and adjusting their respective distributions so that they more closely match each other. Finally, using a self-supervised auxiliary task on the target domain may help the model to focus only on the categories shared with the source as discussed in [130]: *in Section 4.1 we present this contribution.*

**Open Set Domain Adaptation** refers to the scenario where the target domain contains all the source categories but also an additional set of classes that should be considered *unknown*. Here the challenges are two: correctly classify the target samples that belong to the known categories and detect the unknown ones. The earlier approaches proposed in OSDA were all based on *Adversarial Training* [131, 24, 132]. In OSBP [131] a classifier is trained to obtain a large boundary between source and target samples, while the feature generator is trained to move the target samples away from this boundary. *Liu et al.* [24] adopt a weighting mechanism for known-unknown target separation still considering a standard adversarial approach for the adaptation [56]. Besides adversarial techniques, recently [133] introduced a *self-ensembling* based method to reduce the disagreement between the class prediction made with the source classifier and the underlying target cluster distribution. Other strategies consist in *emulating the unknown samples* by suppressing class-specific activation feature maps [134] or are based on *Graph-based* solutions [135, 136]. Another direction explored in OSDA is the use of *Self-Supervised Learning* [19, 137]. For instance, we showed in [19] that, by learning to recognize object orientations a

model gets relevant knowledge to evaluate sample similarity. Thus it can be useful to draw conclusions on whether a certain instance belongs to a known or unknown class regardless of its visual domain. *The dual use of Self-Supervision for both unknown detection and DA is a contribution of this thesis that we introduced in Section 4.2.*

In **Multi-Source** OSDA (MOSDA), besides having several source domains, the target domain contains also *unknown* categories as in OSDA. There are only a few works proposed to deal with this setting [138, 139, 82]. MOSDANET [138] incorporates a clustering objective into a standard supervised classification model to maximize the similarity among samples of the same class but different domains. The adaptation is then performed through adversarial learning [56] with a margin loss to penalize cases with a small difference in known and unknown prediction output. NENO [139] proposes a contrastive learning approach considering a decay multiplier for the loss function when potential known target samples are included in the training. Finally, there are just a few works that explore the problem of category shift in DG like [140] that propose a meta-learning-based solution for the **Open Set Domain Generalization** (OSDG) setting where the label sets of the sources may not overlap and the target could have unknown classes. *In Section 5.1 we present a contrastive learning-based solution for the MOSDA problem [37] where a style transfer technique for source-target adaptation is adopted.*

**Semantic Novelty Detection** (SeND) [141] together with *Anomaly Detection* [142–144] fall under the *Out-Of-Distribution* (OOD) literature whose goal is to identify whether a given test sample belongs to the same *distribution* of the training data or not. In this thesis, we consider the SeND task that focuses on conditional distributions, meaning on differences in the label sets: a sample is defined anomalous if it doesn't belong to any of the categories of the training set ignoring the possible presence of train-test domain-shift. A very basic approach adopted is the Maximum-Softmax Probability (MSP) [145, 146] which consists in applying a threshold on the highest score produced by a trained known-class classifier. Several enhancements to this naïve strategy have been proposed, such as those presented in [147–149] or in [150] where the gradients produced by the network are used to estimate prediction uncertainty. *Generative-based approaches* consist in training a model to reconstruct samples from the reference known classes: at test time, the reconstruction error defines the novelty score [151–156]. Other techniques emulate OOD data by synthesizing them [157–159] or by using external datasets as a source of anomalies during training [160–162]. The degree of normality of test samples is computed

by measuring their distance from the training data by using specific embeddings or metrics [163, 164]. *Self-Supervised-based* approaches remove the focus from the labels and can be effective also in this context since they capture data patterns and relationships [6, 5, 165–167, 137, 168–171]. *In Section 5.2 we propose a relational reasoning-based solution whose model outputs a measure of semantic similarity for unknown detection.*

## 2.2 Self-Supervised Learning

Despite unsupervised data do not come with human annotation they still contain a large amount of structural information that can be captured by a self-supervised model that solves what is known as pretext task. Depending on the number of instances to define it, self-supervised learning can be divided into two main groups: relation- and transformation-based. Relational-based approaches aim at increasing similarity between a sample and its augmented version (positive pair) while minimizing the similarity between that sample and the other training samples (negative pairs) [40, 172, 173]. In transform-based self-supervised tasks instead, a part of the data information is discarded to let then the model recover it. Examples are image completion [174], colorization [31, 175], the relative position of patches [26, 5], rotation recognition [6] and many more [165–167, 176, 177]. It has been shown that the self-supervised learned embedding captures general data knowledge and can be used to improve performance in a wide range of downstream tasks through transfer learning (i.e. fine-tuning) on a limited amount of annotated samples [178–180]. Generally, the quality of a self-supervised learned embedding is evaluated in function of the final performance of the downstream task.

**Jigsaw Puzzle** is a basic pattern recognition problem that consists in recovering an original image from its shuffled parts. *Freeman et al.* [181] first introduced it in computer science followed by [182] which proposes to solve it not only based on the shape of the objects but also looking at the appearance information. Over the years many variations of the standard Jigsaw Puzzle game have been proposed like predicting, together with the correct permutation, if all the patches of the image are present (completeness) and/or if there were parts from other images. Algorithms designed to solve Jigsaw Puzzles have wide applications in many areas like computer graphics [183, 184], archaeology [185, 186], biology [187] and visual representations (see Figure 2.1) [28, 26, 5].

The strategies that have been proposed to solve the task can be roughly divided into greedy methods, based on sequential pairwise matches, and global methods that aim to find a solution minimizing a global compatibility measure over all the patches. Examples of approaches that fall in the first group are [188] that proposes a minimum spanning tree algorithm or [186] that considers the shortest path on a graph that models the puzzle structure. Among the global methods that consider all the patches together, [189] use Markov Random Field formulations, and [184] builds

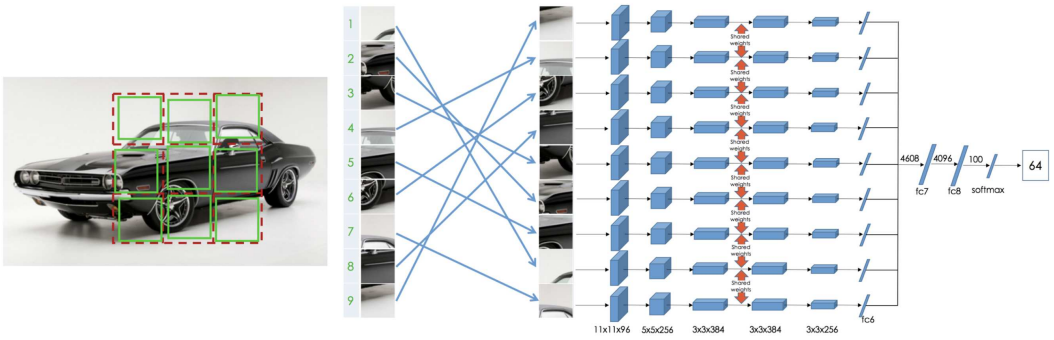



Fig. 2.1 A schematic illustration of the Context Free Network (CFN) proposed in [5] trained to predict the index of the permutation applied to the original image.

on genetic algorithms. More recently [28] solved a bi-level optimization problem to recompose the original image and [190] relies on a transformer-based architecture.

**Rotation Recognition** [6] consists in changing the orientation of an image and asking a model to predict the rotation angle that would bring it back (see Figure 2.2). Originally developed as a representation learning approach, it has been used in many applications including anomaly detection [137], closed-set domain adaptation [191], action recognition [192] and medical images analysis [193, 194].

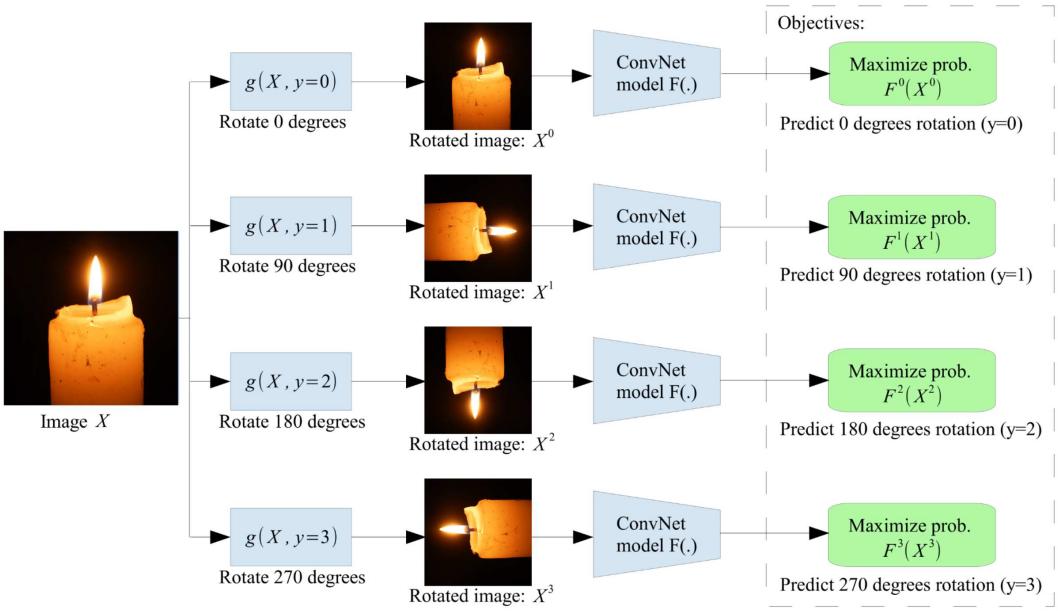



Fig. 2.2 Illustration of the rotation recognition task proposed in [6] where the network is trained to predict the rotation angle applied to the original image.

**Contrastive Learning** is an instance discrimination technique [195] that considers every instance as an independent class. Its aim is to maximize the agreement between many augmentations of the same sample while separating different instances as much as possible. SimCLR [40] and MoCo [172] represent two of the most cited contrastive learning-based approaches. The former needs to be trained with a large batch size while the latter maintains a momentum encoder and a limited queue of previous samples. Current research is trying to improve the contrastive learning formulation with optimized negative sampling [173] or choosing the best augmentation views [196, 197]. Although originally designed as a self-supervised task, more and more methods incorporate supervision [41, 198] getting the best of both worlds for several tasks like novelty detection [169], cross-domain generalization [199] or few-shot classification [200].

**Relational Reasoning** is a highly developed ability of the human being's brain. It has been defined in the ML community as learning a function whose output estimates the relationships between a set of objects. Largely employed for the combination of language and vision [201–203], it has found applications in many area like reinforcement learning [204–206], object detection [207], graph networks [208], and few-shot learning [209, 210]. Recently, it has been shown how effective this task can be when applied as a self-supervised task for representation learning [42], with surprisingly better results than contrastive learning-based strategies [40, 211].

## 2.3 Datasets

Several datasets have been created with the aim of covering a wide variety of visual domains. We present below the ones used in the rest of the thesis.

**Office-31** [7] has been the standard benchmark for testing DA methods for several years. It consists of three domains (Amazon (A), Webcam (W), Dslr (D)) and 31 categories. Specifically, the images are photos of daily office objects, the domain Amazon is made by images collected from *amazon.com* website, Webcam and DSLR are two different types of devices used to record the same objects in an office. As can be seen in Figure 2.3 both the resolution/properties of the devices used for recording and the format of the images on the web cause the domain shift.

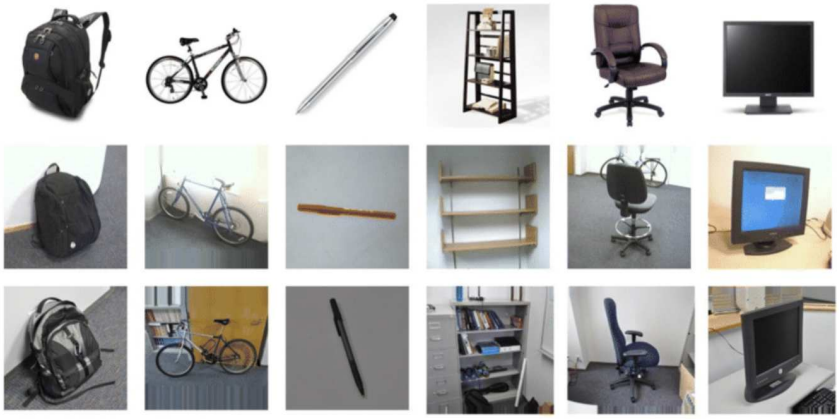


Fig. 2.3 Samples from the Office-31 [7] dataset. Each row contains images from a different domain: starting from the first row, examples from Amazon, Webcam and DSLR domains are shown.

In this thesis, we considered Office-31 for the Partial and Open-Set scenarios. In particular, in PDA the target contains only 10 categories which are those shared by Office-31 and Caltech-256 [212] as in [123]. For OSDA we followed the protocol in [131], where the first 10 classes in alphabetic order are considered known and the last 11 classes are considered unknown. Lastly, in MOSDA, we followed [138] where the first 20 classes in alphabetic order are considered as known, while the remaining 11 are unknown.

**VLCS** [213] is a benchmark made by the aggregation of images of 5 object categories shared by the PASCAL VOC 2007 [8], LabelMe [9], Caltech-101 [10] and SUN09 [11] datasets which are considered as 4 separated domains (see Figure

2.4). Originally proposed to study the problem of dataset bias, it has been widely used over the years to demonstrate the robustness of DG and DA approaches. In this thesis, we employed VLCS for DG following the standard protocol proposed in [110] that divides each domain into a training set (70%) and a test set (30%) by random selection from the whole dataset.

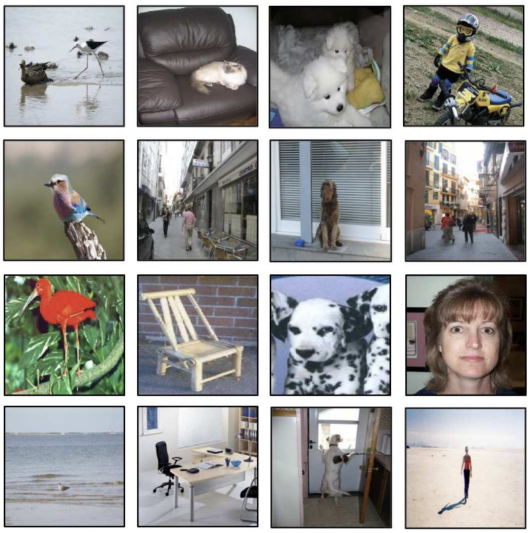


Fig. 2.4 Samples from the VLCS [7] benchmark. Each row contains images from a different domain: starting from the first row, examples from PASCAL VOC 2007 [8], LabelMe [9], Caltech-101 [10] and SUN09 [11] are shown.

**Office-Home** [4] has been proposed to overcome the limitations of previous benchmarks: both the number of categories and the large domain gaps (see Figure 2.5) make this dataset much more challenging than previous ones. It contains 65 categories of everyday objects from 4 domains: Art (Ar), Clipart (Cl), Product (Pr), and Real-World (Rw). Specifically, the Product domain is made by images collected from vendor websites and characterized by a white background, while Real-World images are captured with a regular camera. In this thesis, Office-Home has been considered for DG and Open-Set settings. In DG we followed the standard protocol proposed in [108].

In Open-Set: for Single-Source, we considered the first 25 classes in alphabetic order as known, and the remaining 40 classes as unknown following [24]; for Multi-Source we considered the first 45 categories in alphabetic order as known, and the remaining 20 unknown as in [138]. Finally, we adopted the same protocol of [140] for the OSDG experiments.

Fig. 2.5 Sample images from the Office-Home [4] dataset. The dataset consists of images of everyday objects organized into 4 domains. Each row contains images from a different domain: starting from the first row, examples from Art, Clipart, Product, and Real-World domains are shown.

**PACS** dataset has been proposed in [12] and it has been largely adopted to study the DG problem given the extremely different domains involved as shown in Figure 2.6. It contains 4 domains (Photo (P), Art Painting (A), Cartoon (C), and Sketch (S)) and it covers 7 object categories. For the DG setting, we followed the experimental protocol proposed in [12], and for the Multi-Source DA setting, we followed [87] where also the unlabeled target domain is used during training. Lastly, for OSDG we followed [140] using 6 categories as known and 1 as unknown with also a partial overlap between the sources' label sets.

**VisDA2017** was originally created for the 2017 Visual Domain Adaptation challenge (classification track) [13]. It has two domains, synthetic 2D object renderings (Syn.) and real images (Real) (see Figure 2.7) with a total of 208k images organized into 12 categories. With respect to the other datasets, it allows the investigation of the proposed solutions on a very large-scale sample size scenario. We considered this benchmark only for the PDA problem by following [123] that focuses on the synthetic-to-real shift keeping only the first 6 categories of the target in alphabetic order.

**DomainNet** [1], with about 0.6 million images, currently represents the largest DA benchmark. As can be seen in Figure 2.8 the domain gap is pretty large: it is

Fig. 2.6 Samples from the PACS [12] dataset. Each column contains images from a different domain: starting from the left, examples from Art Painting, Cartoon, Photo, and Sketch domains are shown.

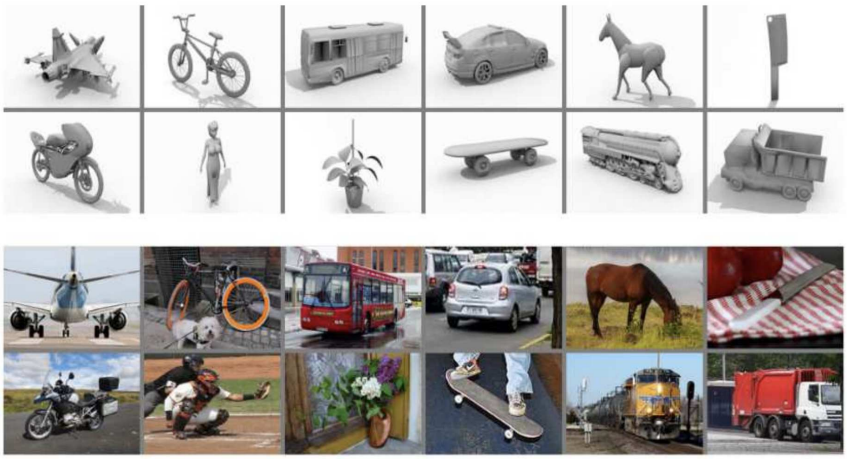


Fig. 2.7 Samples from the VisDA2017 [13] dataset which is made by two domains: Synthetic (on the top) and Real images (on the bottom).

made by six domains (Infograph (I), Painting (P), Sketch (S), Real (R), Quickdraw (Q), Clipart (C)) and 345 categories. We used it in this thesis for the MOSDA and SeND tasks. In the first case by following [138], we focused on Infograph, Painting, Sketch, and Clipart selecting randomly 50 samples per class or using all the images in case of lower cardinality. We used as known the first 100 classes in alphabetic order, while the remaining 245 are unknowns. In the SeND task, we selected 50 categories that do not overlap with ImageNet1k [214] classes using the

Natural Language Toolkit [215], we then randomly split them into 25 known and 25 unknown.


Fig. 2.8 Samples from the DomainNet [1] dataset. Each row contains images from a different domain: starting from the first row, examples from Clipart, Infograph, Painting, Quickdraw, Real, and Sketch domains are shown.

**Multi-Datasets** has been proposed in [140] with the aim of considering a realistic situation where the multi-source condition is naturally determined by the use of several datasets as source domains: Office-31 [7], STL-10 [216], Visda2017 [13]. One among Clipart, Real, Painting, and Sketch from DomainNet [1] serves as target domain. Following [140], we considered this dataset for the OSDG setting with 20 unknown selected categories.

**SUN RGB-D** [2] is the largest multi-modal benchmark for scene recognition across domains. Every scene is represented by both RGB and Depth data and, the properties of the device used for recording (Microsoft Kinect v2, Asus Xtion, Microsoft Kinect v1, and Intel RealSense) define the domain. Section 3.2 presents our experimental testbed based on this dataset, created to study the problem of domain shift cross-devices. As shown in Figure 2.9, there is a significant variation in the appearance of the images captured by different cameras. This means that users who want to test existing scene recognition models need to be careful in selecting a model that has been trained on images captured by the same type of deployment camera to avoid poor performance. In order to study this domain shift more in detail, we selected a subset of scene classes that are shared among the four SUN RGB-D cameras and that contain the largest number of samples per class (to increase

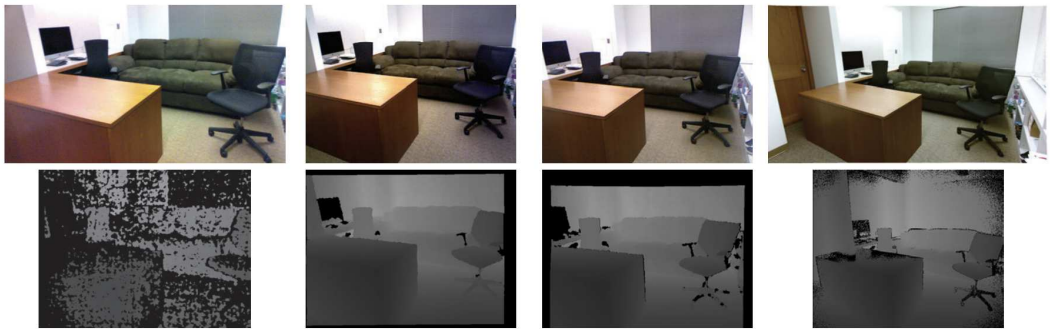


Fig. 2.9 Samples from the SUN RGB-D [2] dataset. Each column contains images from a different domain: starting from the left, images recorded with Intel RealSense, Asus Xtion, Microsoft Kinect v1, and Microsoft Kinect v2 are shown.

the number of samples, we combined the *office_kitchen* and *kitchen* categories). The final subset is summarized in Table 2.1. Overall, there are 10 classes, with *dining_area* and *bedroom* missing for, respectively, Kinect v1 and Realsense.

Table 2.1 Number of images in the considered classes.

| Class name | Kinect v1 | Kinect v2 | Realsense | Xtion |
|---|---|---|---|---|
| 0. bathroom | 147 | 150 | 67 | 260 |
| 1. bedroom | 442 | 121 | 0 | 521 |
| 2. classroom | 49 | 535 | 73 | 366 |
| 3. computer_room | 6 | 65 | 40 | 68 |
| 4. conference_room | 5 | 69 | 53 | 163 |
| 5. dining_area | 0 | 192 | 125 | 80 |
| 6. discussion_area | 6 | 62 | 30 | 103 |
| 7. kitchen | 291 | 86 | 20 | 183 |
| 8. office | 295 | 418 | 46 | 287 |
| 9. rest_space | 6 | 407 | 285 | 226 |
| Total | 1247 | 2105 | 739 | 2257 |

Some of the scene classes are made up of images taken in the same physical location but recorded with multiple cameras. We show this overlap in Figure 2.10. For example, the same group of *office* rooms has been recorded with Kinect v2, Realsense, and Xtion cameras, and all these images belong to the *office* class. Similarly, the *kitchen* class contains images taken in the same locations with Kinect v2 and Realsense, while others are from rooms shared between Xtion and Kinect v2. In the case of the *discussion_area* class, each camera recorded images in different physical locations. Lastly, none of the images captured with Kinect v1 share any location with the other cameras.

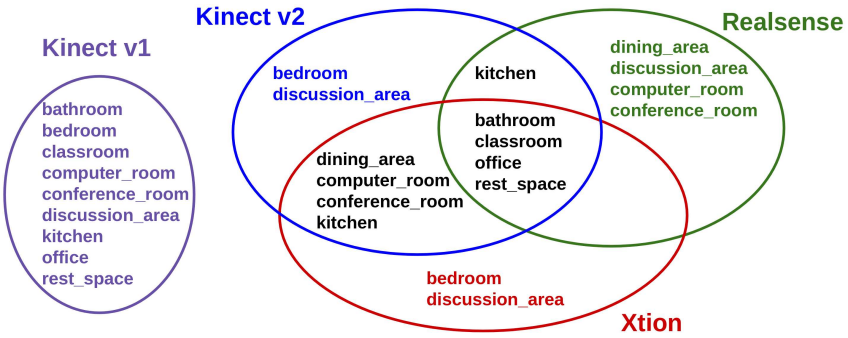



Fig. 2.10 Physical place overlapping. Some of the scene categories contain images taken in the exact same physical place with different devices, while others are recorded from different locations. We adopted the following color legend: black for a class that contains images recorded in the exact same location by multiple cameras, while blue/green/red/violet indicate classes with specific room images captured only with Kinect v2/Realsense/Xtion/Kinect v1.

We decided to focus on the Kinect v2 (K) and Xtion (X) cameras to define a 10-class domain adaptation problem, using them, in turn, as both source and target. Additionally, due to its limited number of samples, we only considered the Realsense (R) images as the target and used K, X, and their combination KX as the source. Furthermore, due to its class imbalance, we did not include Kinect v1 (Kv1) in our current classification setting but we plan to use it as a testbed for modality hallucination (see Section 3.2.2). In all the cases, we employed HHA [16] to encode depth images which have been shown to help in capturing the geometrical properties of depth data. The qualitative t-SNE [14] in Figure 2.11 reveals that the samples from each camera belong to different distributions and tend to occupy different regions of the space, with this being more pronounced for the depth modality, where the samples from Realsense are well separated from the other domains.

**Textures** [15] is made by $5,640$ photo of textures captured from the web (see Figure 2.12). They are labeled with one or more adjectives selected from 47 attributes that capture a wide variety of visual properties of textures, e.g. *banded*, *cobwebbed*, *freckled*, *knitted*, *zigzagged*. This dataset has been originally developed to support real-world applications where the recognition of texture properties is a key component. We considered it for the SeND Intra-Domain experiments by randomly choosing 23 categories as known and 24 as unknown.

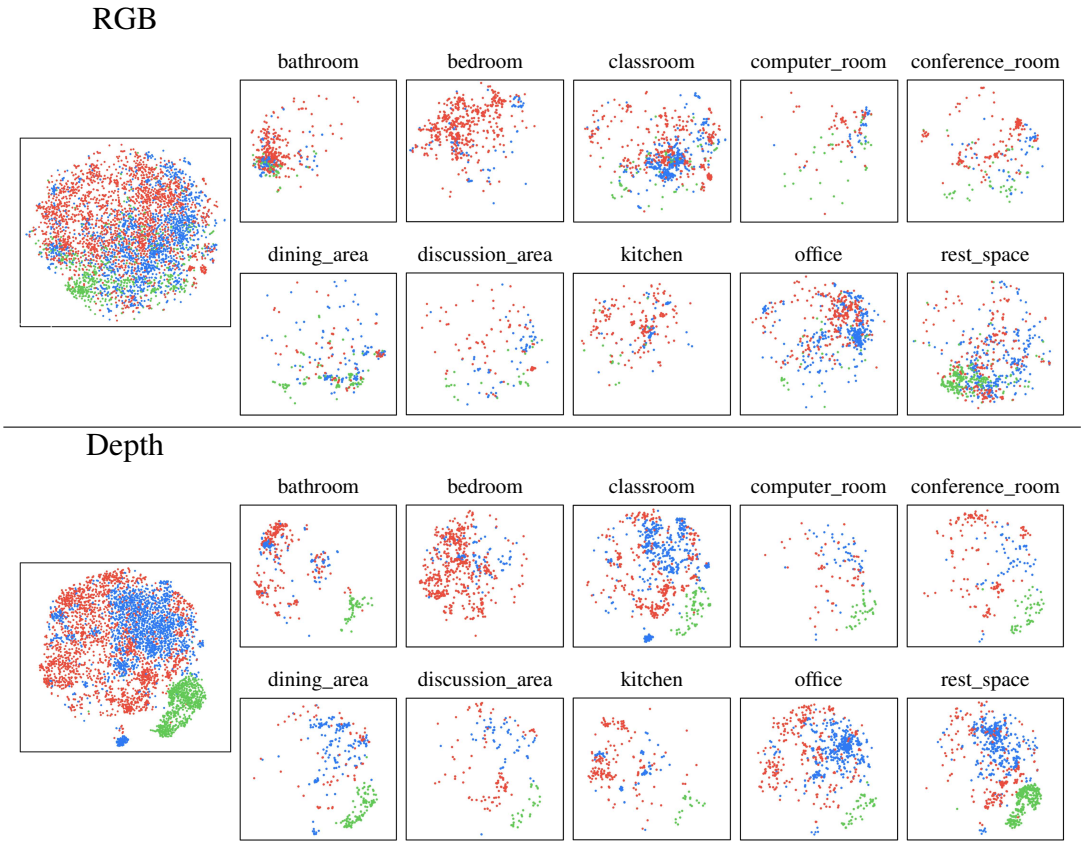



Fig. 2.11 Visualization through t-SNE [14] of the three domains of our multi-modal cross-domain scene classification testbed. Each domain is composed by images of a different camera: Kinect v2 (blue), Realsense (green), Xtion (red).

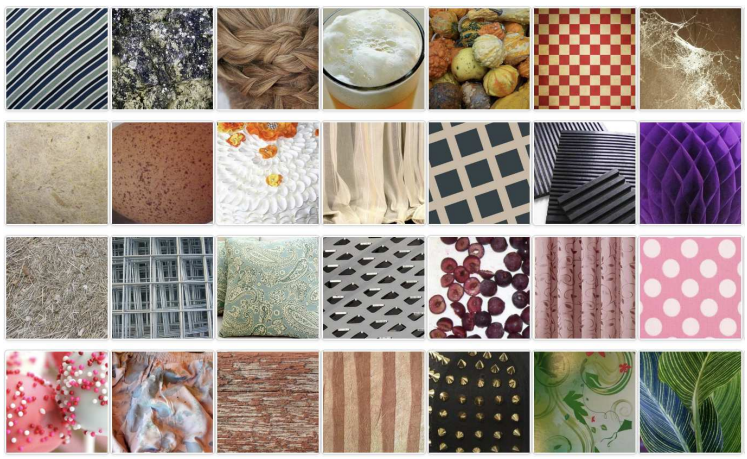


Fig. 2.12 Image samples from Textures [15] dataset.

# Chapter 3

# Auxiliary Self-Supervision for Closed-Set Cross-Domain Learning

*This Chapter introduces the first contribution of this thesis which consists in using Self-Supervised Learning as an auxiliary task for robust classification models across domains. In particular, it presents strategies to deal with Domain Generalization and Unsupervised Domain Adaptation problems under the Closed-Set assumption where the Source/s and Target label sets perfectly overlap. Specifically, two methods are introduced: JiGen and Tran-Adapt. In the first one, Jigsaw puzzle and Rotation recognition are used as auxiliary objectives to improve the network's generalization ability in object recognition. With the same aim, the second approach relies on an inter-modal Self-Supervised task whose goal is reconstructing RGB data from Depth and vice versa to improve scene recognition across domains.*

## 3.1 JiGen: Self-Supervised Learning Across Domains



In this Section we present the first multi-task approach that is simultaneously trained to recognize objects with a supervised objective and to generalize to new domains using self-supervision. We consider in our analysis jigsaw puzzle and rotation recognition as self-supervised tasks (see Figure 3.1) and we compare their effect when exploited as pretext tasks and in combination with supervised learning, exploring their potential through extensive ablation analysis and visualizations of both successful and failure cases. For the jigsaw puzzle task, we propose to reassembly the patches at the image level treating it as a classification problem (see Figure 3.2) as opposed to previous approaches that extracted features from individual image patches [5, 29]. By using the same network backbone for both object recognition and patch reordering/orientation prediction, it is possible to leverage the strengths of any convolutional learning structure and several pre-trained models without the need for architectural changes. We considered domain generalization, single-source, and multi-source domain adaptation settings comparing our performance with the contemporary state-of-the-art approaches.

The section is organized as follows. The proposed multi-task approach is described in subsection 3.1.1. The experimental results and analysis are provided in subsections 3.1.2, and we conclude in subsection 3.1.4.

### 3.1.1 Method

Let us assume that data from one or more source distributions are observed $\{(\mathbf{x}_i^s, \mathbf{y}_i^s)\}_{i=1}^{n^s}$ where $\mathbf{x}_i^s$ refers to the $i$-th image while $\mathbf{y}_i^s$ is the one-hot label vector of dimension $|Y^s|$. Starting from these images it is possible to apply various transformations to generate self-supervised variants. One simple option is to apply rotation to each sample producing four copies with orientations of 0, 90, 180, and 270 degrees considering as self-supervised task the prediction of the rotation angle to get back the original image. A more structured alternative is to decompose the original images into a

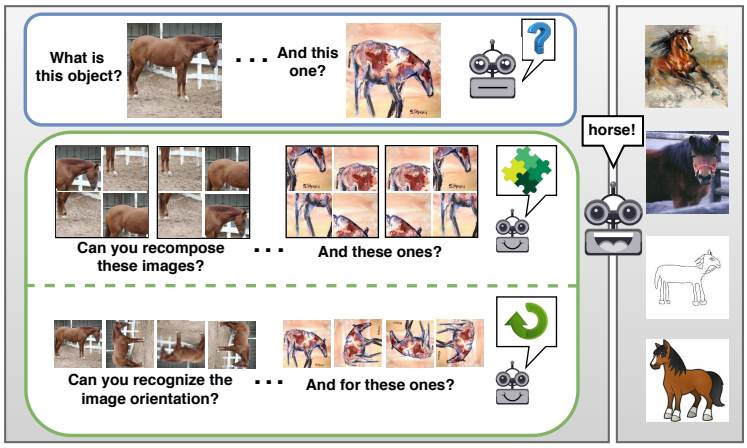



Fig. 3.1 Recognizing objects across different visual domains is a difficult task that requires strong generalization abilities. Self-supervised learning can help to capture natural invariances and regularities, which can then assist in bridging large style gaps. Our multi-task approach learns to jointly classify objects and solve jigsaw puzzles or recognize image orientation, demonstrating its effectiveness in knowledge generalization.

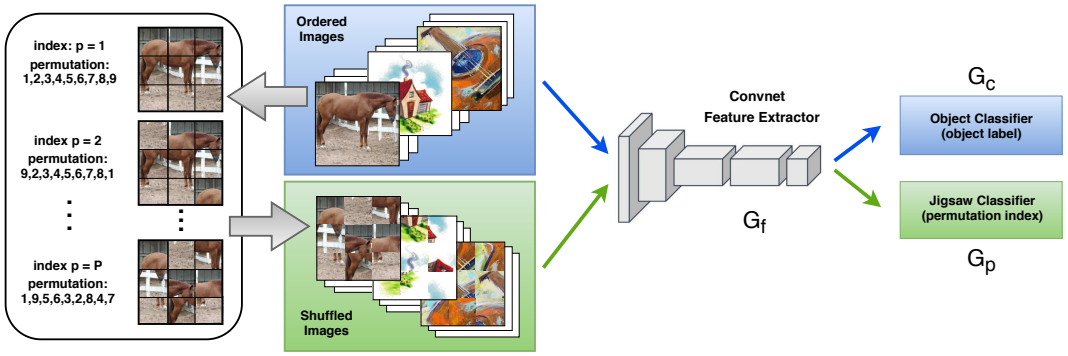



Fig. 3.2 The illustration of the proposed multi-task approach when using the jigsaw puzzle as self-supervised task. It starts from images from multiple domains and breaks them down into a $3 \times 3$ grid of patches which are then randomly shuffled and recomposed back into images of the same dimensions as the original ones. By using the maximal Hamming distance algorithm in [5], we establish a set of $P$ patch permutations and assign an index to each of them. Both the original and the shuffled images are fed into a convolutional network that is trained to meet two objectives: object classification on the ordered images and jigsaw classification (i.e. permutation index recognition) on the shuffled images. An analogous approach is used when using rotation recognition as self-supervised task. Note that the notation used to name the different network parts refers to Section 3.1.1.

$3 \times 3$ grid, creating 9 squared patches from each sample. These patches are then rearranged to create a set of 9! shuffled images. This task is reminiscent of the jigsaw puzzle game, in which the goal is to rearrange the tiles to reconstruct the original

image. In both of the cases described, $(\mathbf{z}_k^s, \mathbf{p}_k^s)_{k=1}^{K^s}$ indicates the newly obtained images where $\mathbf{z}_k^s$ is the transformed sample (rotated or shuffled). The dimension of the one-hot label vector $\mathbf{p}$ is 4 when applying rotation, while for patch shuffling, a subset $P$ of the 9! possible permutations is selected using the Hamming distance-based algorithm [5]. Depending on the self-supervised task, the total number of images changes: $K^s = 4 \times n^s$ for rotation and $K^s = P \times n^s$ for patch shuffling. Regardless of the self-supervised task considered, a multi-task model with a multi-branch ending network is used. A shared backbone is used to combine the chosen self-supervised objective with supervised learning [217]. One final branch will be used to elaborate the source data and solve the supervised task. A second branch is dedicated to the self-supervised task: permutation or rotation recognition (see Figure 3.2). The self-supervised auxiliary objective helps to extract meaningful, semantically important features from the data, which ultimately improves the performance of the object recognition task. The self-supervised objective does not require labels, so it can be used on both supervised and unsupervised data, supporting generalization and adaptation.

**Domain Generalization.** We denote the convolutional feature extractor of our network as $G_f$, which is parametrized by $\theta_f$ while the object classifier $G_c$ and the classifier for the self-supervised task $G_p$ are parametrized, respectively, with $\theta_c$ and $\theta_p$. The overall objective function is:

$$\begin{aligned} \arg\min_{\theta_f,\theta_c,\theta_p} \; & \frac{1}{n^s}\sum_{i=1}^{n^s} L_c(G_c(G_f(\mathbf{x}_i^s)),\mathbf{y}_i^s) + \\ & \alpha^s \frac{1}{K^s}\sum_{k=1}^{K^s} L_p(G_p(G_f(\mathbf{z}_k^s)),\mathbf{p}_k^s) \end{aligned} \tag{3.1}$$

where $L_c$ and $L_p$ indicate the cross entropy losses, respectively, for the object and self-supervised classifiers. Note that the self-supervised loss can be computed not just on transformed versions of the images, but also on the original images: the $0^\circ$ orientation, indeed, as well as the original patch ordering corresponds also to one of the possible transformations of the self-supervised task. Conversely, shuffled or rotated images will influence negatively the supervised classification objective making the object recognition task tougher. At test time, only the object classifier $G_c$ is used to predict the object categories of the target samples.

**Domain Adaptation.** Self-supervised learning does not require manual labeling and can take advantage of the available unlabeled target data $\{\mathbf{x}_j^t\}_{j=1}^{n^t}$ in the domain adaptation setting. The target samples are transformed (rotated or shuffled) to create new instances $\{\mathbf{z}_k^t\}_{k=1}^{K^t}$ associated with self-supervised labels $\mathbf{p}_k^t$. Another common strategy used to include target data into the learning process consists in applying the supervised knowledge learned from the source data to generate pseudo-labels $\hat{\mathbf{y}}^t = G_c(G_f(\mathbf{x}^t))$ for the target data, and minimize the prediction uncertainty, which is measured by the entropy $H = -\sum_{l=1}^{|Y^s|} \hat{\mathbf{y}}_l^t \log \hat{\mathbf{y}}_l^t$ [87, 53]. This semi-supervised method guides the class decision boundary to go through low-density target areas. However, its effectiveness when working across domains depends on the source-target domain gap: it needs to be moderate to prevent the generation of incorrect pseudo-labels. Since the entropy term, the supervised loss, and the self-supervised loss may be orthogonal and potentially complementary, we combine them in our domain adaptation analysis. The overall learning objective is formalized as:

$$\begin{aligned} \arg\min_{\theta_f,\theta_c,\theta_p} \; & \frac{1}{n^s}\sum_{i=1}^{n^s} L_c(G_c(G_f(\mathbf{x}_i^s)),\mathbf{y}_i^s) + \\ & \alpha^s \frac{1}{K^s}\sum_{k=1}^{K^s} L_p(G_p(G_f(\mathbf{z}_k^s)),\mathbf{p}_k^s) + \\ \eta \, \frac{1}{n^t}\sum_{j=1}^{n^t} H(G_c(G_f(\mathbf{x}_j^t))) + & \alpha^t \frac{1}{K^t}\sum_{k=1}^{K^t} L_p(G_p(G_f(\mathbf{z}_k^t)),\mathbf{p}_k^t). \end{aligned} \tag{3.2}$$

**Implementation details.** We designed[1] our multi-task network with a convolutional deep architecture $G_f$ that can be any standard network like AlexNet [218] or ResNet [219] while the object and self-supervised classifiers $G_c$ and $G_p$ are made by two different fully connected layers. When both Jigsaw and Rotation tasks are included in the model, for each task a $G_p$ head is assigned: the Jigsaw final head $G_p^J$ is used for the shuffled images, the Rotation recognition head $G_p^R$ for the rotated images. The network is trained end-to-end for all the experiments, starting from the pre-trained model on ImageNet [214] for $G_f$, while $G_c$ and $G_p$ are learned from scratch. In the DG setting, the main hyper-parameters of the proposed multi-task approach are $\alpha$ that weights the self-supervised loss and $\beta$ that controls the input data. Since the original images and the self-supervised variants are fed into the network together, $\beta$ is the hyper-parameter that controls their relative ratio in the image batch.

[1] Code available at: https://github.com/fmcarlucci/JigenDG

For example, setting $\beta = 0.6$ causes an image batch made by 60% of original images and 40% or transformed images for the self-supervised task. In our experiments, we tune $\alpha$ and $\beta$ by using a portion (10%) of our training data as validation and using it to perform the model selection by following [220]. When using both Jigsaw and Rotation methods, we have distinct parameters $\alpha_J$ and $\alpha_R$, and $\beta$ controls the proportion of images that are either rotated or shuffled, with equal probability. In the DA setting, the parameter $\alpha$ is decoupled into two separate parameters, $\alpha^s$ and $\alpha^t$, which are used, respectively, for the source and target data. When presenting our experimental results, we examine the impact of using cross-validation to determine the value of $\alpha$ on the source data and then setting $\alpha = \alpha^s = \alpha^t$, as well as the effect of fixing $\alpha^s = 0$ and manually adjusting $\alpha^t$. Another parameter in DA is $\eta$ which weighs the contribution of the entropy loss that we safely fixed to a small value, 0.1. When designing the jigsaw puzzle task, the grid of image patches ($n \times n$), and the number of permuted patches ($P$) have to be set. We show that our multi-task approach is not sensitive to these values and for all the experiments we maintained a consistent choice ($3 \times 3$ grid, $P = 30$). We employed a basic data augmentation technique consisting of randomly cropping the images to retain between $80 - 100\%$ of the original size and randomly flipping them horizontally. Additionally, following [29], we randomly converted 10% of the images to grayscale. Our DG and DA models have been trained using the SGD optimization algorithm, over 30 training epochs, with a batch size of 128. We set the initial learning rate to 0.001 and reduced it to 0.0001 after reaching 80% of the total training epochs.

### 3.1.2 Experiments

In this subsection, we conduct a thorough evaluation of the effectiveness of self-supervised learning across different visual domains. We start by analyzing the rotation and jigsaw puzzle, separately, as pretexts tasks, then we focus on DG and lastly, we extend our analysis to the DA scenario.

**Self-Supervised Pretraining.** With this first analysis, we investigate the generalization ability acquired by a model by using, separately, rotation prediction and jigsaw puzzle as pretext tasks. We trained three models with different implementations of jigsaw puzzle and one model with rotation recognition, in both cases using ImageNet [214] as training dataset. For the jigsaw puzzle, we employed two different Context-Free-Network (CFN) models, as described in [5, 29]. The CFN architecture

Table 3.1 We evaluated the performance of our model on different tasks and architectures using domain generalization (DG) classification accuracy. The column titles indicate the target domain. The best results are highlighted in bold. The top part of the table shows the results of self-supervised pretraining on Imagenet, followed by fine-tuning on the source. Methods that use patch-based networks are indicated by (p) while those that use whole-image networks are indicated by (w). The bottom part of the table shows the results of supervised pretraining on ImageNet followed by the multi-task combination of self-supervised objectives and supervised fine-tuning. The numbers reported correspond to the accuracy (%) averaged over three runs.

| **PACS** | **art_paint.** | **cartoon** | **sketches** | **photo** | **Avg.** |
|---|---|---|---|---|---|
| Self-Supervised Pretraining | | | | | |
| J-CFN (p) | 47.23 | 62.18 | 58.03 | 70.18 | 59.41 |
| J-CFN+ (p) | 51.14 | 58.83 | 54.85 | 73.44 | 59.57 |
| J-AlexNet (w) | 38.93 | 53.75 | 49.00 | 64.23 | 51.48 |
| R-AlexNet (w) | 52.08 | 59.24 | 56.54 | 72.91 | **60.19** |
| Supervised Pretraining and Multitask | | | | | |
| C-CFN-DeepAll (p) | 59.69 | 59.88 | 45.66 | 85.42 | 62.66 |
| C-CFN-Jigsaw (p) | 60.68 | 60.55 | 55.66 | 82.68 | 64.89 |
| AlexNet-DeepAll (w) | 66.50 | 69.65 | 61.42 | 89.68 | 71.81 |
| AlexNet-Jigsaw (w) | 67.79 | 70.79 | 64.01 | 89.64 | 73.05 |
| AlexNet-Rotation (w) | 69.43 | 69.40 | 65.20 | 89.17 | **73.30** |

is composed of 9 AlexNet-based siamese branches that extract features independently from each image patch, which are then recombined before being fed into the final classification layer. We refer to these models as J-CFN [5] and J-CFN+ [29] respectively. In addition to these two models, we trained a third jigsaw puzzle-based model, J-AlexNet, which is an AlexNet model trained on whole images recomposed from disordered patches. For the rotation recognition task (R-AlexNet), we followed the approach in [6] training an AlexNet model to solve the task. The results of these experiments are summarized in the top part of Table 3.1, and demonstrate that using a patch-based (p) jigsaw method generally produces a more dependable pretext model compared to using the whole (w) recomposed image. Furthermore, the results indicate that using rotation recognition as pretext task is the best choice for generalization purposes. To sum up, we observed that moving the jigsaw puzzle task from feature level to image level when training a pretext model is not the optimal solution and that the rotation task is the simplest and most effective approach among the evaluated pretext methods.

**Supervised Pretraining and Multi-task Learning.** In designing our multi-task approach, we have multiple options available in terms of architecture and in selecting the most appropriate auxiliary task. We evaluate the performance of the multi-branch Context-Free-Network (CFN) architecture against a plain AlexNet backbone. To distinguish the CFN model used for classification from the self-supervised pretraining described in the previous paragraph, we refer to it as C-CFN. Regardless of the specific architecture used, with *DeepAll*, we refer to the single-task supervised model trained without self-supervision (i.e. $\alpha = 0$), while we use *Jigsaw* or *Rotation* to indicate the multi-task cases where self-supervised tasks were trained jointly with object classification. From the results in the bottom part of Table 3.1 we can draw three key conclusions. First, combining supervised and self-supervised learning leads to better performance regardless of the architecture used. Second, a single-branch architecture is better suited for the multi-task problem at hand. Lastly, the whole-image Rotation auxiliary task supports generalization slightly better than the Jigsaw task.

**Domain Generalization.** Here we provide an extensive evaluation of our multi-task approach against state-of-the-art multi-source DG methods. We evaluate different types of DG methods based on: low-rank constraints on the network parameters (TF [12], SLRC [221]), domain-specific component aggregation (Epi-FCR [120], D-SAM [108]), meta-learning strategies (MLDG [121], MetaReg [119] and adversarial techniques (DDAIG [222], PAR [223], MMLD [224]). We provide detailed results for each method, including the DeepAll reference, to observe the relative advantage of each approach[2].

Table 3.2 shows the results of our multi-task approach on the PACS dataset considering Jigsaw puzzle, Rotation recognition, and their combination. On average, our approach produces equal or better results than those of the competitors, with the only exception of DDAIG which achieved the best results on Resnet-18. It's worth noting that DDAIG, unlike our multi-task method, relies on domain annotation for each source sample, which may not always be available in practical conditions [87]. Furthermore, DDAIG benefits from tailored per-domain model parameter selection, as opposed to our approach where the parameters are fixed and shared by all the

[2]The variations between the DeepAll results are probably due to small, undocumented inconsistencies and/or variations in library implementations of these baseline methods. Reporting all of them is the only fair way to demonstrate the relative improvement provided by each approach and to highlight any potential inconsistencies.

Table 3.2 We compare our approach with state-of-the-art DG methods on the PACS dataset. The column titles indicate the target domain. The table shows the hyperparameters used for each experiment, obtained through source cross-validation. The best results are highlighted in bold. The numbers reported correspond to the accuracy (%) averaged over three runs.

| | PACS | art_paint. | cartoon | sketches | photo | Avg. |
|---|---|---|---|---|---|---|
| | **Alexnet** | | | | | |
| [12] | DeepAll | 63.30 | 63.13 | 54.07 | 87.70 | 67.05 |
| | TF | 62.86 | 66.97 | 57.51 | 89.50 | 69.21 |
| [108] | DeepAll | 64.44 | 72.07 | 58.07 | 87.50 | 70.52 |
| | D-SAM | 63.87 | 70.70 | 64.66 | 85.55 | 71.20 |
| [120] | DeepAll | 63.40 | 66.10 | 56.60 | 88.50 | 68.70 |
| | Epi-FCR | 64.70 | 72.30 | 65.00 | 86.10 | 72.00 |
| [121] | DeepAll | 64.91 | 64.28 | 53.08 | 86.67 | 67.24 |
| | MLDG | 66.23 | 66.88 | 58.96 | 88.00 | 70.01 |
| [119] | DeepAll | 67.21 | 66.12 | 55.32 | 88.47 | 69.28 |
| | MetaReg | 69.82 | 70.35 | 59.26 | 91.07 | 72.62 |
| [223] | DeepAll | 63.30 | 63.10 | 54.00 | 87.70 | 67.03 |
| | PAR | 68.70 | 70.50 | 64.60 | 90.40 | 73.54 |
| [224] | DeepAll | 68.09 | 70.23 | 61.80 | 88.86 | 72.25 |
| | MMLD | 66.99 | 70.64 | 67.78 | 89.35 | 73.69 |
| | DeepAll | 66.50 | 69.65 | 61.42 | 89.68 | 71.81±0.26 |
| | Jigsaw$_{\alpha=0.9,\beta=0.6}$ | 67.76 | 70.79 | 64.01 | 89.64 | 73.05±0.20 |
| | Rotation$_{\alpha=0.4,\beta=0.4}$ | 69.43 | 69.40 | 65.20 | 89.17 | 73.30±0.47 |
| | Jigsaw+Rotation$_{\alpha_J=0.9,\alpha_R=0.9,\beta=0.4}$ | 69.70 | 71.00 | 66.00 | 89.60 | **74.08**±0.32 |
| | **Resnet-18** | | | | | |
| [108] | DeepAll | 77.87 | 75.89 | 69.27 | 95.19 | 79.55 |
| | D-SAM | 77.33 | 72.43 | 77.83 | 95.30 | 80.72 |
| [120] | DeepAll | 77.60 | 73.90 | 70.30 | 94.40 | 79.10 |
| | Epi-FCR | 82.10 | 77.00 | 73.00 | 93.90 | 81.50 |
| [119] | DeepAll | 79.90 | 75.10 | 69.50 | 95.20 | 79.90 |
| | MetaReg | 83.70 | 77.20 | 70.30 | 95.50 | 81.70 |
| [222] | DeepAll | 77.00 | 75.90 | 69.20 | 96.00 | 79.50 |
| | DDAIG | 84.20 | 78.10 | 74.70 | 95.30 | **83.10** |
| [224] | DeepAll | 78.34 | 75.02 | 65.24 | 96.21 | 78.70 |
| | MMLD | 81.28 | 77.16 | 72.29 | 96.09 | 81.83 |
| | DeepAll | 77.83 | 74.26 | 65.81 | 95.71 | 78.40±0.28 |
| | Jigsaw$_{\alpha=0.7,\beta=0.9}$ | 79.28 | 75.74 | 68.31 | 95.71 | 79.80±0.55 |
| | Rotation$_{\alpha=0.8,\beta=0.4}$ | 81.07 | 74.13 | 76.17 | 96.10 | 81.87±0.49 |
| | Jigsaw+Rotation$_{\alpha_J=0.7,\alpha_R=0.7,\beta=0.8}$ | 81.07 | 73.97 | 74.67 | 95.93 | 81.41±0.50 |

Table 3.3 We compare our approach to state-of-the-art DG methods on the VLCS dataset. For more information on the notation used, please refer to Table 3.2. The numbers reported correspond to the accuracy (%) averaged over three runs.

| | **VLCS** | **Caltech** | **Labelme** | **Pascal** | **Sun** | **Avg.** |
|---|---|---|---|---|---|---|
| | | **Alexnet** | | | | |
| [12] | DeepAll | 93.40 | 62.11 | 68.41 | 64.16 | 72.02 |
| | TF | 93.63 | 63.49 | 69.99 | 61.32 | 72.11 |
| [221] | DeepAll | 86.67 | 58.20 | 59.10 | 57.86 | 65.46 |
| | SLRC | 92.76 | 62.34 | 65.25 | 63.54 | 70.97 |
| [108] | DeepAll | 94.95 | 57.45 | 66.06 | 65.87 | 71.08 |
| | D-SAM | 91.75 | 56.95 | 58.59 | 60.84 | 67.03 |
| [120] | DeepAll | 93.10 | 60.60 | 65.40 | 65.80 | 71.20 |
| | Epi-FCR | 94.10 | 64.30 | 67.10 | 65.90 | 72.90 |
| [224] | DeepAll | 95.89 | 57.88 | 72.01 | 67.76 | 73.39 |
| | MMLD | 96.66 | 58.77 | 71.96 | 68.13 | **73.88** |
| | DeepAll | 96.15 | 59.05 | 70.84 | 63.92 | 72.49±0.21 |
| | Jigsaw$_{\alpha=0.5,\beta=0.8}$ | 96.46 | 59.51 | 72.95 | 64.40 | 73.33±0.16 |
| | Rotation$_{\alpha=0.9,\beta=0.6}$ | 97.30 | 60.30 | 71.93 | 65.97 | **73.88**±0.62 |
| | Jigsaw+Rotation$_{\alpha_J=0.9,\alpha_R=0.5,\beta=0.7}$ | 96.30 | 59.20 | 70.73 | 66.37 | 73.15±0.36 |

domain pairs in each dataset. Similar observations hold for the VLCS results in Table 3.3 and for Office-Home in Table 3.4. In the Office-Home dataset, the Rotation task performed better than Jigsaw, improving more than three percentage points over the DeepAll baseline with an even larger advantage for the Jigsaw+Rotation case. In the same table, it can be observed that DDAIG produced the highest average results but the improvement over its DeepAll reference is slightly over one percentage point.

**Ablation and hyper-parameter tuning.** In our multi-task approach, there are two key parameters, $\alpha$, and $\beta$, that play different roles in regulating the training process. Specifically, $\alpha$ controls the importance of the self-supervised auxiliary loss,

Table 3.4 We compare our approach to state-of-the-art DG methods on the Office-Home dataset. For more information on the notation used, please refer to Table 3.2. The numbers reported correspond to the accuracy (%) averaged over three runs.

| | **Office-Home** | **Art** | **Clipart** | **Product** | **Real-World** | **Avg.** |
|---|---|---|---|---|---|---|
| | | **Resnet-18** | | | | |
| [108] | DeepAll | 55.59 | 42.42 | 70.34 | 70.86 | 59.81 |
| | D-SAM | 58.03 | 44.37 | 69.22 | 71.45 | 60.77 |
| [222] | DeepAll | 58.90 | 49.40 | 74.30 | 76.20 | 64.70 |
| | DDAIG | 59.20 | 52.30 | 74.60 | 76.00 | **65.50** |
| | DeepAll | 52.15 | 45.86 | 70.86 | 73.15 | 60.51±0.12 |
| | Jigsaw$_{\alpha=0.9,\beta=0.8}$ | 53.04 | 47.51 | 71.47 | 72.79 | 61.20±0.11 |
| | Rotation$_{\alpha=0.8,\beta=0.4}$ | 57.80 | 48.73 | 72.70 | 74.87 | 63.53±0.25 |
| | Jigsaw+Rotation$_{\alpha_J=0.4,\alpha_R=0.5,\beta=0.9}$ | 58.33 | 49.67 | 72.97 | 75.27 | 64.06±0.31 |

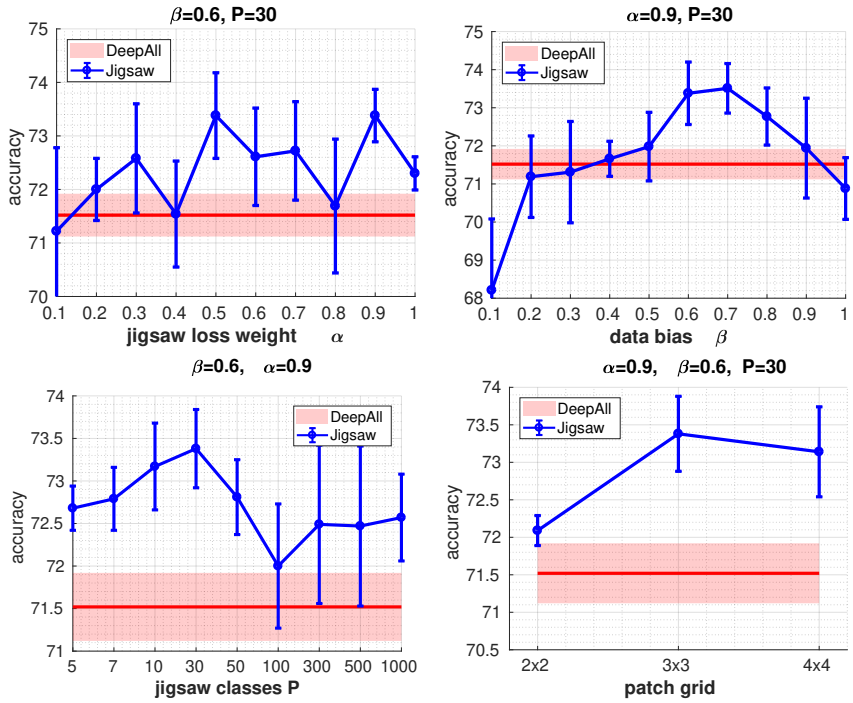



Fig. 3.3 We perform an ablation study and analyze the effects of different hyperparameters on our approach when using Jigsaw on the Alexnet-PACS domain generalization setting. The reported accuracy is the global average over all target domains, and each run was repeated three times. The red line in the figures represents the average accuracy of our DeepAll model from Table 3.2.

while $\beta$ regulates the proportion of samples that are directed to the self-supervised branch. To understand the impact of the two key parameters we conduct an ablation study by considering a wide range of values. For these experiments, we focused on Alexnet using PACS in DG setting with Jigsaw puzzle as self-supervised task. We changed one parameter at a time while keeping the remaining ones fixed: $3 \times 3$ the grid size and $P = 30$ permutations. Note that by selecting $\alpha = 0, \beta = 1$, we are deactivating the self-supervised task and the data batches are made up only of ordered images (DeepAll). The data bias value $\beta$ directs the training process, it shifts the emphasis from the self-supervised task with low values ($\beta < 0.5$) to primarily object classification when using higher values ($\beta \geq 0.5$). We specifically set it to 0.6, which indicates that we fed more ordered than shuffled images to the network, keeping the primary focus on object classification. With this setting, changing the loss weight $\alpha$ in $0.1, 1$, we consistently observe results that are statistically equal or better than the DeepAll baseline as indicated in the first plot on the top left of Figure 3.3. The

plot on the top right in Figure 3.3 reveals that when $\alpha$ is high, adjusting $\beta$ has a significant impact on the overall performance. When $\alpha \sim 1, \beta = 1$, this means that Jigsaw is active and highly influences the learning process, but only ordered images are used for training. This can make the puzzle task too easy and lead the network to always recognize the same permutation class, which can result in overfitting rather than regularizing the learning process. Setting $\beta = 0$ denotes feeding the network with only shuffled images. Each image generates $P$ variations, with only one variant having the patches in the correct order and being able to enter the object classifier. This results in a substantial decrease in the real batch size. Under this condition, the object classifier is unable to converge, regardless of whether Jigsaw is active or not. In such scenarios, the accuracy is quite low ($< 20\%$), and as a result, it is not shown in the plots to improve the readability.

Additionally, we test the robustness of our method by varying the number of Jigsaw classes (patch permutations) $P$, as well as the dimensions of the patch grid $n \times n$. The first plot in the bottom part of Figure 3.3 shows the change in performance when the number of Jigsaw classes $P$ varies between 5 and 1000. As can be seen, the overall variation in accuracy is about 1.5 percentage points and it still generally remains higher than the DeepAll baseline. As a final step, we conducted an evaluation to understand the effect of changing the grid size (i.e. the number of patches). Even in this case, the change in performance is minimal when comparing results obtained with a $2 \times 2$ grid to a $4 \times 4$ grid, confirming the conclusions of robustness already obtained for this parameter in [5, 28].

Compared to patch decomposition and puzzle reordering, rotating an image has a relatively minor impact on its overall appearance. Still, it has a number of self-supervised classes significantly lower than the jigsaw puzzle ($P = 4$ vs $P \sim 10 - 50$). With Rotation, even using a low value for $\beta = 0.4$ does not divert the focus of the network from the primary task of object classification, and, when combined with $\alpha = 0.4$, produces the results reported in Table 3.2. For the ablation analysis in Figure 3.4, we kept one parameter fixed while changing the other one. The results are consistently above the DeepAll baseline, with a limited change in performance indicating a low sensitivity to the specific parameter settings.

We have seen how the self-supervised tasks support the main supervised classifier for domain generalization, but it is also interesting to check their own internal functioning and whether those tasks get meaningful results. The first plot in Figure

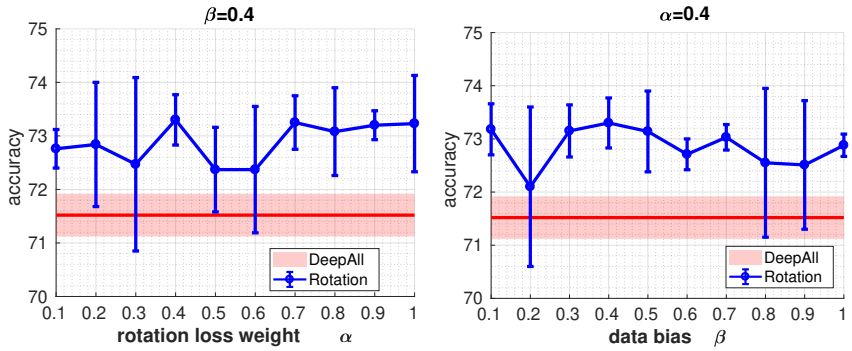



Fig. 3.4 Ablation analysis on the Alexnet-PACS DG setting when using Rotation. The reported accuracy is the global average over all the target domains, and each run was repeated three times. The red line in the figures represents the average accuracy of our DeepAll model from Table 3.2.

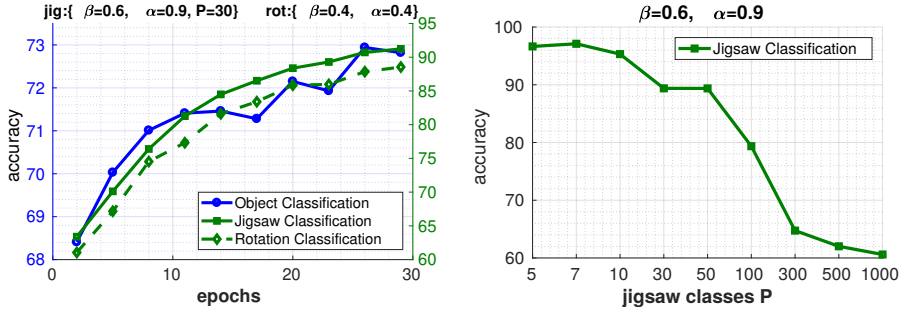



Fig. 3.5 Analysis of the Jigsaw classifier on Alexnet-PACS DG setting. In the plot on the left, each axis refers to the color matching curve in the graph.

3.5 shows the accuracy during training on the target domain for the Object, Rotation, and Jigsaw classifiers, showing that they all increase simultaneously but at different rates. The second plot shows the Jigsaw task accuracy when varying the number of permutation classes $P$. It can be observed that performance decreases as the task become more challenging, but, overall, the results indicate that the Jigsaw model is always effective in reordering the shuffled patches.

**Visual Explanation and Failure Cases.** As noted in [165], models trained through supervised learning tend to rely heavily on local image statistics, resulting in a limited generalization and robustness of the learned representations. The jigsaw puzzle and the rotation recognition task, by forcing the network to use the whole image, allow to capture global information and to identify domain-agnostic object shapes. By integrating both supervised and self-supervised objectives, we aim at learning a representation that more effectively captures discriminative cues, leading

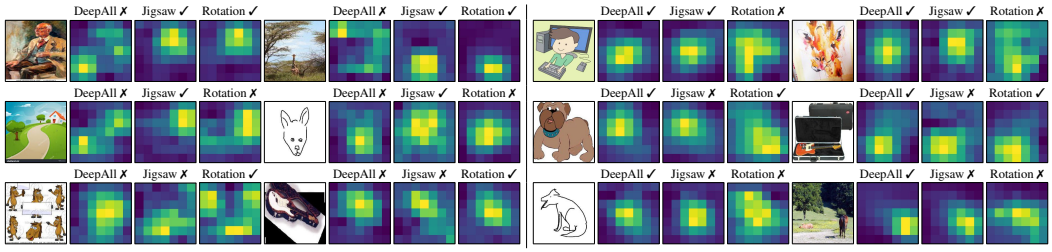



Fig. 3.6 In CAM activation maps, yellow represents high values and dark blue represents low values. The jigsaw puzzle task is effective in identifying the most informative parts of an image for object class prediction across different visual domains. Similarly, rotation recognition can also be useful, but it tends to be less accurate in terms of localization, particularly for sketches, cartoons, and paintings.

to improved object recognition across domains. To confirm this observation, in Figure 3.6 we show the Class Activation Maps (CAM) of a ResNet-18 trained for DG on the PACS dataset. The first two rows show that the models produced with our multi-task approach using Jigsaw or Rotation as auxiliary tasks are better at identifying the object class in comparison to the DeepAll model. The Rotation task appears to be slightly less precise in capturing object shapes, particularly when dealing with sketches (as seen in the dog on the second and sixth rows), cartoons, and paintings (fourth and fifth rows), but still performs reasonably well with photos. The last two rows show that for both Jigsaw and Rotation the recognition errors are related to some flaw in the data interpretation, while the localization remains meaningful.

**Domain Adaptation.** If unsupervised target data is available during training, we can use both source and target to solve the self-supervised tasks, making the model more robust to domain shift. To analyze the performance of our approach, we evaluated it against a number of different DA approaches. These methods can be broadly grouped into four categories: 1) those that aim to reduce domain shift by measuring and minimizing the Maximum Mean Discrepancy [225], such as DAN [44] and JAN [45], 2) adversarial-based approaches, like DANN [56], 3) those that use batch normalization to match source and target distributions as Dial [226] and DDiscovery [87], and 4) those that focus on increasing feature norms, such as HAFN [53] and its step-wise variant SAFN. We run our experiments using the Office-Home and PACS datasets for the single-source and multi-source settings respectively. Several DA approaches minimize the entropy loss as an extra domain alignment condition (e.g. SAFN+ENT). For a fair comparison, we also turned on

Table 3.5 Accuracy on *Office-Home* under single-source DA setting. The top result is highlighted in bold. The numbers reported correspond to the accuracy (%) averaged over three runs.

| Office-Home-DA | Ar→Cl | Ar→Pr | Ar→Rw | Cl→Ar | Cl→Pr | Cl→Rw | Pr→Ar | Pr→Cl | Pr→Rw | Rw→Ar | Rw→Cl | Rw→Pr | Avg. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Resnet-50** | | | | | | | | | | | | | |
| ResNet-50 | 34.90 | 50.00 | 58.00 | 37.40 | 41.90 | 46.20 | 38.50 | 31.20 | 60.40 | 53.90 | 41.20 | 59.90 | 46.10 |
| DAN [44] | 43.60 | 57.00 | 67.90 | 45.80 | 56.50 | 60.40 | 44.00 | 43.60 | 67.70 | 63.10 | 51.50 | 74.30 | 56.30 |
| DANN [56] | 45.60 | 59.30 | 70.10 | 47.00 | 58.50 | 60.90 | 46.10 | 43.70 | 68.50 | 63.20 | 51.80 | 76.80 | 57.60 |
| JAN [45] | 45.90 | 61.20 | 68.90 | 50.40 | 59.70 | 61.00 | 45.80 | 43.40 | 70.30 | 63.90 | 52.40 | 76.80 | 58.30 |
| ResNet-50 | 49.36 | 68.86 | 76.25 | 58.71 | 66.18 | 69.33 | 56.59 | 44.80 | 75.80 | 67.66 | 51.21 | 79.52 | 63.69 |
| HAFN [53] | 50.20 | 70.10 | 76.60 | 61.10 | 68.00 | 70.70 | 59.50 | 48.40 | 77.30 | 69.40 | 53.00 | 80.20 | 65.40 |
| SAFN [53] | 52.00 | 71.70 | 76.30 | 64.20 | 69.90 | 71.90 | 63.70 | 51.40 | 77.10 | 70.90 | 57.10 | 81.50 | 67.30 |
| SAFN+ENT [53] | 52.26 | 73.04 | 77.06 | 66.12 | 72.30 | 72.27 | 64.96 | 52.67 | 78.81 | 72.96 | 58.05 | 82.12 | **68.55** |
| ResNet-50 | 48.30 | 59.80 | 68.40 | 54.70 | 62.40 | 65.10 | 53.70 | 46.70 | 73.70 | 66.80 | 54.10 | 77.30 | 60.91±0.15 |
| Jigsaw$_{\alpha^s=\alpha^t=0.7,\beta=0.8}$ | 47.70 | 58.80 | 67.90 | 57.20 | 64.30 | 66.10 | 56.20 | 50.80 | 75.10 | 67.90 | 55.60 | 78.40 | 62.17±0.10 |
| Jigsaw$_{\alpha^s=0,\alpha^t=0.7,\beta=0.8}$ | 47.33 | 58.07 | 67.70 | 57.77 | 63.47 | 65.70 | 56.43 | 50.13 | 74.70 | 68.40 | 55.77 | 79.23 | 62.06±0.23 |
| Rotation$_{\alpha^s=\alpha^t=0.8,\beta=0.6}$ | 49.00 | 59.20 | 67.40 | 56.90 | 64.10 | 65.60 | 56.60 | 52.90 | 74.70 | 68.70 | 57.90 | 78.60 | 62.64±0.13 |
| Rotation$_{\alpha^s=0,\alpha^t=0.8,\beta=0.6}$ | 48.83 | 56.67 | 67.50 | 57.47 | 63.90 | 65.47 | 56.33 | 52.23 | 74.33 | 68.97 | 57.53 | 78.20 | 62.29±0.14 |

the entropy loss for our approach. Additionally, we solve the self-supervised task either involving both source and target or considering only the latter balancing the self-supervised losses through cross-validation. The results presented in Table 3.5 demonstrate the performance of our single source approach on the Office-Home dataset. Our multi-task approach improves over its baseline and over DAN, JAN, and DANN but has worse performance than HAFN, SAFN, and SAFN+ENT. In order to more thoroughly assess the relative gain of HAFN/SAFN methods, we show their baseline results (ResNet-50) which are not typically shown. It can be observed that the inclusion of an additional fully connected layer in the basic architecture of HAFN [53] is particularly beneficial in cross-domain settings.

We also performed an ablation analysis by disabling the self-supervised task on the source by setting $\alpha_s = 0$: the slight variation in the results suggests that most of the adaptation effect is derived from the self-supervised task on the target. The multi-source experiments presented in Table 3.6 provide additional insight into the adaptability of the auxiliary self-supervised objective in our multi-task approach. When the source domain contains information on a wide range of styles, our method surpasses not only Dial and DDiscovery, but also the more advanced DA techniques HAFN and SAFN. After evaluating both Jigsaw and Rotation, it seems that Rotation is better suited for domain adaptation as it produces higher and more stable performances. The bottom part of Table 3.6 also reveals the impact of varying the $\alpha$ value which appears more relevant for Jigsaw than for Rotation. On average, when Jigsaw and Rotation are combined, it results in a slight improvement with respect to DeepAll. We also include the DG results for the Jigsaw+Rotation model, which is the case where $\alpha^t = 0$ and $\eta = 0$, while all other chosen parameters

are kept the same. We also examined the separate effects of turning off only the self-supervised tasks on the target ($\alpha^t = 0$) or the entropy loss ($\eta = 0$), from this analysis it can be observed that the major adaptation effect is derived from the self-supervised tasks running on the target rather than from the entropy loss.

Table 3.6 Multi-source Domain Adaptation results on PACS. The numbers reported correspond to the accuracy (%) averaged over three runs.

| | PACS-DA | art_paint. | cartoon | sketches | photo | Avg. |
|---|---|---|---|---|---|---|
| | **Resnet-18** | | | | | |
| | DeepAll | 74.70 | 72.40 | 60.10 | 92.90 | 75.03 |
| [87] | Dial | 87.30 | 85.50 | 66.80 | 97.00 | 84.15 |
| | DDiscovery | 87.70 | 86.90 | 69.60 | 97.00 | 85.30 |
| | DeepAll | 76.17 | 73.58 | 55.65 | 96.07 | 75.37±0.42 |
| [53] | HAFN | 84.95 | 79.64 | 64.24 | 97.70 | 81.63±0.50 |
| | SAFN | 86.78 | 82.72 | 60.26 | 98.26 | 82.01±0.32 |
| | SAFN+ENT | 89.22 | 87.39 | 60.02 | 98.14 | 83.69±0.17 |
| | DeepAll | 77.83 | 74.26 | 65.81 | 95.71 | 78.40±0.28 |
| | Jigsaw$_{\alpha^s=\alpha^t=0.7,\beta=0.8}$ | 84.49 | 82.07 | 79.86 | 97.98 | 86.10±0.26 |
| | Rotation$_{\alpha^s=\alpha^t=0.8,\beta=0.4}$ | 89.97 | 82.60 | 82.00 | 98.07 | 88.16±0.51 |
| | Jigsaw+Rotation $_{\substack{\alpha^s_j=\alpha^t_j=0.2,\\ \alpha^s_R=\alpha^t_R=0.8,\beta=0.8}}$ | 89.67 | 82.87 | 83.93 | 98.17 | **88.66**±0.36 |
| | Jigsaw$_{\alpha^s=0.7,\alpha^t=0.1,\beta=0.8}$ | 85.40 | 81.49 | 76.93 | 98.35 | 85.54±1.63 |
| | Jigsaw$_{\alpha^s=0.7,\alpha^t=0.3,\beta=0.8}$ | 85.92 | 81.61 | 79.74 | 98.04 | 86.33±0.58 |
| | Jigsaw$_{\alpha^s=0.7,\alpha^t=0.5,\beta=0.8}$ | 87.01 | 81.25 | 78.87 | 98.00 | 86.28±0.67 |
| | Jigsaw$_{\alpha^s=0.7,\alpha^t=0.9,\beta=0.8}$ | 84.21 | 80.38 | 76.64 | 97.86 | 84.77±0.76 |
| | Rotation$_{\alpha^s=0.8,\alpha^t=0.1,\beta=0.4}$ | 89.27 | 81.30 | 82.23 | 89.73 | 87.71±0.13 |
| | Rotation$_{\alpha^s=0.8,\alpha^t=0.3,\beta=0.4}$ | 88.73 | 82.20 | 81.47 | 98.27 | 87.67±0.07 |
| | Rotation$_{\alpha^s=0.8,\alpha^t=0.5,\beta=0.4}$ | 89.83 | 80.10 | 81.13 | 98.00 | 87.27±0.94 |
| | Rotation$_{\alpha^s=0.8,\alpha^t=0.9,\beta=0.4}$ | 89.17 | 81.47 | 82.73 | 97.87 | 87.81±0.21 |
| | Jigsaw+Rotation $_{\alpha^t=0,\eta=0}$ | 81.07 | 73.97 | 74.67 | 95.93 | 81.41±0.50 |
| | Jigsaw+Rotation $_{\alpha^t=0}$ | 82.80 | 77.23 | 77.70 | 97.17 | 83.73±0.39 |
| | Jigsaw+Rotation $_{\eta=0}$ | 84.67 | 78.63 | 80.37 | 97.27 | 85.23±0.51 |

### 3.1.3 Computational Cost

Before concluding the section we find it relevant to discuss the computational cost of the proposed multi-task supervised and self-supervised based solution. Indeed often models that produce good results come with a significant computational effort which makes their use practically unaffordable. In Table 3.7 we compare Jigsaw+Rotation with its best competitor MMLD [224] according to the results in Table 3.2. We show the total number of FLOPs (Floating Point Operations) required for a single forward pass of the network, the training, and the inference time (in milliseconds). As can be noticed the higher performance of our proposed approach is not correlated with the

size of the model, nor with the time required for the model to converge. Overall the cost analysis reveals that Jigsaw+Rotation is slightly more convenient than MMLD.

Table 3.7 Cost analysis on PACS with AlexNet in DG setting. Hardware - CPU: Intel(R) Core(TM) i7-9800X CPU @ 3.80GHz, GPU (x1): Nvidia TITAN Xp.

| | **Cost analysis** | | | |
|---|---|---|---|---|
| | FLOPs | Training Time (ms) | Inference Time (ms) | Accuracy (Avg.) |
| Jigsaw+Rotation | $7.16 \times 10^8$ | $1.13 \times 10^6$ | 2.28 | **74.08** $\pm$ 0.32 |
| MMLD [224] | $7.22 \times 10^8$ | $2.42 \times 10^6$ | 2.48 | 73.69 |

### 3.1.4 Conclusions

This Section presented an extensive study of the application of self-supervised learning across domains. Specifically, we focused on solving jigsaw puzzles and recognizing image orientation, demonstrating how they can be effectively incorporated into a multi-task approach that also includes supervised learning. The results indicate that this learning strategy improves cross-domain robustness and generalization performance, and is on par with more complex domain adaptation and domain generalization methods. Our work paves the way for many other adaptive methods exploiting the invariances captured by the most recent self-supervised solutions [227, 165] , not just in object classification, but also in other difficult tasks such as semantic segmentation [228], detection [229] or 3D visual learning [230] where the domain shift effect has a significant impact on the effectiveness of methods in real-world scenarios.

## 3.2 TranAdapt: Multi-Modal RGB-D Scene Recognition Across Domains



*Scene recognition* consists in assigning a label as *kitchen*, *office*, *bakery*, or *beach* to an image, and it is a crucial vision problem for robot localization and decision making [231, 232]. To successfully recognize a scene, an agent should be able to identify objects, understand their relationships, and be resilient to large intra-scene variations and inter-scene overlaps. In this context, RGB images offer important information about the way objects look, while depth information is necessary to understand the 3D environment. However, obtaining a large dataset of RGB-D images can be challenging, particularly compared to gathering RGB scene images alone (*e.g.* by crawling the web). This has led to focus only on RGB data in scene recognition research, as demonstrated by the use of CNN models on Places dataset [233]. Only in recent years, with the increased availability of low-cost depth sensors, a larger number of RGB-D images has become accessible. Multi-modal scene recognition research has evolved over the years, moving from models based on handcrafted features [234, 235] to complex deep networks that can learn representations from large amounts of data [236–238]. One of the earliest solutions was to fuse the RGB and depth data at the input level, treating the depth as an additional channel of the image [239] or fusing the output scores [240]. The majority of the recent methods focus on combining the features in the middle layers of the network [2, 238, 241] proposing techniques to better capture the relationship between RGB and depth data [242–244, 36]. Still, the existing literature leaves behind some important analysis on the nature of the data used, considered as drawn from a single domain distribution. The term RGB-D includes a wide range of 3D cameras that can vary significantly in terms of depth sensing technology, image range, and field of view. This, combined with the fact that images labeled with the same class can be taken in different physical locations by heterogeneous cameras, leads to a significant domain shift among the data (see Fig 3.7). This variability raises questions about the robustness of the developed approaches and highlights the need to further investigate the nature of the data used. *Domain adaptation* addresses the problem

of learning models on some source labeled data distribution that generalizes to a different unlabeled target distribution [20]. Most of the existing techniques have focused on single-modal data, meaning that they only consider RGB information or include the multi-modality only for one domain, such as RGB-D in the source domain and RGB in the target domain. Additionally, these methods typically focus on cross-domain object classification [53, 56, 59] or scene segmentation [245–247], but less attention has been given to the problem of scene recognition. In this section, we explore for the first time the intersection of three important research areas: *scene recognition*, *multi-modal learning*, and *domain adaptation*. The main contributions of this section can be summarized as follows:

- We propose a benchmark testbed[3] for unsupervised domain adaptation in the field of scene recognition. We use a subset of scene classes from the SUN RGB-D [2] dataset captured with four different 3D cameras. Each camera is considered as an RGB-D domain, resulting in an experimental framework with five multi-modal domain pairs (see Section 2.1 for more details).

- We conduct a comprehensive evaluation of various state-of-the-art techniques that were developed to address one or two of the three research areas we focus on. Specifically, we evaluate: (a) the performance of *multi-modal scene recognition* models in a cross-domain setting [36, 244]; (b) the effectiveness of single-modal *domain adaptation* approaches when applied to multi-modal scene recognition [53, 56, 59]; (c) the performance of a recent *multi-modal domain adaptation* approach that was originally developed for object classification in the context of scene recognition [18].

- We propose a method called *Translate-to-Adapt* that is inspired by a previous work on inter-modal translation [36]. Our method uses the task of generating depth images from their RGB counterparts, and vice-versa, as a self-supervised task that can be applied to labeled source data and unlabeled target data. We use both modality translation directions as part of an end-to-end classification model and obtain promising results across different domains.

[3]Dataset and code available at https://github.com/silvia1993/Multi-Modal_RGB-D_Scene_Recognition_Across_Domains

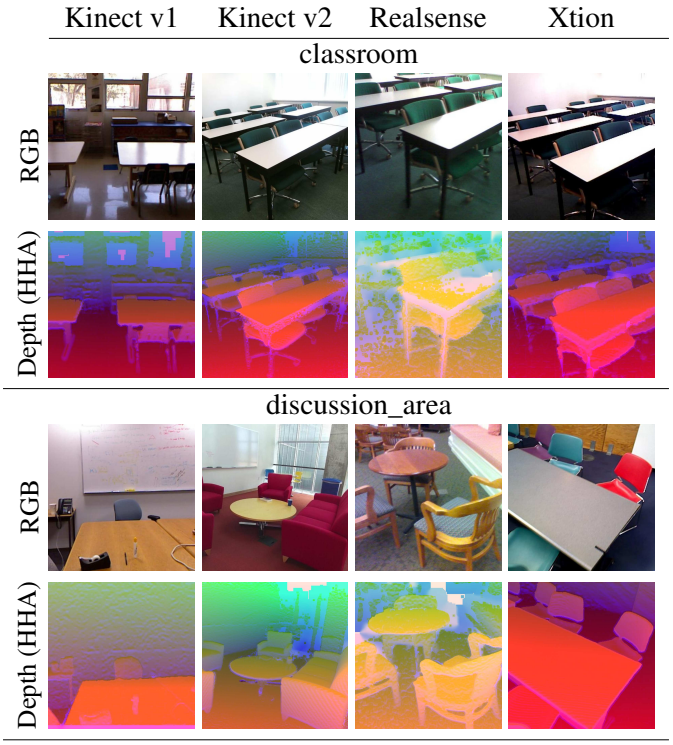



Fig. 3.7 Examples of RGB and Depth HHA [16] images from all the cameras within the SUN RGB-D dataset [2]. The category *classroom* contains images taken in the exact same place with Kinect v2, Realsense, and Xtion cameras, while the physical location captured with Kinect v1 is different although annotated with the same label. For the category *discussion_area* there is no room overlap: despite the shared label, each camera captured images in a different physical location. As can be noticed, the specific camera characteristics contribute to producing significant appearance differences. Best seen in color.

### 3.2.1 Method

**Intuition.** In Section 3.1 we showed that self-supervised learning can improve the ability of a model to generalize across different visual domains [34, 81]. One common self-supervised task for handling multi-modal source and target domains is the conversion of one modality into the other. In this particular case, this would involve predicting depth information from an RGB image and generating an RGB image from depth. By training a model to perform both tasks, it learns to identify the underlying relationship between the two modalities. By applying this process to both source and target domains, the model becomes able at identifying the invariant characteristics of the relationship between RGB and depth. This is expected to result in improved performance for cross-domain scene classification.

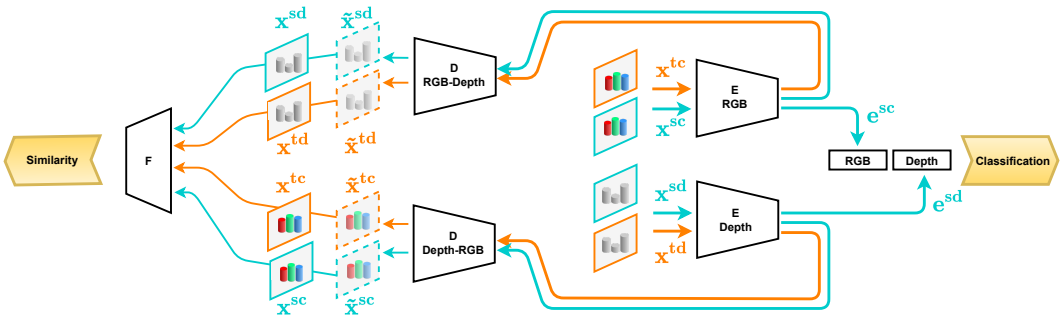



Fig. 3.8 Our Translate-to-Adapt method for RGB-D scene recognition across domains consists of several key components: encoders (E), inter-modality decoders (D), a semantic feature extractor (F), classification and similarity evaluation heads. The encoders process both the RGB and depth images separately and then combine their features for the classification task. The decoders are responsible for converting one modality into the other, and both have the same structure but focus on different translation directions. The generated images are compared to their original versions using the semantic feature extractor and the similarity head. It's worth noting that while only the supervised source data is used in the classification task, both source and target data are used in the inter-modality generation self-supervised task. We use the notation presented in Section 3.2.1.

**In more Technical Terms.** Our goal is to use source labeled and target unlabeled multi-modal images to predict the scene class of the target data. The source data is represented by $S = \{(\mathbf{x}_i^{sc}, \mathbf{x}_i^{sd}), \mathbf{y}_i^s\}_{i=1}^{N^s}$, which includes pairs of RGB and depth images, along with their one-hot encoded scene class labels. The target data, represented by $T = \{(\mathbf{x}_i^{tc}, \mathbf{x}_i^{td})\}_{i=1}^{N^t}$, consist of unlabeled images that are drawn from a different distribution but share the same class set as the source data. To exploit the relationship between the two modalities, we use an auxiliary objective for inter-modal translation, converting both RGB to depth $\mathbf{x}^{*c} \rightarrow \mathbf{x}^{*d}$ and depth to RGB $\mathbf{x}^{*d} \rightarrow \mathbf{x}^{*c}$, which can be applied to both source and target data effectively bridging the two domains and adapting the learned representation.

**Network architecture and Optimization.** The Translate-to-Adapt method consists of a six-part architecture, as shown in Figure 3.8. It includes two modality-specific encoders (E), two decoders (D), a feature extractor (F), and a final classifier. The source and target data are processed by the encoders, which map the original images into feature embeddings of equal dimensionality for the two modalities: $\mathbf{e}_i^{*c} = E_{rgb}(\mathbf{x}_i^{*c})$, $\mathbf{e}_i^{*d} = E_{depth}(\mathbf{x}_i^{*d})$. The concatenated features of the source data, $\{(\mathbf{e}_i^{sc}, \mathbf{e}_i^{sd})\}_{i=1}^{N^s}$, are used for the main classification task. The feature embeddings for both source and target data are then provided as input to their corresponding decoders, which translate them into the other modality: $\tilde{\mathbf{x}}_i^{*d} = D_{rgb-depth}(E_{rgb}(\mathbf{x}_i^{*c}))$

and $\tilde{\mathbf{x}}_i^{*c} = D_{depth-rgb}(E_{depth}(\mathbf{x}_i^{*d}))$. The generated images are paired with their original version and the difference among the features extracted by F is minimized for each case: $\{\tilde{\mathbf{x}}_i^{sc}, \mathbf{x}_i^{sc}\}_{i=1}^{N^s}$, $\{\tilde{\mathbf{x}}_i^{sd}, \mathbf{x}_i^{sd}\}_{i=1}^{N^s}$, $\{\tilde{\mathbf{x}}_i^{tc}, \mathbf{x}_i^{tc}\}_{i=1}^{N^t}$, $\{\tilde{\mathbf{x}}_i^{td}, \mathbf{x}_i^{td}\}_{i=1}^{N^t}$.

The final model jointly optimizes the classification and instance similarity objectives via a cross-entropy loss function $L_{cls}$ and the content similarity loss $L_{sim}$ among the generated-original sample pairs. The latter is an L1 loss

$$\sum_{l=1}^{L} ||F^l(\tilde{\mathbf{x}}_i^{*c}) - F^l(\mathbf{x}_i^{*c})||_1 + ||F^l(\tilde{\mathbf{x}}_i^{*d}) - F^l(\mathbf{x}_i^{*d})||_1 \tag{3.3}$$

measured over multiple internal layers of the F module (l= layer1-layer4 in ResNet-18). Finally, the total loss is

$$L_{cls} + \alpha^s L_{sim}^s + \alpha^t L_{sim}^t \ . \tag{3.4}$$

**Implementation Details.** The defined optimization problem guides the training of encoders and decoders, while for F we used a frozen model. All the components use a ResNet-18 structure that is pre-trained on ImageNet[214]. The loss hyperparameters $\alpha^s$ and $\alpha^t$ are set at 10 and 3 respectively (see the ablation analysis in Section 3.2.2). We designed the network modules by following [36], but the learning procedure differs. Besides including the target data, in our Translate-to-Adapt the multi-modal fusion strategy for classification is learned end-to-end with all the other network components, rather than with a two-step process. The model is trained using ADAM optimization with a batch size of 40 and a total of 70 epochs. The learning rate starts at $2 \times 10^{-4}$ and decreases linearly over the last 50 epochs. Depth images are processed offline to HHA [16] representation and along with the RGB images they are resized and randomly cropped. The central crop is used during testing.

### 3.2.2 Experiments

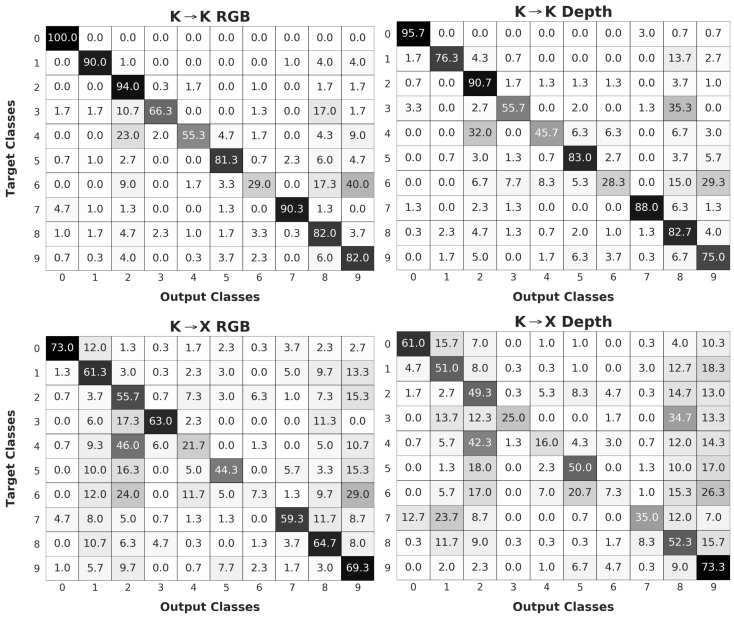
**Reference Methods.** To understand the challenges of learning a multi-modal cross-domain scene recognition model, we perform a benchmark analysis with existing approaches originally developed either for multi-modal scene recognition or single-modal cross-domain object classification. From the first category, we consider Translate-to-Recognize (Tran-Rec) [36], and Centroid Based Concept Learning

(CBCL) [244] which employs a linear combination of multi-modal sample distances to determine class assignments. Both those methods were developed to work on training and test data drawn from a single domain. We also consider as baseline the basic ResNet-18. In general, those methods are *Source Only*, meaning that during training the target test data is not available. We also consider a simple multi-modal strategy called "Fusion" in which networks for each modality are trained independently and then the feature extractors are frozen while the representations are concatenated and input into a fully connected layer for scene classification. Our approach *Fusion++* instead deals with both modalities simultaneously by training the feature representation and multi-modal classifier jointly. For the second category of methods, the unlabeled target data is used together with the labeled training data. These include: Gradient Reversal Layer (GRL) [56] which uses an adversarial domain classifier to reduce the feature distribution difference between source and target, Adversarial Feature Normalization (AFN) [53] that starts from the observation that target samples are often characterized by feature norm values much lower than those of the source data and propose to progressively increase them, and CycleGAN [59] which is an unsupervised generative approach that modifies the style of source data to resemble the target. This method is used to create target-like RGB and depth images from the annotated source samples and the models trained on them are combined with the *Fusion* strategy. To the best of our knowledge, there is only one previous work that has specifically focused on multi-modal cross-domain object classification, which we refer to as the Relative Rotation (Rel. Rot) method [18]. This approach leverages a self-supervised task to understand the correlation between RGB and depth images in order to create robust, domain-invariant features for the main recognition task. We compare our Translate-to-Adapt (Tran-Adapt) method against these existing approaches.

**Preliminary Analysis.** We conducted a quantitative experiment focusing on the K and X cameras, organizing their images into three 70%/30% train/test splits. We trained a simple ResNet-18 classification model and evaluated it both within each camera and across cameras: the average results over the splits are reported in the first and second row of Table 3.8 for each of the two modalities. The drop in performance (summarized in the last row) clearly demonstrates the existence of a significant domain shift. The confusion matrices of the K→X case show how the domain shift affects the per-class recognition accuracy.

Table 3.8 Accuracy (%) across domains for single modality. The performance drop shows the effect of the domain shift. The confusion matrices for the K→X case also reveal that the behavior across domains varies for the two modalities: classes 2 (classroom) and 7 (kitchen) are the ones most affected by the domain shift for RGB and depth modalities, respectively.

| | RGB | Depth | | RGB | Depth |
|---|---|---|---|---|---|
| K → K | 77.09 | 72.09 | X → X | 79.98 | 72.79 |
| K → X | 51.90 | 42.07 | X → K | 57.50 | 54.43 |
| drop | 25.19 | 30.02 | drop | 22.48 | 18.36 |

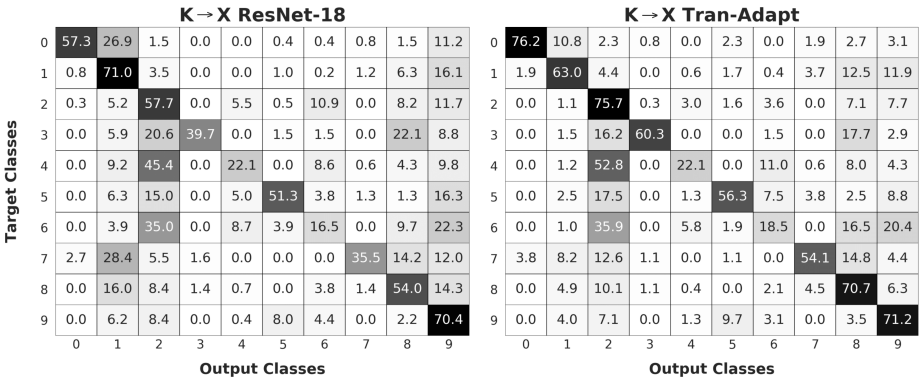



**Results and Ablation.** Table 3.9 shows the classification accuracy values obtained by the considered reference approaches and by our Tran-Adapt method. Specifically, the top part contains the Source Only baselines whose results indicate that combining the two modalities of the source data improves the recognition performance across domains. For completeness, we also developed the Fusion++ version of the Tran-Rec method, although the end-to-end training procedure was not included in the original paper [36]. The CBCL Fusion approach outperforms the others.

The central part of the table presents the results of the domain adaptive methods. Even in this case, the multi-modal versions improve over the corresponding single-modal ones. The style-transfer-based CycleGAN method demonstrates the most significant improvement. Rel. Rot., the only existing approach that utilizes the inter-modality relation for cross-domain learning, falls slightly behind the performance of

Table 3.9 Accuracy (%) of several methods for RGB-D domain adaptation. Top results in bold. The confusion matrices show the K→X per-class results for the ResNet-18 baseline and Tran-Adapt (Fusion++).

| Method | | K → X | X → K | K → R | X → R | KX → R | AVG |
|---|---|---|---|---|---|---|---|
| ResNet-18 | RGB | 47.56 | 57.55 | 38.34 | 44.88 | 41.82 | 46.03 |
| | Depth | 38.76 | 54.42 | 26.56 | 26.87 | 30.98 | 35.52 |
| | Fusion | 50.66 | 62.91 | 44.54 | 46.54 | 42.56 | 49.44 |
| | Fusion++ | 47.54 | 60.27 | 39.56 | 36.32 | 43.71 | 45.48 |
| Tran-Rec [36] | RGB-D | 52.54 | 61.68 | 38.63 | 46.24 | 44.59 | 48.74 |
| | D-RGB | 37.13 | 53.49 | 29.77 | 29.06 | 32.25 | 36.34 |
| | Fusion | 53.92 | 63.40 | 39.35 | 43.40 | 48.29 | 49.67 |
| | Fusion++ | 51.17 | 62.62 | 39.53 | 41.38 | 50.87 | 49.11 |
| CBCL [244] | Fusion | 55.35 | 60.57 | 50.51 | 42.45 | 49.94 | 51.76 |
| GRL [56] | RGB | 50.11 | 59.88 | 53.30 | 51.18 | 46.82 | 52.26 |
| | Depth | 45.25 | 54.29 | 37.30 | 32.41 | 37.80 | 41.41 |
| | Fusion | 48.28 | 64.73 | **53.53** | 51.91 | 47.51 | 53.19 |
| | Fusion++ | 50.94 | 61.91 | 53.45 | 48.90 | 48.85 | 52.81 |
| AFN [53] | RGB | 51.59 | 56.73 | 52.11 | 47.63 | 46.86 | 50.98 |
| | Depth | 40.22 | 51.88 | 34.20 | 32.33 | 35.20 | 38.77 |
| | Fusion | 51.29 | 61.88 | 47.84 | 50.25 | 50.07 | 52.27 |
| | Fusion++ | 56.74 | 57.89 | 52.13 | 49.05 | 45.66 | 52.30 |
| CycleGAN [59] | Fusion | 54.25 | 63.19 | 53.02 | 48.02 | 54.65 | 54.63 |
| Rel. Rot. [18] | Fusion++ | 50.98 | **65.99** | 48.33 | 52.24 | 53.53 | 54.21 |
| Tran-Adapt | RGB-D | 52.11 | 61.91 | 46.93 | 51.27 | 54.88 | 53.42 |
| | D-RGB | 48.09 | 55.69 | 38.95 | 38.78 | 40.79 | 44.46 |
| | Fusion | 55.61 | 65.23 | 41.90 | 43.59 | 48.03 | 50.87 |
| | Fusion++ | **56.79** | 64.41 | 48.13 | 51.02 | 55.31 | 55.13 |
| Tran-Adapt Aug | Fusion++ | 55.65 | 65.92 | 53.01 | **52.56** | **55.59** | **56.55** |




CycleGAN. The bottom part of the table shows the results of our Tran-Adapt with the Fusion++ version that outperforms all the considered competitors. Analysis of the confusion matrices in the K→X experiment reveal a clear performance enhancement

with Tran-Adapt, as seen by a reduction in misassignments between classes such as *kitchen* and *bedroom*, and between *classroom* and *computer room*. We can also take advantage of the generative nature of Tran-Adapt by exploiting the produced images as data augmentation. Inspired by [36], we used the images generated by the RGB-D and D-RGB models as additional input for the Fusion++ network. By including a random subset of these generated images, equivalent to 30% of the batch size, we were able to achieve an accuracy of 56.55% (*Tran-Adapt Aug*).

As specified in the previous section, for Tran-Adapt we set $\alpha^s = 10$ and $\alpha^t = 3$. We kept the first value fixed using the same value as Tran-Rec [36]. The second one weights the importance of the self-supervised task applied to the target data: to get discriminative features the main focus should remain on the annotated source, with $\alpha^t < \alpha^s$. By conducting a preliminary validation analysis on the separate RGB-D and D-RGB directions, we selected $\alpha^t = 3$, maintained also for the Fusion and Fusion++ versions. To evaluate the impact of the source and target self-supervised translation tasks, we performed an ablation analysis by disabling the target contribution and observing a decrease in performance in comparison to Tran-Rec ($\alpha^s = 10$, $\alpha^t = 0$, Fusion++ 49.11% ). Instead, by turning off the source contribution and keeping the target one ($\alpha^s = 0$, $\alpha^t = 3$, Fusion++ 54.22%), we observe an adaptation effect. This effect further improves when utilizing both source and target components with ($\alpha^s = 10$ and $\alpha^t = 3$, Tran-Adapt, Fusion++ 55.13%). Additionally, minimal variations occur in the average results of Fusion++ when keeping $\alpha^s = 10$ and changing $\alpha^t = 1,2,3,4$ obtaining 54.71, 54.44, 55.13, 54.81 (%)

**Self-supervision for Cross-Domain Scene Recognition.** Both Rel. Rot. and Tran-Adapt exploits self-supervised tasks (rotation recognition and RGB-depth image mapping) to learn inter-modality cues that support cross-domain adaptation. Still, considering the observed performance difference, we decided to investigate their behavior more in-depth. Specifically, we searched for possible shortcuts followed by the rotation auxiliary task that might have misled the scene recognition process. Indeed, Rel. Rot. was originally designed for object recognition on datasets where the objects are typically well-centered in the images and the background information is marginal. When dealing with scenes, the risk of focusing on low semantically meaningful cues to predict the image orientation increases, affecting also the final scene class assignment.

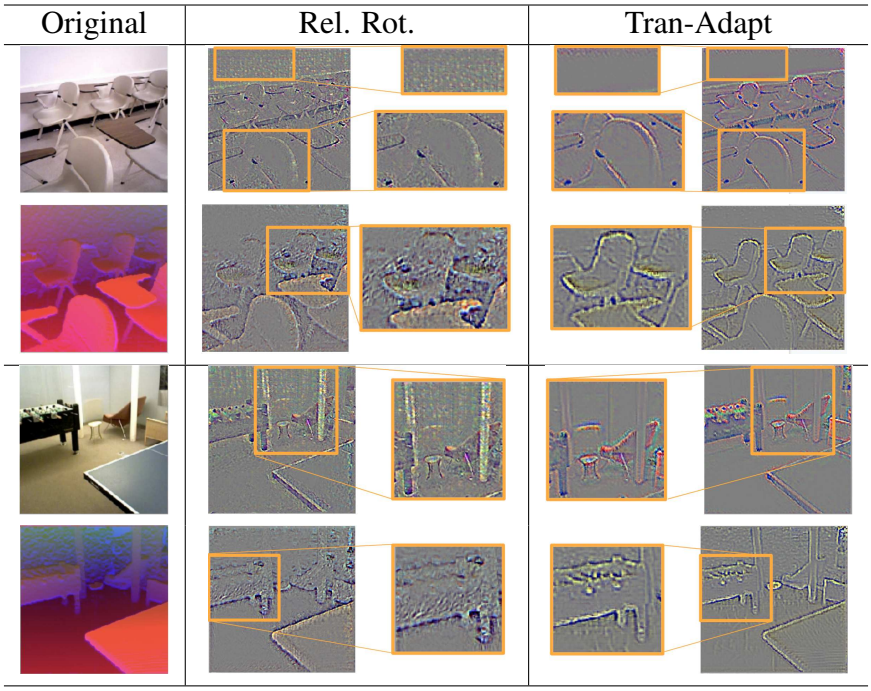



Fig. 3.9 Visualizations obtained by guided backpropagation [17] that show the most important pixels used by Rel. Rot. [18] and our Tran-Adapt.

In Figure 3.9, we present the outputs of the *guided backpropagation*[17] technique, which allows us to see the regions of the image that are most important in the model's prediction. By comparing the visualizations of Rel. Rot. and Tran-Adapt, we can observe that both methods pay attention to object boundaries, but Rel. Rot. tends to also include non-meaningful information from uniform regions and rely on neat lines (see the third image row and the columns in the image) in the background.

**Missing Modality prediction on Novel Target Scenes.** The primary goal of the proposed approach is scene recognition, however, it also includes a generative component that can be used for additional tasks. One potential use is producing images for a missing modality, in case of problems with the sensing devices. When the image generation involves scene categories never seen during training the task becomes particularly challenging. To evaluate this capability, we selected three classes not included in our original dataset from SUN RGB-D creating a new small dataset which considers all four available cameras (see Table 3.10).

We tested the performance of both the Tran-Rec and Tran-Adapt models on this new dataset, by measuring the pixel-to-pixel L2 difference between the generated and original images. The results in Table 3.11 show that Tran-Adapt performed

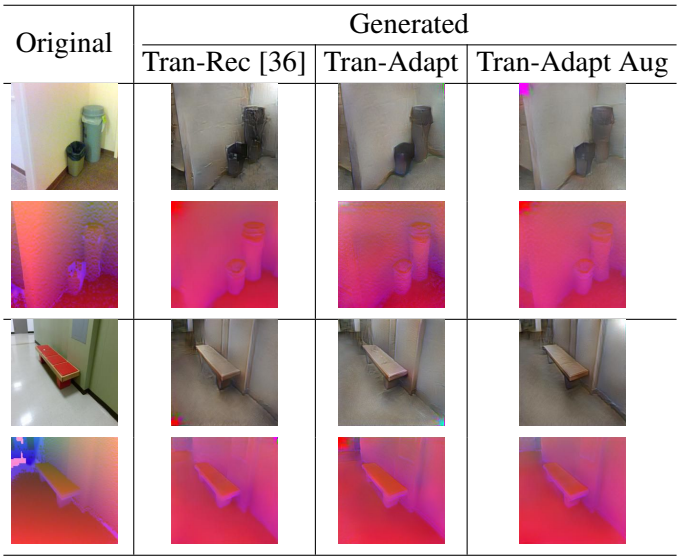



Fig. 3.10 Qualitative comparison of real and generated images on the unseen class *corridor*. It's particularly evident the effectiveness of Tran-Adapt and its Aug version on the RGB images considering the uniform regions like walls and floors that appear smoother than in Tran-Rec.

Table 3.10 Number of samples in extra classes considered for the missing modality prediction.

| Class name | Kinect v1 | Kinect v2 | Realsense | Xtion |
|---|---|---|---|---|
| corridor | 15 | 153 | 23 | 182 |
| printer_room | 4 | 43 | 9 | 21 |
| study_space | 7 | 121 | 26 | 38 |
| Total | 26 | 317 | 58 | 241 |

better than Tran-Rec, further demonstrating its ability to generalize. Some examples of the generated images can be seen in Figure 3.10.

### 3.2.3 Computational Cost

In Table 3.12 we show the computational cost of our multi-task approach and its best competitor CycleGAN [59] for the experiments in Table 3.9. We show the total number of FLOPs (Floating Point Operations) required for a single forward pass of the network, the training, and the inference time (in milliseconds). As can be noticed, the number of FLOPs and the time required for training is much lower for our proposed solution. Indeed, CycleGAN is a two-stage approach: first, an

Table 3.11 Pixel-to-pixel L2 distance between real and generated images from unseen classes of the target domain. Top results in bold (the lower the better).

| | Tran-Rec [36] | | Tran-Adapt | | Tran-Adapt Aug | |
|---|---|---|---|---|---|---|
| | RGB | Depth | RGB | Depth | RGB | Depth |
| K $\rightarrow$ X | 0.33 | 0.13 | 0.37 | 0.14 | 0.28 | 0.12 |
| X $\rightarrow$ K | 0.25 | 0.12 | 0.19 | 0.12 | 0.21 | 0.12 |
| K $\rightarrow$ R | 0.26 | 0.22 | 0.22 | 0.17 | 0.25 | 0.18 |
| X $\rightarrow$ R | 0.26 | 0.20 | 0.24 | 0.22 | 0.23 | 0.17 |
| KX $\rightarrow$ R | 0.24 | 0.22 | 0.26 | 0.18 | 0.22 | 0.19 |
| AVG | 0.27 | 0.18 | 0.26 | 0.17 | **0.24** | **0.15** |

Table 3.12 Cost analysis on SUN RGB-D with ResNet-18 in DA setting. Hardware - CPU: Intel(R) Core(TM) i7-9800X CPU @ 3.80GHz, GPU (x1): Nvidia TITAN Xp.

| | **Cost analysis** | | | |
|---|---|---|---|---|
| | FLOPs | Training Time (ms) | Inference Time (ms) | Accuracy (Avg.) |
| Tran-Adapt Fusion++ | $3.09 \times 10^{10}$ | $6.85 \times 10^{7}$ | 3.41 | **55.13** |
| CycleGAN [59] | $9.48 \times 10^{10}$ | $1.39 \times 10^{8}$ | 3.69 | 54.63 |

encoder-decoder structure is trained to learn a mapping of the source domain in the target style, then the produced target-style images are used as training set for the standard network. The high number of FLOPs and time for training is due to the double training required to obtain good results. Our solution instead is an end-to-end approach able to reach higher performance than CycleGAN using fewer resources and a significantly lower amount of time.

### 3.2.4 Conclusions

In this section, we focused on cross-domain learning for multi-modal scene recognition. We started by observing the large variability introduced by the plethora of 3D cameras used to collect images in existing scene databases and highlighted that this can cause a significant domain shift that needs a tailored solution. We defined a testbed for studying this problem and performed an evaluation benchmark on several existing methods to evaluate how approaches originally developed for single-domain multi-modal scene recognition and multi-modal cross-domain object classification work on the considered task. Moreover, we presented a classification model that exploits self-supervised inter-modality translation as an auxiliary task to reduce domain shift. Our Translate-to-Adapt successfully outperforms the competitors, showing the effectiveness of its self-supervised task in scene recognition.

We believe that the novel setting can be of interest to the computer vision and robotics community: the testbed and the experimental analysis are proposed as baselines to pave the way for future research.

# Chapter 4

# Auxiliary Self-Supervision for Partial and Open-Set Cross-Domain Learning

*In this Chapter we face problem settings that combine category and domain shift. We show how self-supervision remains an effective auxiliary task also in these more challenging cases. Specifically, we present two solutions for Unsupervised Domain Adaptation, respectively under PDA and Open-Set assumptions. The first one is an extended version of JiGen (Section 3.1) where self-supervision is exploited to focus on the shared categories. The second one (ROS) uses rotation recognition to detect unknown samples in the target domain.*

## 4.1 Tackling Partial Domain Adaptation with Self-Supervision

*Reproduced with permission from Springer Nature: Bucci, S., D'Innocente, A., & Tommasi, T. Tackling partial domain adaptation with self-supervision. International Conference on Image Analysis and Processing (pp. 70-81) Springer, Cham (ICIAP 2019)*

Closed-set DA and DG models are not suitable when a part of the source classes is missing at test time. Those models show a drop in performance which indicates the effect of negative transfer in the PDA setting. Indeed, the model has to simultaneously handle two complex tasks: one that exploits all the available labeled source data to train a reliable classification model in the source domain and another that estimates and minimizes the marginal distribution difference between source and target, but disregards the potential presence of a conditional distribution shift. Recent research has shown that this second task can be substituted with self-supervised objectives that are not affected by the domain identity of each sample. In this section, we explore the use of jigsaw puzzles and rotation recognition as self-supervised tasks for DA when some source classes are missing at test time. We investigate how these tasks perform in the PDA setting and how they can be modified to reduce the number of required parameters. We show results on three different datasets that demonstrate the superiority of our approach against several competitors that use specific strategies to down-weight samples from classes that are not present in the target. We also discuss how combining this re-scaling process with self-supervision leads to further improvements in performance.

### 4.1.1 Method

**Problem Setting.** Let us introduce the technical terminology for the PDA scenario. We have a source domain $D_s = \{(\mathbf{x}_i^s, \mathbf{y}_i^s)\}_{i=1}^{n^s}$, drawn from a distribution $S$ with $n^s$ samples and a target domain $D_t = \{\mathbf{x}_j^t\}_{j=1}^{n^t}$, drawn from the distribution $T$ with $n^t$ samples. The label space of the target domain is contained in that of the source domain $Y_t \subseteq Y_s$. This makes the problem more difficult than standard UDA, where the goal is simply to learn domain-invariant feature models dealing with the marginal shift $S \neq T$. In PDA, it is also necessary to learn class-discriminative

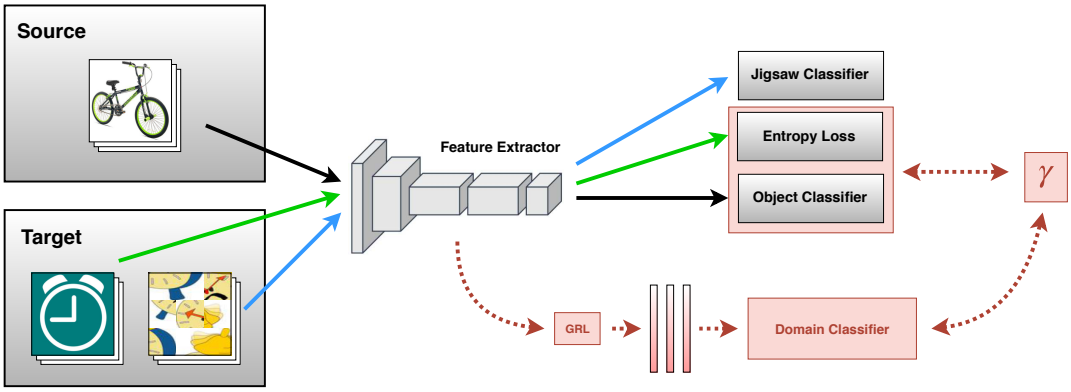



Fig. 4.1 Schematic representation of our approach. All the parts in gray illustrate the main blocks of the network with the solid line arrows indicating the contribution of each group of training samples to the corresponding final tasks and related optimization objectives according to the assigned blue/green/black colors. The blocks in red describe the domain adversarial classifier with the gradient reversal layer (GRL) and a weighting procedure for source samples (weight $\gamma$), which can be incorporated into our method.

models while taking into account the difference in label spaces. This task can be formulated as a multi-task learning problem [217], where the goal is to learn both class-discriminative and domain-invariant feature models while avoiding negative transfer. Instead of just focusing on reducing the discrepancy between the feature domains, one could consider some inherent characteristics that are present in any visual domain, regardless of the assigned labels. By incorporating these characteristics as an additional task, it can regularize the overall model and improve generalization, thus reducing domain bias. This reasoning is at the basis of the approach proposed in Section 3.1 which is also the starting point for the proposed approach in PDA. We point the reader to Section 3.1 where the approach and the notation used in the following paragraphs are described.

**Self-Supervision for Partial Domain Adaptation.** We consider the loss function 3.2 where the two $L_p$ terms are used to reduce the domain shift in the learned feature representation. However, it appears that their co-presence is redundant: the features are already chosen to minimize the source classification loss and the self-supervised jigsaw puzzle task on the target back-propagates its effect directly on the learned features inducing a cross-domain adjustment. With the idea of streamlining the learning process and eliminating unnecessary components, we decided to remove the source jigsaw puzzle term from our approach, by setting $\alpha^s = 0$. This not only simplifies the learning process by reducing the number of hyper-parameters, but it

also allows the self-supervised module to focus solely on the target domain samples, without involving any additional classes from the source domain. The structure of our approach is shown in Figure 4.1.

**Combining Self-Supervision with other PDA Strategies.** To further optimize the performance of our approach and focus on the shared classes, an additional weighting mechanism can be applied, similar to the one presented in [123]. This mechanism consists in accumulating the source classification output on the target data $\gamma = \frac{1}{n^t}\sum_{j=1}^{n^t} \hat{\mathbf{y}}_j^t$, normalizing the resulting vector $\gamma \leftarrow \gamma / \max(\gamma)$, and obtaining a $|Y_t|$-dimensional vector to use as a weighting factor to emphasize the classes that have a higher contribution. This modified version of our approach is intended to improve the overall performance and focus on the shared classes. Additionally, we can incorporate a domain classifier $G_d$ with a gradient reversal layer, similar to the approach used in [56], and adversarially increase the binary cross-entropy to enhance domain confusion, taking into account the class weighting method for the source samples. In more formal terms, the final objective of our multi-task problem is

$$\begin{aligned}\arg \min_{\theta_f,\theta_c,\theta_p} \max_{\theta_d} \; & \frac{1}{n^s}\sum_{i=1}^{n^s} \gamma_y \Big( L_c(G_c(G_f(\mathbf{x}_i^s), y_i^s)) + \lambda \log(G_d(G_f(\mathbf{x}_i^s))) \Big) + \\ & \frac{1}{n^t}\sum_{j=1}^{n^t} \gamma_y \Big( \eta H(G_c(G_f(\mathbf{x}_j^t))) + \lambda \log(1 - G_d(G_f(\mathbf{x}_j^t))) \Big) + \\ & \alpha_t \frac{1}{K^t}\sum_{k=1}^{K^t} L_p(G_p(G_f(\mathbf{z}_k^t), p_k^t)) \,, \end{aligned} \tag{4.1}$$

where $\lambda$ is a hyper-parameter that adjusts the importance of the introduced domain discriminator. We considered the same scheduling of [56] to change the value of $\lambda$ increasing the importance of the domain discriminator with the training epochs to avoid noisy signals of the early stages of the training.

### 4.1.2 Experiments

**Implementation Details.** The main backbone of our network, $G_f$, is a ResNet-50 pre-trained on ImageNet, while the specific object and self-supervised task classifiers $G_c, G_p$ are implemented with an ending fully connected layer. The domain classifier $G_d$ is built by adding three fully connected layers after the final pooling layer of the

main backbone, using a sigmoid activation function as in previous works [56]. By training the network end-to-end we fine-tune all the feature layers, while $G_c, G_p$ and $G_d$ are learned from scratch. The network for DG has two main hyperparameters: $\alpha$, which weights the self-supervised loss, and the data bias parameter $\beta$, which regulates the data input process. The self-supervised variants of the images are input into the network alongside the original images, with $\beta$ specifying the ratio between them. For example, $\beta = 0.6$ means that 60% of the images in each batch are original while the remaining 40% are rotated or composed of shuffled patches. In DA, there are additional parameters: $\alpha$ is separated into $\alpha_s$ and $\alpha_t$ for the source and target data respectively, and $\eta$ is the weight assigned to the entropy loss. Lastly, $\lambda$ is used to balance the importance of the gradient reversal layer when included in PDA. In the case of jigsaw puzzle task, two additional parameters are needed: the grid size used to divide the images into patches, and the number of permutations considered for the puzzle. Following [33], the size of the grid is fixed to $3 \times 3$ and the number of permutations is set at $P = 30$. We used a standard data augmentation protocol for our experiments, which involved randomly cropping the images to retain between 80-100% of the original image, applying random horizontal flipping, and converting a random 10% of the image tiles to grayscale, as proposed in [29]. In terms of training, we used the Stochastic Gradient Descent (SGD) optimizer with a momentum of 0.9, a weight decay of 0.0005, a batch size of 64 and 24 training epochs. The initial learning rate was set to 0.0005 and the hyper-parameters of our model were fine-tuned by setting $\alpha_s$ to 0, $\alpha_t$ to 1, and $\eta$ to 0.2.

**Model Selection.** As standard practice, we set aside 10% of the source training data as a validation set used to evaluate the model after each epoch. We combine the accuracy obtained during the current epoch, denoted as $A_e$ with the previous ones through a weighted average $A_e \leftarrow wA_{e-1} + (1-w)A_e$. The final model used on the target data is chosen as the one with the highest accuracy over all the epochs $e = 1, \ldots, E$. This method allows for a more reliable selection of the best-trained model, preventing the choice of a model that may have overfitted to the validation set. For all our experiments, we kept at $w = 0.6$. We highlight that the smoothing procedure proposed has been applied to all our experiments and we adopted the same hyper-parameters values for all the domain pairs within and across all the datasets. So, as opposed to previous works [123, 122], we did not select tailored values for the parameters for each sub-task that would lead to more performance gains.

**Baselines.** We compare our approach with several PDA methods: SAN [122], PADA [123], DRCN [248], IWAN [124], ETN [249] and AFN [53]. Specifically, SAN, PADA, and DRCN make use of the predictions from the source model to determine the distribution of target classes. IWAN employs a separate feature extractor for each domain and derives the weight of the source samples from a domain recognition model rather than from the source classifier. The most recent method ETN employs only relevant source examples to train both the label classifier and domain discriminator: the transferability of each source sample is determined through an auxiliary domain discriminator, which is not used in the adaptation process. AFN aligns sample feature norms demonstrating to be resistant to negative transfer without needing any weighting mechanism. Lastly, we also include standard DA approaches, DAN and DANN, to demonstrate the impact of methods that were not specifically designed to handle PDA.

**Results.** Tables 4.1 and 4.2 show the results for Office-31 and VisDA2017 datasets. Both tables are divided into four sections. The first section shows the results obtained without using any adaptation techniques or using standards DA approaches. The second section shows the results obtained using algorithms specifically designed for PDA. The third section presents the performance of the norm-based adaptation techniques HAFN and SAFN together with its ResNet-50 baseline. Lastly, the fourth section of the tables shows the results of our proposed approach.

The tables demonstrate that the Jigsaw and Rotation methods outperform the adaptive techniques of the first group. When comparing our method to the PDA techniques in the second group, our method outperforms on VisDA2017 dataset, despite the fact that some of the competitors use a ten-crop image evaluation method, denoted by a star ($^*$) in the table. On Office-31 dataset, the best result is obtained by ETN, however, it has a dedicated parameter selection method for each domain pair, while our method uses fixed and shared parameters across all domain pairs. Finally, the results for HAFN and SAFN, in the third group show the effectiveness of norm-based methods for PDA, but their results are not better than our method. Although not specifically designed for partial domain adaptation, the results obtained demonstrate that our auxiliary self-supervised task can effectively support adaptation in this scenario. Given that our solution is orthogonal to the sample selection strategies, we further tried to combine them together to evaluate if they complement each other. Specifically, we focused on Office-31 and the Jigsaw: we estimated the target class statistics through the weight $\gamma$ and included also a domain discriminator

Table 4.1 Accuracy (%) in the PDA setting on Office-31 dataset (source: 31 classes, target: 10 classes). The results are obtained by averaging over three repetitions of each run. With * we indicate ten-crop testing.

| **Office-31-PDA** | **A→W** | **D→W** | **W→D** | **A →D** | **D→A** | **W→A** | **Avg.** |
|---|---|---|---|---|---|---|---|
| | | | | **Resnet-50** | | | |
| Resnet-50 | 75.37 | 94.13 | 98.84 | 79.19 | 81.28 | 85.49 | 85.73 |
| DAN[44] | 59.32 | 73.90 | 90.45 | 61.78 | 74.95 | 67.64 | 71.34 |
| DANN[56] | 75.56 | 96.27 | 98.73 | 81.53 | 82.78 | 86.12 | 86.50 |
| IWAN [124] | 89.15 | 99.32 | 99.36 | 90.45 | 95.62 | 94.26 | 94.69 |
| SAN*[122] | 93.90 | 99.32 | 99.36 | 94.27 | 94.15 | 88.73 | 94.96 |
| PADA*[123] | 86.54 | 99.32 | 100 | 82.17 | 92.69 | 95.41 | 92.69 |
| DRCN*[248] | 86.00 | 88.05 | 95.60 | 100.0 | 95.80 | 100.0 | 94.30 |
| ETN [249] | 94.52 | 100.0 | 100.0 | 95.03 | 96.21 | 94.64 | 96.73 |
| Resnet-50 | 76.05 | 97.52 | 99.36 | 83.23 | 83.89 | 86.18 | 87.71 |
| HAFN [53] | 79.89 | 97.63 | 99.57 | 84.93 | 89.59 | 90.08 | 90.28 |
| SAFN [53] | 84.52 | 97.40 | 98.94 | 84.50 | 92.07 | 92.90 | 91.72 |
| SAFN+ENT [53] | 87.57 | 98.08 | 99.36 | 88.11 | 93.95 | 93.77 | 93.47 |
| Resnet-50 | 74.35 | 93.90 | 96.81 | 78.13 | 78.46 | 86.81 | 84.74±0.71 |
| **Jigsaw** | 91.75 | 94.12 | 98.93 | 90.87 | 89.95 | 93.42 | 93.18±0.46 |
| **Rotation** | 87.91 | 95.14 | 99.57 | 86.84 | 88.73 | 93.98 | 92.03±1.29 |
| **Jigsaw*-$\gamma$** | 99.32 | 94.69 | 99.36 | 96.39 | 86.36 | 94.22 | 95.06±1.86 |
| **Jigsaw*-$\gamma,\lambda$** | 99.66 | 94.46 | 99.57 | 97.67 | 87.33 | 94.26 | 95.49±1.19 |

weighted by the parameter $\lambda$, following [123]. The last two rows of Table 4.1 show that incorporating target statistics improves the network's focus on shared categories, resulting in an average improvement of 2% in accuracy compared to Jigsaw method, achieving results comparable to ETN (considering standard deviation). Additionally, by comparing the $\gamma$ values for the A→W domain shift, we observe that Jigsaw-$\gamma$ is more precise in identifying the missing classes of the target (see Figure 4.2). We indicate with Jigsaw-$\gamma,\lambda$ the case that includes the domain classifier: since the produced features are already well aligned across domains, we fixed $\lambda$-max to 0.1 and observed a further small average improvement. From the last bar plot on the right of Figure 4.2 we also observe a better identification of the target classes.

### 4.1.3 Computational Cost

In Table 4.3 we show the computational cost of the best performing configuration of our multi-task approach (excluding the results obtained with ten-crop test) and its best competitor IWAN [124] for the experiments in Table 4.1. We exclude ETN

Table 4.2 Accuracy (%) in the PDA setting on VisDA2017 dataset (source: 12 classes, target: 6 classes). The results are obtained by averaging over three repetitions of each run.

| **VisDA2017-PDA** | **Synthetic→Real** |
|---|---|
| **Resnet-50** | |
| Resnet-50 | 45.26 |
| DAN[44] | 47.60 |
| DANN[56] | 51.01 |
| PADA*[123] | 53.53 |
| DRCN*[248] | 58.20 |
| Resnet-50 | 49.89 |
| HAFN[53] | 65.06 |
| SAFN[53] | 67.65 |
| SAFN+ENT*[53] | 70.40 |
| Resnet-50 | 58.65±0.66 |
| **Jigsaw** | 68.18±1.36 |
| **Rotation** | **71.95±0.39** |

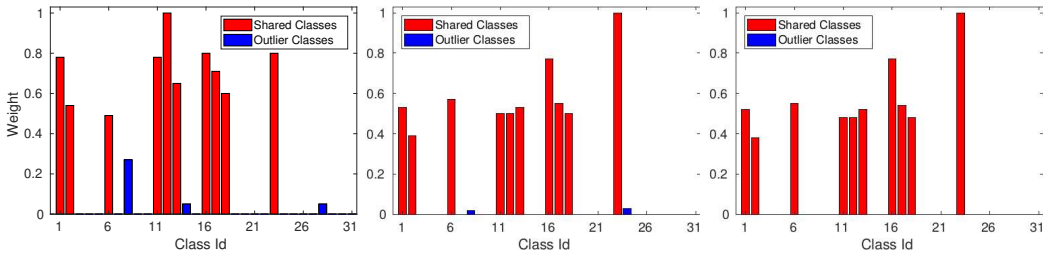



Fig. 4.2 Histogram showing the values of $\gamma$ vector which correspond to the class weight learned by PADA, SSPDA-$\gamma$ and SSPDA-PADA for the A→W domain shift.

[249] from this analysis as the need of running a dedicated parameter selection for each domain pair makes it significantly inefficient. We present the total number of FLOPs (Floating Point Operations) required for a single forward pass of the network, the training, and the inference time (in milliseconds). Even if IWAN shows better performance ($+\sim 1.5$), it has a significant higher number of FLOPs with also higher convergence time with respect to Jigsaw. Indeed, in IWAN each domain has its own feature extractor and the source sample weight is obtained from the domain recognition model.

Table 4.3 Cost analysis on Office-31 with ResNet-50 in PDA setting. Hardware - CPU: Intel(R) Core(TM) i7-9800X CPU @ 3.80GHz, GPU (x1): Nvidia TITAN Xp.

| | **Cost analysis** | | | |
|---|---|---|---|---|
| | FLOPs | Training Time (ms) | Inference Time (ms) | Accuracy (Avg.) |
| Jigsaw | $4.14 \times 10^9$ | $3.13 \times 10^6$ | 3.59 | $93.18 \pm 0.46$ |
| IWAN [124] | $8.29 \times 10^9$ | $2.52 \times 10^7$ | 3.61 | **94.69** |

### 4.1.4 Conclusions

In this section, we have presented how the self-supervised jigsaw puzzle and rotation recognition tasks can be used for domain adaptation in the challenging partial setting where some of the source classes are missing in the target. The high-level knowledge captured by the spatial co-location of patches and rotation prediction, being unsupervised with respect to the object content, can be applied to unlabeled target samples, helping to bridge the domain gap without negative transfer. We also demonstrated that the proposed solution can be combined with other existing partial domain adaptation methods, resulting in improved recognition performance, especially in identifying categories absent in the target.

## 4.2 ROS: On the Effectiveness of Image Rotation for Open-Set DA

*Reproduced with permission from Springer Nature: Bucci, S., Borlino, F. C., Caputo, B., & Tommasi, T., On the effectiveness of image rotation for open set domain adaptation. European Conference on Computer Vision, Springer, Cham (ECCV 2020)*

In open-set DA it's important to recognize and isolate samples from unknown classes before reducing the domain shift to prevent negative transfer. Recent research has shown that self-supervision can be used for anomaly detection, to distinguish between normal and anomalous data [137, 168]. However, these works only address the binary problem and only consider a single domain. In this section, we propose to use the properties of self-supervision to address both cross-domain robustness and novelty detection in the task of *Open-Set Domain Adaptation* (OSDA). To accomplish this, we propose a two-stage method, named Rotation-based Open Set (ROS), which is shown in Figure 4.3.

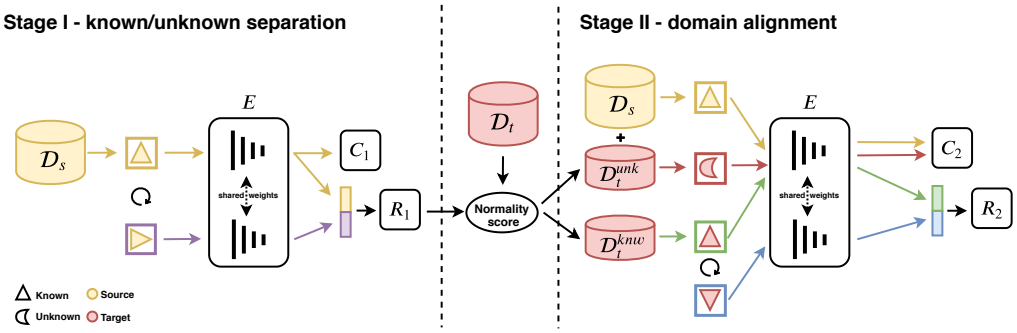



Fig. 4.3 Schematic illustration of our Rotation-based Open Set (**ROS**). In Stage 1, we use the source dataset $D_s$ to train an encoder $E$, a semantic classifier $C_1$, and a multi-rotation classifier $R_1$, for known/unknown separation. $C_1$ is trained using the features of the original images, and $R_1$ is trained using the concatenated features of the original and rotated images. After training, we use the prediction of $R_1$ on the target dataset $D_t$ to generate a normality score that separates the target samples into a known target dataset $D_t^{knw}$ and an unknown target dataset $D_t^{unk}$. In Stage 2, we train $E$, the semantic+unknown classifier $C_2$, and the rotation classifier $R_2$ to align the source and target distributions and to recognize the known classes while rejecting unknowns. $C_2$ is trained using original images from $D_s$ and $D_t^{unk}$, and $R_2$ is trained using the concatenated features of the original and rotated known target samples.

In the first stage, it separates known and unknown samples in the target domain by training the model on a modified version of the rotation task whose goal is to predict the relative rotation between a reference image and its rotated counterpart. In the second stage, the method reduces the domain shift between the source and known target domains by using again the rotation recognition task. The result is a classifier that can predict each target sample as one of the known classes or reject it as an unknown sample.

When evaluating ROS on the popular benchmark datasets Office-31 [7] and Office-Home [4], we expose the reproducibility problem of existing OSDA approaches and assess them with a new evaluation metric that better represents the performance of open-set methods.

In summary, the main contributions of this section are:

- a novel OSDA method that uses rotation recognition for both known/unknown target separation and domain alignment;
- a new OSDA evaluation metric that properly considers both known class recognition and unknown samples rejection;
- an extensive experimental benchmark against existing OSDA methods with two conclusions: (a) we put under the spotlight the urgent need of a rigorous experimental validation to guarantee result reproducibility; (b) our ROS defines the new state-of-the-art on two benchmark datasets.

The implementation of our method in Pytorch, along with instructions to replicate our experiments, can be found at the following link: https://github.com/silvia1993/ROS

### 4.2.1 Method

**Problem formulation.** We denote the labeled source domain as $D_s = (\mathbf{x}_j^s, y_j^s) j = 1^{N_s} \sim p_s$ drawn from distribution $p_s$, and the unlabeled target domain as $D_t = \mathbf{x}_j^t j = 1^{N_t} \sim p_t$ drawn from distribution $p_t$. In Open-Set Domain Adaptation, the source domain is associated with a set of known classes $y^s \in 1, \ldots, |C_s|$ that is shared with the target domain $C_s \subset C_t$, but the target also has a set of additional classes $C_{t \setminus s}$ which are considered unknown. Similarly to Closed-Set Domain Adaptation, we assume $p_s \neq p_t$ and we further have that $p_s \neq p_t^{C_s}$, where $p_t^{C_s}$ denotes the distribution of

the target domain belonging to the shared label space $C_s$. In OSDA, we face both a domain gap ($p_s \neq p_t^{C_s}$) and a category gap ($C_s \neq C_t$). The goal of OSDA approaches is to assign the target samples to one of the $|C_s|$ shared classes or reject them as unknown, using only annotated source samples and the unlabeled target samples available. A crucial metric in characterizing OSDA problems is their *openness* level which expresses the proportion of known classes in the target domain. For a given dataset pair $(D_s, D_t)$, following the definition of [250], the openness $\mathbb{O}$ is calculated as $\mathbb{O} = 1 - \frac{|C_s|}{|C_t|}$. In closed-set domain adaptation $\mathbb{O} = 0$ while in OSDA $\mathbb{O} > 0$.

**Overview.** When developing a method for OSDA, we face two main challenges: *negative transfer* and *known/unknown separation*. Negative transfer happens when the entire source and target distributions are aligned, and this includes unknown target samples which are mistakenly aligned with source data. To prevent this, cross-domain adaptation should focus only on the shared $C_s$ classes, which reduce the gap between $p_t^{C_s}$ and $p_s$. This leads to the challenge of known/unknown separation: identifying each target sample as belonging to one of the shared classes $C_s$ (known) or one of the target-specific classes $C_{t\setminus s}$ (unknown). With this in mind, we divide our method into two stages: (i) we separate the target samples into known and unknown, and (ii) we align the target samples identified as known with the source samples (see Figure 4.3). The first stage is treated as an anomaly detection problem, where the unknown samples are considered anomalies. The second stage is treated as a Closed-Set Domain Adaptation problem between source and known target distribution. We draw inspiration from recent advancements in anomaly detection and Closed-Set Domain Adaptation [191, 137] to address both stages by utilizing self-supervised techniques. Specifically, we use two variations of the rotation classification task to calculate a normality score for the known/unknown separation of the target samples, and to reduce the domain gap.

**Rotation recognition for open set domain adaptation.** Let's denote with $rot90(\mathbf{x}, i)$ the function that rotates clockwise a 2D image $\mathbf{x}$ by $i \times 90°$. Rotation recognition is a self-supervised task that involves rotating an image $\mathbf{x}$ by a random $i$ in $[1,4]$ and using a CNN to predict $i$ from the rotated image $\tilde{\mathbf{x}} = rot90(\mathbf{x}, i)$. We represent with $|r| = 4$ the cardinality of the label space for this classification task. To apply rotation recognition effectively to Open-Set Domain Adaptation, we propose the following variations.

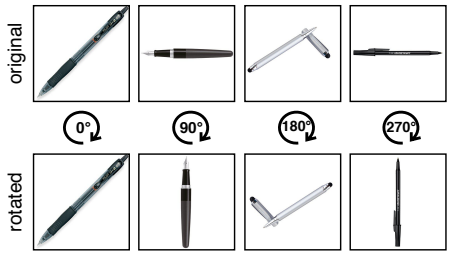



Fig. 4.4 Are you able to infer the rotation degree of the rotated images without looking at the respective original one?

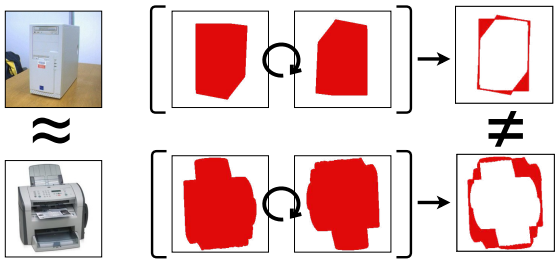


Fig. 4.5 The objects on the left may be confused. The relative rotation guides the network to focus on discriminative shape information

**Relative rotation.** Inferring the rotation angle of an image without comparing it to its original (non-rotated) version is a challenging problem as object classes could not have a consistent orientation across the dataset (see Figure 4.4). However, comparing the original and rotated image and determining the relative rotation between them is a well-defined task. In light of this, we make a change to the standard rotation classification task [6] by including the original image as a reference point. The rotation classifier is trained to predict the rotation angle, given the combined features of both the original (reference) and rotated image. As seen in Figure 4.5, this new form of relative rotation increases the discriminatory power of the learned features. It guides the network to focus more on specific shape details rather than on confusing texture information across different object classes.

**Multi-rotation classification.** In standard anomaly detection, samples from one particular semantic category are classified as normal, and samples from all other categories as anomalies. Rotation recognition has been effectively applied in this scenario, but its performance deteriorates when there is more than one semantic category included in the normal class [137]. This is the case when tackling known/unknown separation in OSDA, where we have $|C_s|$ semantic categories as known data. To address this issue, we introduce a modified approach to rotation recognition. Instead of treating it as a 4-class problem, we expand it to $(4 \times |C_s|)$ classes, where the set of classes represents the combination of semantic and rotation labels. This allows for a more accurate classification by taking into account both the semantic category and the rotation angle of the image. For example, if we rotate an image of category $y^s = 2$ by $i = 3$, its label for the multi-rotation classification task is $z^s = (y^s \times 4) + i = 11$. We provide further evidence of the effectiveness of this multi-rotation classification

task in the appendix. In the following, we indicate with $y, z$ the one-hot vectors respectively for the class and multi-rotation labels.

**Stage I: known/unknown separation.** To differentiate between known and unknown samples of $D_t$, we train a CNN to perform multi-rotation classification using the dataset $\tilde{D}_s = (\mathbf{x}_j^s, \tilde{\mathbf{x}}_j^s, z_j^s)_{j=1}^{4\times N_s}$. The network includes an encoder $E$ and two output branches, a multi-rotation classifier $R_1$, and a semantic label classifier $C_1$. The prediction of the rotation task is obtained on the original and rotated image produced by the encoder $\hat{\mathbf{z}}^s = \text{softmax}\big(R_1([E(\mathbf{x}^s), E(\tilde{\mathbf{x}}^s)])\big)$, the object prediction is obtained only from the features of the original image as $\hat{\mathbf{y}}^s = \text{softmax}\big(C_1(E(\mathbf{x}^s)\big)$. The objective function used to train the network is made by the cross-entropy loss for object recognition $L_{C_1}$ and $L_{R_1}$ that combines the cross-entropy and the center loss [251] for the multi-rotation classifier. So the total loss function is $L_1 = L_{C_1} + L_{R_1}$. More precisely,

$$L_{C_1} = -\sum_{j\in D_s} \mathbf{y}_j^s \cdot \log(\hat{\mathbf{y}}_j^s), \tag{4.2}$$

$$L_{R_1} = \sum_{j\in \tilde{D}_s} -\lambda_{1,1}\mathbf{z}_j^s \cdot \log(\hat{\mathbf{z}}_j^s) + \lambda_{1,2}||\mathbf{v}_j^s - \gamma(\mathbf{z}_j^s)||_2^2, \tag{4.3}$$

where $||.||_2$ is the $l_2$-norm operator, $\mathbf{v}_j$ represents the output of the penultimate layer of $R_1$ and $\gamma(\mathbf{z}_j)$ indicates the corresponding centroid of the class associated with $\mathbf{v}_j$. By using the center loss, we encourage the network to learn more compact and discriminative feature representations for each class. This makes the rotation classifier output more reliable as a metric for detecting samples from unknown categories.

Once the training is complete, we use the encoder $E$ and the rotation classifier $R_1$ to compute the *normality score* $N \in [0,1]$ for each target sample. This score is a measure of how well the sample aligns with the known classes, with high scores indicating that the sample is likely to be a known class, and low scores indicating that the sample is likely to be an unknown class. To compute this score, we evaluate the network's prediction on all the relative rotation variants of a target sample $\hat{\mathbf{z}}_i^t = \text{softmax}\big(R_1([E(\mathbf{x}^t), E(\tilde{\mathbf{x}}_i^t)])\big)_i$ and calculate their related entropy $H(\hat{\mathbf{z}}_i^t) = \big(\hat{\mathbf{z}}_i^t \cdot \log(\hat{\mathbf{z}}_i^t)/\log|C_s|\big)_i$ with $i = 1,\ldots,|r|$. We indicate with $[\hat{\mathbf{z}}^t]_m$ the $m$-th component of the $\hat{\mathbf{z}}^t$ vector. Then, we use this information to assign a score to each sample, which can be used to separate known samples from unknown samples.

The full expression of the normality score is:

$$N(\mathbf{x}^t) = \max \left\{ \max_{k=1,\dots,|C_s|} \left( \sum_{i=1}^{|r|} [\hat{\mathbf{z}}_i^t]_{k\times|r|+i} \right), \left( 1 - \frac{1}{|r|} \sum_{i=1}^{|r|} H(\hat{\mathbf{z}}_i^t) \right) \right\} . \tag{4.4}$$

In words, our method uses a rotation recognition task to calculate a score for each target sample, which indicates how likely it is to belong to one of the known classes. This score is based on the network's ability to correctly predict the class and orientation of the sample (first term in Equation 4.4, *Rotation Score*), as well as the confidence of the prediction (second term, *Entropy Score*). We use this score to separate known and unknown samples in the target dataset.

Finally, through the normality score obtained, we separate the target dataset into known $D_t^{knw}$ and unknown $D_t^{unk}$. It is made directly through the data statistics using the average of the normality score over the whole target $\bar{N} = \frac{1}{N_t} \sum_{j=1}^{N_t} N_j$, without the need to introduce any further parameters:

$$\begin{cases} x^t \in D_t^{knw} & \text{if} \quad N(x^t) > \bar{N} \\ x^t \in D_t^{unk} & \text{if} \quad N(x^t) < \bar{N} \, . \end{cases} \tag{4.5}$$

It is important to note that only $R_1$ plays a role in calculating the normality score, while $C_1$ is used for regularization and as a preparation for the next stage. For a detailed pseudo-code on how to compute $N$ and create $D_t^{knw}$ and $D_t^{unk}$, refer to the appendix.

**Stage II: domain alignment.** After identifying the target unknown samples, the situation is similar to the standard CSDA problem. Using $D_t^{knw}$ allows for closing the domain gap without the risk of negative transfer, while using $D_t^{unk}$ allows for expanding the original semantic classifier to recognize the unknown category. The network in Stage II, similar to that of Stage I, is made up of an encoder $E$ and two heads: a rotation classifier $R_2$ and a semantic label classifier $C_2$. The encoder is inherited from the previous stage. The heads also leverage the previous training phase but have two key differences with respect to Stage I: (1) The dimension of the output of $C_1$ is $|C_s|$, while $C_2$ has an output dimension of $|C_s| + 1$ due to the inclusion of the unknown class; (2) $R_1$ is designed to classify multiple rotations, with a $(4 \times |C_s|)$-dimensional output, while $R_2$ is used for classifying rotations with a 4-dimensional output. The prediction of the rotation is computed as $\hat{\mathbf{q}} = \text{softmax}\big(R_2([E(\mathbf{x}), E(\tilde{\mathbf{x}})])\big)$ and the

semantic prediction is computed as $\hat{\mathbf{g}} = \text{softmax}\big(C_2(E(\mathbf{x}))\big)$. The objective function $L_2 = L_{C_2} + L_{R_2}$ is made by $L_{C_2}$ for the classification composed by a supervised cross-entropy loss and an unsupervised entropy loss, and $L_{R_2}$ is the cross-entropy loss for the rotation prediction. The unsupervised entropy loss is employed in the semantic classification process to incorporate the unlabeled target samples that are recognized as known. This loss encourages the decision boundary to pass through regions with low density. More precisely,

$$L_{C_2} = -\sum_{j \in \{D_s \cup D_t^{unk}\}} \mathbf{g}_j \cdot \log(\hat{\mathbf{g}}_j) - \lambda_{2,1} \sum_{j \in D_t^{knw}} \hat{\mathbf{g}}_j \cdot \log(\hat{\mathbf{g}}_j), \tag{4.6}$$

$$L_{R_2} = -\lambda_{2,2} \sum_{j \in D_t^{knw}} \mathbf{q}_j \cdot \log(\hat{\mathbf{q}}_j)\ . \tag{4.7}$$

After the training is finished, $R_2$ is no longer needed and the target labels are predicted using $c_j^t = C_2(E(\mathbf{x}_j^t))$ for all $j = 1, \ldots, N_t$.

### 4.2.2 On reproducibility and Open-Set metrics

OSDA is a relatively new area of research that was first introduced in 2017. With increasing interest in this field, it is essential to ensure the *reproducibility* of the proposed methods and have an appropriate *metric* to evaluate them effectively.

**Reproducibility.** In recent years, the machine learning community has become increasingly aware of the reproducibility crisis [252–254]. Reproducing the results of state-of-the-art deep learning models is often difficult due to a combination of non-deterministic factors in standard benchmark environments and poor documentation from authors. Although the problem is yet to be fully resolved, various efforts have been made to promote reproducibility, such as checklists [255], challenges [256], and encouraging authors to submit their code. We contribute to this effort by re-running the state-of-the-art methods for OSDA and comparing them with the results reported in the papers (see Section 4.2.3). Our results are produced using the original public implementation along with the parameters reported in the paper and, in some cases, by communicating with the authors. We believe that this approach, as opposed to simply reproducing the results reported in the papers, can provide significant value to the community.

**Open set metrics.** The commonly used metrics for evaluating OSDA are the average class accuracy over the known classes $OS^*$, and the accuracy of the unknown class *UNK*. They are typically combined in $OS = \frac{|C_s|}{|C_s|+1} \times OS^* + \frac{1}{|C_s|+1} \times UNK$ as a measure of overall performance. However, we argue that treating the unknown as an additional class does not provide an appropriate metric (as previously demonstrated in [257]). For example, let's consider an approach that is not designed to handle unknown categories (*UNK*=0.0%) but has optimal accuracy over 10 known classes ($OS^*$=100.0%). Even though this algorithm is not suitable for open set scenarios as it totally ignores false positives, it still presents a high score of *OS*=90.9%. This effect on OS becomes more pronounced as the number of known classes increases, making the value of *UNK* meaningless. Considering that, we propose a new metric defined as the harmonic mean of $OS^*$ and *UNK*, $HOS = 2\frac{OS^* \times UNK}{OS^* + UNK}$. Unlike *OS*, *HOS* only provides a high score if the model is able to perform well on both known and unknown samples, regardless of $|C_s|$. Using the harmonic mean instead of a standard average penalizes large gaps between $OS^*$ and *UNK*.

### 4.2.3 Experiments

**Setup.** We evaluate ROS against state-of-the-art approaches previously described in Section 2.1: STA [24], OSBP [131], UAN [258], and AoD [132]. For each method, we conduct experiments using the official code provided by the authors, with the exact hyper-parameters specified in the corresponding papers. The only exception is AoD, as the authors have not released the code at the time of writing, so we report the values presented in their original paper. We would also like to note that STA presents a practical issue related to the similarity score used to separate known and unknown categories. The formulation is based on the *max* operator according to the Equation (2) in [24], but it appears to be based on *sum* in the implementation code. In our analysis, we considered both variants ($\text{STA}_{\text{sum}}$, $\text{STA}_{\text{max}}$) for the sake of completeness. All the results in this section, for both ROS and the baseline methods, are an average of three separate runs.

**Implementation details.** By following standard practice, we evaluate the performance of ROS on Office-31 using two different backbones ResNet-50 [219] and VGGNet [259], both pre-trained on ImageNet [214], focusing only on ResNet-50 for Office-Home. We used the same values for the hyper-parameters regardless of the backbone and the dataset used. Specifically, the batch size is 32 in both Stage I and

Stage II while the learning rate is 0.0003 which decreases following an inverse decay scheduling during training. We set the learning rate 10 times higher for the layers trained from scratch with respect to the pre-trained ones using SGD as optimizer with weight decay 0.0005 and momentum 0.9. The weight of the self-supervised task, in both stages, is set three times the weight of the semantic classification task, i.e. $\lambda_{1,1} = \lambda_{2,2} = 3$. The weight of the center loss in Stage I is $\lambda_{1,2} = 0.1$ while, in Stage II, the weight of the entropy loss is $\lambda_{2,1} = 0.1$. The model obtained in Stage I is used as starting point for the training in Stage II. Lastly, in Stage II the learning rate for the new unknown class is set to twice that of the known classes since these have already been learned in Stage I. More implementation details and sensitivity analysis of the hyper-parameters are provided in the Appendix.

**Results.**

*How does our method compare to the state-of-the-art?* The results of our experiment are presented in Table 4.4 and Table 4.5 and they show the average performance of ROS, across three runs for each domain shift, respectively of Office-31 and Office-Home. To discuss the results, we focus on the HOS metric, as it is a combination of the performance on known and unknown classes (see Section 4.2.2). Our approach outperforms the state-of-the-art in 13 out of 18 domain shifts and has the highest average performance on both Office-31 and Office-Home. The improvement in HOS gets to 2.2% compared to the second-best method, OSBP. Specifically, ROS has a significant advantage over STA, regardless of its max or sum implementation, and UAN is not a strong competitor due to its poor performance on the unknown class. Comparison against AoD is only possible when using VGG for Office-31 (we report the original results in gray in Table 4.5) showing a clear advantage of ROS. An examination of the results reveals that the strength of ROS primarily lies in its ability to distinguish between known and unknown samples. Specifically, while the performance of ROS on known samples is similar to that of other methods, its performance on unknown samples is remarkably superior. This can be observed in the t-SNE visualizations shown in Figure 4.6, where the features of known and unknown data are less separated in the second-best method OSBP compared to ROS.

*Is it possible to reproduce the reported results of the state-of-the-art?* An analysis of the published papers on OSDA revealed inconsistencies in the reported results. For example, we noticed that some papers like OSBP presented different results for the same method and hyper-parameters in their pre-print [260] and published

Table 4.4 Accuracy (%) averaged over three runs of each method on Office-31 dataset using ResNet-50 and VGGNet as backbones

| **Office-31** | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **ResNet-50** | | | | | | | | | | | | | | | | | | | | | | |
| | A → W | | | A → D | | | D → W | | | W → D | | | D → A | | | W → A | | | **Avg.** | | |
| | OS* | UNK | **HOS** | OS* | UNK | **HOS** | OS* | UNK | **HOS** | OS* | UNK | **HOS** | OS* | UNK | **HOS** | OS* | UNK | **HOS** | OS* | UNK | **HOS** |
| $STA_{sum}$ [24] | 92.1 | 58.0 | 71.0 | 95.4 | 45.5 | 61.6 | 97.1 | 49.7 | 65.5 | 96.6 | 48.5 | 64.4 | 94.1 | 55.0 | 69.4 | 92.1 | 46.2 | 60.9 | 94.6 | 50.5 | 65.5±0.3 |
| $STA_{max}$ [24] | 86.7 | 67.6 | 75.9 | 91.0 | 63.9 | 75.0 | 94.1 | 55.5 | 69.8 | 84.9 | 67.8 | 75.2 | 83.1 | 65.9 | 73.2 | 66.2 | 68.0 | 66.1 | 84.3 | 64.8 | 72.5±0.8 |
| OSBP [131] | 86.8 | 79.2 | **82.7** | 90.5 | 75.5 | **82.4** | 97.7 | 96.7 | **97.2** | 99.1 | 84.2 | 91.1 | 76.1 | 72.3 | 75.1 | 73.0 | 74.4 | 73.7 | 87.2 | 80.4 | 83.7±0.4 |
| UAN [258] | 95.5 | 31.0 | 46.8 | 95.6 | 24.4 | 38.9 | 99.8 | 52.5 | 68.8 | 81.5 | 41.4 | 53.0 | 93.5 | 53.4 | 68.0 | 94.1 | 38.8 | 54.9 | 93.4 | 40.3 | 55.1±1.4 |
| **ROS** | 88.4 | 76.7 | 82.1 | 87.5 | 77.8 | **82.4** | 99.3 | 93.0 | 96.0 | 100.0 | 99.4 | **99.7** | 74.8 | 81.2 | **77.9** | 69.7 | 86.6 | **77.2** | 86.6 | 85.8 | **85.9±0.2** |
| **VGGNet** | | | | | | | | | | | | | | | | | | | | | | |
| | A → W | | | A → D | | | D → W | | | W → D | | | D → A | | | W → A | | | **Avg.** | | |
| | OS* | UNK | **HOS** | OS* | UNK | **HOS** | OS* | UNK | **HOS** | OS* | UNK | **HOS** | OS* | UNK | **HOS** | OS* | UNK | **HOS** | OS* | UNK | **HOS** |
| OSBP [131] | 79.4 | 75.8 | 77.5 | 87.9 | 75.2 | 81.0 | 96.8 | 93.4 | **95.0** | 98.9 | 84.2 | 91.0 | 74.4 | 82.4 | **78.2** | 69.7 | 76.4 | 72.9 | 84.5 | 81.2 | 82.6±0.8 |
| **ROS** | 80.3 | 81.7 | **81.0** | 81.8 | 76.5 | 79.0 | 99.5 | 89.9 | 94.4 | 99.3 | 100.0 | **99.7** | 76.7 | 79.6 | 78.1 | 62.2 | 91.6 | **74.1** | 83.3 | 86.5 | **84.4±0.2** |
| AoD [132] | 87.7 | 73.4 | 79.9 | 92.0 | 71.1 | **79.3** | 99.8 | 78.9 | 88.1 | 99.3 | 87.2 | 92.9 | 88.4 | 13.6 | 23.6 | 82.6 | 57.3 | 67.7 | 91.6 | 63.6 | 71.9 |

Table 4.5 Accuracy (%) averaged over three runs of each method on Office-Home dataset using ResNet-50 as backbone.

| **Office-Home** | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Pr → Rw | | | Pr → Cl | | | Pr → Ar | | | Ar → Pr | | | Ar → Rw | | | Ar → Cl | | | | | |
| | OS* | UNK | **HOS** | OS* | UNK | **HOS** | OS* | UNK | **HOS** | OS* | UNK | **HOS** | OS* | UNK | **HOS** | OS* | UNK | **HOS** | | | |
| $STA_{sum}$ [24] | 78.1 | 63.3 | 69.7 | 44.7 | 71.5 | 55.0 | 55.4 | 73.7 | 63.1 | 68.7 | 59.7 | 63.7 | 81.1 | 50.5 | 62.1 | 50.8 | 63.4 | 56.3 | | | |
| $STA_{max}$ [24] | 76.2 | 64.3 | 69.5 | 44.2 | 67.1 | 53.2 | 54.2 | 72.4 | 61.9 | 68.0 | 48.4 | 54.0 | 78.6 | 60.4 | 68.3 | 46.0 | 72.3 | 55.8 | | | |
| OSBP [131] | 76.2 | 71.7 | 73.9 | 44.5 | 66.3 | 53.2 | 59.1 | 68.1 | **63.2** | 71.8 | 59.8 | 65.2 | 79.3 | 67.5 | 72.9 | 50.2 | 61.1 | 55.1 | | | |
| UAN [258] | 84.0 | 0.1 | 0.2 | 59.1 | 0.0 | 0.0 | 73.7 | 0.0 | 0.0 | 81.1 | 0.0 | 0.0 | 88.2 | 0.1 | 0.2 | 62.4 | 0.0 | 0.0 | | | |
| **ROS** | 70.8 | 78.4 | **74.4** | 46.5 | 71.2 | **56.3** | 57.3 | 64.3 | 60.6 | 68.4 | 70.3 | **69.3** | 75.8 | 77.2 | **76.5** | 50.6 | 74.1 | **60.1** | | | |
| | Rw → Ar | | | Rw → Pr | | | Rw → Cl | | | Cl → Rw | | | Cl → Ar | | | Cl → Pr | | | **Avg.** | | |
| | OS* | UNK | **HOS** | OS* | UNK | **HOS** | OS* | UNK | **HOS** | OS* | UNK | **HOS** | OS* | UNK | **HOS** | OS* | UNK | **HOS** | OS* | UNK | **HOS** |
| $STA_{sum}$ | 67.9 | 62.3 | 65.0 | 77.9 | 58.0 | 66.4 | 51.4 | 57.9 | 54.2 | 69.8 | 63.2 | 66.3 | 53.0 | 63.9 | 57.9 | 61.4 | 63.5 | 62.5 | 63.4 | 62.6 | 61.9±2.1 |
| $STA_{max}$ | 67.5 | 66.7 | 67.1 | 77.1 | 55.4 | 64.5 | 49.9 | 61.1 | 54.5 | 67.0 | 66.7 | 66.8 | 51.4 | 65.0 | 57.4 | 61.8 | 59.1 | 60.4 | 61.8 | 63.3 | 61.1±0.3 |
| OSBP | 66.1 | 67.3 | 66.7 | 76.3 | 68.6 | 72.3 | 48.0 | 63.0 | 54.5 | 72.0 | 69.2 | **70.6** | 59.4 | 70.3 | **64.3** | 67.0 | 62.7 | 64.7 | 64.1 | 66.3 | 64.7±0.2 |
| UAN | 77.5 | 0.1 | 0.2 | 85.0 | 0.1 | 0.1 | 66.2 | 0.0 | 0.0 | 80.6 | 0.1 | 0.2 | 70.5 | 0.0 | 0.0 | 74.0 | 0.1 | 0.2 | 75.2 | 0.0 | 0.1±0.0 |
| **ROS** | 67.0 | 70.8 | **68.8** | 72.0 | 80.0 | **75.7** | 51.5 | 73.0 | **60.4** | 65.3 | 72.2 | 68.6 | 53.6 | 65.5 | 58.9 | 59.8 | 71.6 | **65.2** | 61.6 | 72.4 | **66.2± 0.3** |

versions [131]. Additionally, in some papers, important comparisons are missing like in AoD [132] that compares against the pre-print results of OSBP, while omitting the results of STA. To dispel these ambiguities and gain a better understanding of the current state-of-the-art methods, we conducted a reproducibility study by re-running the experiments and comparing the results with those reported in previous works. This analysis, presented in Table 4.6, focuses on Office-31 dataset and the OS metric is shown, as it is the only metric reported for some of the approaches. We used the original implementations provided by the authors, but despite this, the OS results obtained from re-running the experiments were between 1.3% and 4.9% lower than the originally published results. This gap in the results highlights the importance of providing detailed information for reproducing experimental results and calls for a more thorough reproducibility study, which is provided in the Appendix.

*Why is it important to use the HOS metric?* One major flaw of using OS as a metric for OSDA is demonstrated by the results of the UAN method. As seen in Table 4.4 and Table 4.5, UAN has an OS of 72.5% for Office-Home and 91.4% for Office-31. This mainly reflects UAN's ability to recognize known classes (OS*) but ignores its performance in identifying unknown samples (UNK). For example, for most domain

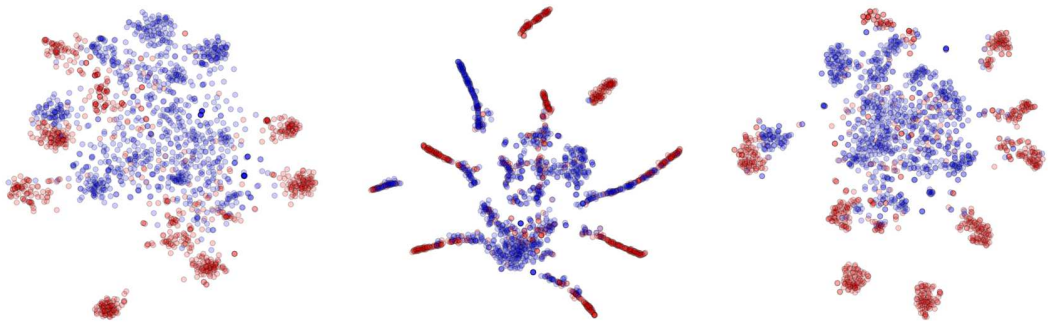



Fig. 4.6 t-SNE visualization of the target features for the W→A domain shift from Office-31. Red and blue points are respectively features of known and unknown classes.

Table 4.6 Reported vs reproduced OS accuracy (%) averaged over three runs

| **Reproducibility Study** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Office-31 (ResNet-50) | | | | | | Office-31 (VGGNet) | | |
| $STA_{sum}$ | | | UAN | | | OSBP | | |
| $OS_{reported}$ | $OS_{ours}$ | gap | $OS_{reported}$ | $OS_{ours}$ | gap | $OS_{reported}$ | $OS_{ours}$ | gap |
| 92.9 | 90.6±1.8 | **2.3** | 89.2 | 87.9±0.03 | **1.3** | 89.1 | 84.2 ±0.4 | **4.9** |

shifts in Office-Home, UAN does not assign (almost) any samples to the unknown class, resulting in UNK=0.0%. In contrast, HOS is a better indicator of the open set scenario as it only assumes a high value when both OS* and UNK are high.

*Is rotation recognition effective for known/unknown separation in OSDA?* To evaluate the effectiveness of rotation recognition for known/unknown separation, we compare the performance of our Stage I with that of Stage I of STA. Both methods have a two-stage structure, but while ROS uses a multi-rotation classifier, STA uses a multi-binary classifier. We measure performance using the area under the receiver operating characteristic curve (AUC-ROC) on the normality scores $N$ calculated on Office-31. The results in Table 4.7 show that the AUC-ROC of ROS (91.5) is significantly higher than that of the multi-binary classifier used by STA (79.9). The table also shows the results when removing the center loss from Equation (4.3) ($\lambda_{1,2} = 0$) (No Center Loss) and the anchor image is not considered (No Anchor) when training $R_1$. In both cases, the performance drops significantly compared to the full method but still outperforms the multi-binary classifier of STA.

*Why is the normality score defined the way it is?* Our normality score, as defined by Equation (4.4), is a combination of the rotation score and the entropy score. The rotation score measures the accuracy of $R_1$ in predicting the rotation of the target samples, while the entropy score represents the level of confidence of these

Table 4.7 Ablation Analysis on Stage I and Stage II

| **Ablation Study** | | | | | | | |
|---|---|---|---|---|---|---|---|
| **STAGE I** (AUC-ROC) | A → W | A → D | D → W | W → D | D → A | W → A | **Avg.** |
| **ROS** | 90.1 | 88.1 | 99.4 | 99.9 | 87.5 | 83.8 | **91.5** |
| Multi-Binary (from STA [24]) | 83.2 | 84.1 | 86.8 | 72.0 | 75.7 | 78.3 | 79.9 |
| ROS - No Center loss | 88.8 | 83.2 | 98.8 | 99.8 | 84.7 | 84.5 | 89.9 |
| ROS - No Anchor | 84.5 | 84.9 | 99.1 | 99.9 | 87.6 | 86.2 | 90.4 |
| ROS - No Rot. Score | 86.3 | 82.7 | 99.5 | 99.9 | 86.3 | 82.9 | 89.6 |
| ROS - No Ent. Score | 80.7 | 78.7 | 99.7 | 99.9 | 86.6 | 84.4 | 88.3 |
| ROS - No Center loss, No Anchor | 76.5 | 79.1 | 98.3 | 99.7 | 85.2 | 83.5 | 87.1 |
| ROS - No Rot. Score, No Anchor | 83.9 | 84.6 | 99.4 | 99.9 | 84.7 | 84.9 | 89.6 |
| ROS - No Ent. Score, No Anchor | 80.1 | 81.0 | 99.5 | 99.7 | 84.3 | 83.3 | 87.9 |
| ROS - No Rot. Score, No Center loss | 80.9 | 81.6 | 98.9 | 99.8 | 85.6 | 83.3 | 88.3 |
| ROS - No Ent. Score, No Center loss | 76.4 | 79.8 | 99.0 | 98.3 | 84.4 | 84.3 | 87.0 |
| ROS - No Ent. Score, No Center loss, No Anchor | 78.6 | 80.4 | 99.0 | 98.9 | 86.2 | 83.2 | 87.7 |
| ROS - No Rot. Score, No Center loss, No Anchor | 78.7 | 82.2 | 98.3 | 99.8 | 85.0 | 82.6 | 87.8 |
| **STAGE II** (HOS) | A → W | A → D | D → W | W → D | D → A | W → A | **Avg.** |
| **ROS** | 82.1 | 82.4 | 96.0 | 99.7 | 77.9 | 77.2 | **85.9** |
| ROS Stage I - GRL [56] Stage II | 83.5 | 80.9 | 97.1 | 99.4 | 77.3 | 72.6 | 85.1 |
| ROS Stage I - No Anchor in Stage II | 80.0 | 82.3 | 94.5 | 99.2 | 76.9 | 76.6 | 84.9 |
| ROS Stage I - No Anchor, No Entropy in Stage II | 80.1 | 84.4 | 97.0 | 99.2 | 76.5 | 72.9 | 85.0 |

predictions. In order to evaluate the importance of both scores in our methodology, we ran the experiment without either rotation score (No Rot. Score) or entropy score (No Ent. Score) observing the change in performance. The results in Table 4.7 in both cases the AUC-ROC significantly decreases compared to the full version, justifying our choice.

*Is rotation recognition effective for domain alignment in OSDA?* The use of rotation classification in the context of CSDA has been previously studied [191], however, its application in OSDA, where the shared target distribution may contain unknown samples, has not yet been explored. On the other hand, GRL [56] is a commonly used technique in OSDA methods. To compare the effectiveness of rotation recognition and GRL in this context, we evaluated the performance of our Stage II when replacing $R_2$ with a domain discriminator. Table 4.7 shows that rotation recognition performs similarly to, if not slightly better than, GRL. Additionally, we evaluated the role of relative rotation in Stage II and found that it improves performance over standard absolute rotation (No Anchor in Stage II), as seen in the last row of Table 4.7. Furthermore, the cosine distance between the source and the target domains without adaptation in Stage II is 0.188, and with our full method is 0.109, which confirms that rotation recognition is effective in reducing the domain gap.

*Is our method effective on problems with a high degree of openness?* Traditional approaches in OSDA usually involve scenarios where the number of shared and private target classes is balanced, with openness near 0.5. For instance, in the Office-

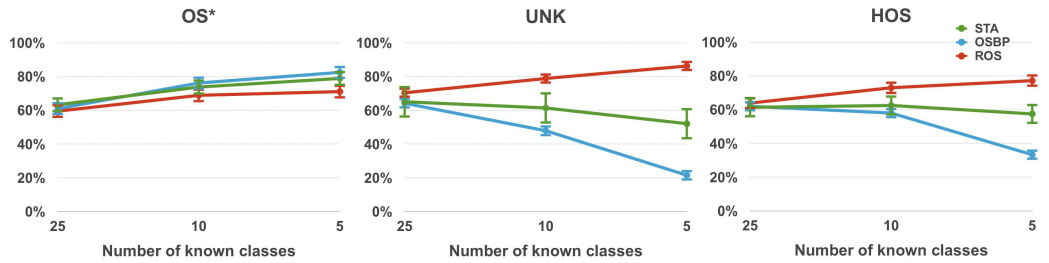



Fig. 4.7 Accuracy (%) averaged over the three openness configurations.

31 dataset, openness is $\mathbb{O} = 1 - \frac{10}{21} = 0.52$, and in the Office-Home dataset, it is $\mathbb{O} = 1 - \frac{25}{65} = 0.62$. However, in real-world cases, the number of unknown target classes may greatly surpass the known classes, with openness approaching 1. To explore this scenario, we consider Office-Home and, starting from the classes sorted with ID from 0 to 64 in alphabetic order, we define the following settings with increasing openness:: **25** known classes $\mathbb{O} = 0.62$, ID:0-24, 25-49, 40-64, **10** known classes $\mathbb{O} = 0.85$, ID:0-9, 10-19, 20-29, **5** known classes $\mathbb{O} = 0.92$, ID:0-4, 5-9, 10-14. Our findings, shown in Figure 4.7, indicate that, as the openness level increases, our major competitors like STA and OSBP, fail to identify unknown samples leading to a decrease in their performance, while our method ROS maintains consistent performance in all the openness levels.

### 4.2.4 Computational Cost

In Table 4.8 we show the computational cost of our multi-task approach and its best competitor OSBP [131] for the experiments in Table 4.4. Our analysis shows the total number of FLOPs (Floating Point Operations) required for a single forward pass of the network, the training, and the inference time (in milliseconds). ROS is a two-stage approach and, even if the architecture of the first and second steps is mostly preserved allowing a comparable number of FLOPs, it requires a significantly higher training time than the end-to-end OSBP. This could be a bottleneck in specific situations where it's needed to trade off accuracy for faster training.

### 4.2.5 Conclusions

In this section, we propose a new approach called ROS for addressing the problem of OSDA. This method relies on the self-supervised task of predicting image rotation

Table 4.8 Cost analysis on Office-31 with ResNet-50 in OSDA setting. Hardware - CPU: Intel(R) Core(TM) i7-9800X CPU @ 3.80GHz, GPU (x1): Nvidia TITAN Xp.

| | **Cost analysis** | | | |
|---|---|---|---|---|
| | FLOPs | Training Time (ms) | Inference Time (ms) | HOS (Avg.) |
| ROS | $4.16 \times 10^9$ | $2.78 \times 10^7$ | 3.23 | **85.90** $\pm$ 0.2 |
| OSBP [131] | $4.14 \times 10^9$ | $6.73 \times 10^6$ | 3.27 | 83.70 $\pm$ 0.4 |

and demonstrates that by making small changes to this task, we can effectively distinguish between known and unknown target samples also aligning the target samples predicted as known with the source samples. Moreover, we introduce HOS: a new metric for evaluating OSDA that takes into account the accuracy of recognizing known classes and the ability to identify unknown samples. HOS addresses the limitations of the current metric, OS, which becomes less effective as the number of known classes increases. In this section, we test the performance of ROS, on two commonly used benchmarks Office-31 and Office-Home demonstrating that ROS outperforms existing approaches. Furthermore, when evaluated in settings with increasing openness, ROS is the only method that demonstrates consistent performance. HOS reveals to be crucial in this evaluation to correctly assess the performance of the methods on both known and unknown samples. Finally, the failure in reproducing the reported results of existing methods exposes an important issue in OSDA that echoes the current reproducibility crisis in machine learning. We hope that our contributions can help lay a more solid foundation for the field.

# Chapter 5

# Improved Supervised Models for Cross-Domain Learning

*In this Chapter, we explored the relation-based self-supervised approaches whose formulation allows for an easy extension to supervised learning. Indeed, differently from transformation-based self-supervised strategies for which resorting to multi-task was essential, in relation-based approaches it is possible to include supervision by simply exploiting data annotation while defining instance relations. Two patches are similar if they come from images belonging to the same class and different otherwise. In particular, with HyMOS, in Section 5.1 we propose a contrastive learning-based approach for Open-Set across domains. In Section 5.2 ReSeND, a relational reasoning-based approach for semantic novelty detection.*

## 5.1 HyMOS: Distance-based Hyperspherical Classification for Multi-source Open-Set Domain Adaptation



Standard CSDA [20] focuses on minimizing the difference between the labeled source data used for training and the unlabeled target data used for testing when the target data covers the same class set as the source data. OSDA [261], on the other hand, not only aims to bridge the domain gap but also to identify and reject samples of unknown classes. Naïvely applying adaptive solutions with category shift can lead to *negative transfer* and irreversible misalignment of the categories [24]. While dealing with multiple sources is a common occurrence in real-world conditions, only one recent work has begun to explore the task of multi-source OSDA [138]. This faces the challenge of learning a feature space that is common across domains while also maximizing the distinction between *known* and *unknown* categories within the unlabeled target. All current open-world adaptive learning models try to address the issue of the limited generalization ability of deep learning models. This problem can be attributed to two known limitations of CNNs: (1) deep models tend to generate features that primarily describe local statistics, which leads to a bias towards the style of the training data [165]; (2) the commonly used cross-entropy loss for supervised learning tends to produce overconfident predictions, which biases the model towards the labeled class set [41, 159].

Existing solutions use multi-stage learning procedures, incorporate multiple losses to compensate for the cross-entropy overreliance, and use adversarial techniques to close the domain gap. These approaches often require numerous hyperparameters to be tuned with manually defined thresholds, or including elaborated models to generate samples of a synthetic unknown source class (see Table 5.1). In this section, we propose a supervised model that overcomes the limitations of the cross-entropy loss and learns a style-invariant embedding space that naturally separates unknown categories. We build upon the recent trend of contrastive learning [40, 172, 41], where the encoder learns the invariance between two augmented

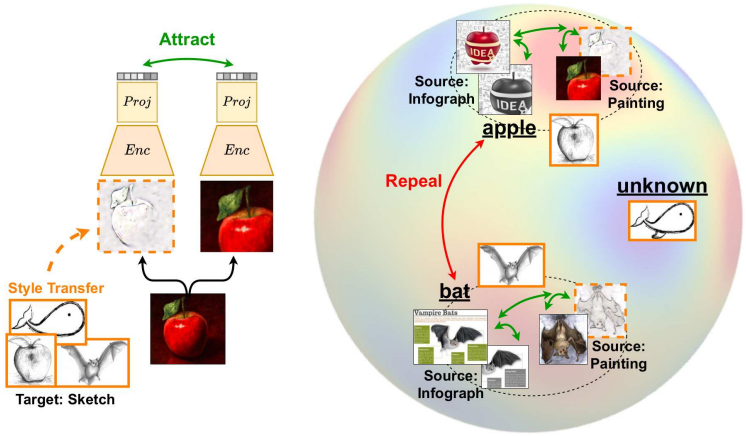



Fig. 5.1 HyMOS uses supervised contrastive learning to address all the challenges of multi-source open-set domain adaptation. We integrate style transfer into the double path contrastive logic to achieve domain-invariant representation. By balancing the class and source domains in each training batch, we achieve class-wise domain alignment. The embedding space learned by HyMOS inherently separates *unknown* target samples in low-density regions, while the *known* samples are located close to their corresponding class cluster and can be easily used in self-training for further adaptation.

versions of the same image (positive pair) while maximizing the distance between augmented versions of different instances (negative pair). We demonstrate that a single supervised contrastive learning objective can effectively address all the challenges of multi-source OSDA (see Figure 5.1):

- class-wise alignment between source domains is achieved by balancing data batches across classes and domains;

| Method | No. of Losses | No. of HPs | Threshold |
|---|---|---|---|
| Inheritable [134] | 4 | 2 | not used - synthesize *unknown* target |
| ROS [19] | 6 | 4 | reject a fixed portion of Target |
| CMU [262] | $2+\|C_s\|$ | 3 | validated |
| DANCE [263] | 3 | 3 | fixed value depending on $\|C_s\|$ |
| PGL [135] | 3 | 4 | reject a fixed portion of Target |
| MOSDANET [138] | $4+\|S\|$ | 2 | validated |
| **HyMOS** | 1 | 1 | self-paced, updates online while training |

Table 5.1 Comparison with existing open-set and universal domain adaptation approaches. HPs indicate the hyperparameters, $|C_s|$ the number of source categories, $|S|$ is the number of source domains. Note that synthesizing new samples is a time-consuming operation and any validation procedure requires at least a dedicated per-dataset tuning.

- adaptation to the target domain is obtained by creating domain-invariant features through the use of style transfer among the augmentations of contrastive learning. This is followed by a progressive and self-regulated self-training procedure that improves alignment between target and source class clusters;
- the separation between known and unknown target data is achieved through a self-paced threshold based on the distribution of data in the learned hyperspherical feature embedding. The contrastive objective results in compact and well-separated known class clusters [264], leaving unknown samples isolated in low-density regions

To highlight the important role of the **Hy**perspherical feature space for our **M**ulti-source **O**pen-**S**et approach, we named it **HyMOS**. Our experiments on three multi-source open-set datasets demonstrate the superiority of HyMOS compared to current state-of-the-art methods. Furthermore, we also show promising results when applying HyMOS to related challenging scenarios such as multi-source closed-set and multi-source universal DA settings. The obtained results show that HyMOS outperforms several competitors, defining the new state-of-the-art. Our code is available at https://github.com/silvia1993/HyMOS

### 5.1.1 Method

Our approach to addressing multi-source Open Set domain adaptation involves building a robust and highly structured feature space with class clusters compact, and well-separated class clusters, keeping *unknown* target samples away from the centers. We obtain this effect by minimizing the supervised contrastive loss and paying attention to how data are fed to the model. In particular: (a) we develop a strategy for creating mini-batches that balance the domains and classes, resulting in perfect class alignment among the sources; (b) we incorporate style transfer into the contrastive learning method to create sample pairs, leading to a domain-invariant feature embedding; (c) we gradually include the target domain into the learning objective through self-training improving source-target alignment; (d) we use a self-paced threshold, based on data distribution, to distinguish between known and unknown samples in the target domain. We apply this threshold during both the inference and the selection of known target samples for self-training. In the next section, we will delve into each of the points listed above more in-depth. A summary

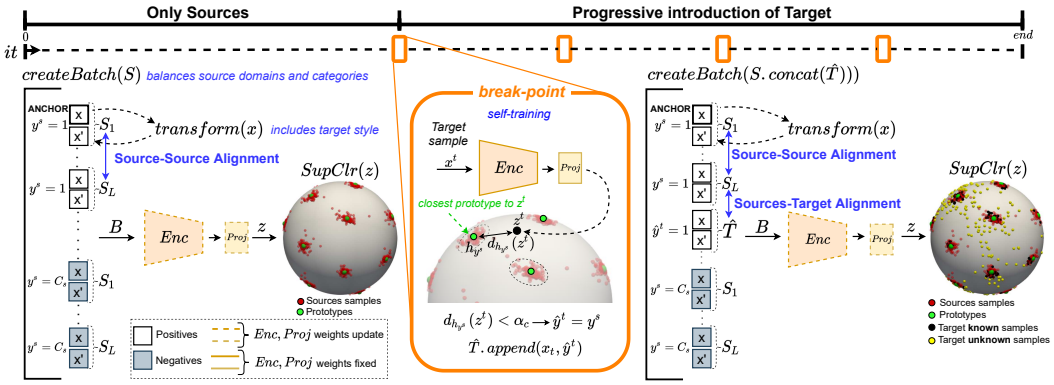



Fig. 5.2 Schematic illustration of HyMOS (best viewed in color). We use the same notation adopted in Algorithm 1, please refer to it to follow the flow of the method.

of our approach, HyMOS, is illustrated in Figure 5.2 and summarized in Algorithm 1 (see the Appendix for the evaluation procedure).

**Problem Framework.** In multi-source Open-Set domain adaptation we are given $L$ labeled source domains $S = \{S_1, S_2, \ldots, S_L\}$, where $S_i = \{x_j^{s_i}, y_j^{s_i}\}_{j=1}^{N^{s_i}} \sim p_i$, and one unlabeled target domain $T = \{x_j^t\}_{j=1}^{N^t} \sim q$, all drawn from different data distributions $p_{i=1,\ldots,L}, q$. The source domains have the same set of labels $y^s \in C_s$, and $C_s$ is a subset of the labels in $C_t$. This means that the target includes additional classes $C_{t \setminus s}$, referred to as *unknown*. Our objective is to train a model using the source data that can accurately classify each target sample by either assigning it to one of the known $C_s$ classes or identifying it as unknown. Given the different relatedness levels of the target with each of the available sources, reducing the domain shift while avoiding the risk of *negative transfer* may be difficult, especially when the openness $\mathbb{O} = 1 - \frac{|C_s|}{|C_t|}$ increases.

**Contrastive Learning Formulation.** In self-supervised contrastive learning [40, 172], two transformed views of every input image are propagated through a CNN network. These two versions of the image are created by applying standard augmentation techniques such as grayscale conversion, random cropping, and color jittering. For each sample $\{\mathbf{x}_k^s, y_k^s\}$ in the double batch $B = \{k = 1, \ldots, 2K\}$, the encoder network is applied to extract the features $Enc(\mathbf{x}_k^s)$. These features are then passed through a final contrastive head, which maps them to a normalized embedding space $\mathbf{z}_k^s = Proj(Enc(\mathbf{x}_k^s))$. In this space, the similarity between two augmented versions of the same instance is maximized, while the similarity between different instances is minimized. When the labels of the images are available, the comparison of samples can be done both on an individual basis, as in the self-supervised

case, and by category, [41]. Samples of the same class $y_k^s$ are considered positive, while samples from different classes are considered as negative pairs. The set of positive pairs for sample of index $k$ is defined as $\pi(k) = \{k' \in \nu(k) : y_{k'}^s = y_k^s\}$ where $\nu(k) = B \setminus k$ is the double batch without the anchor sample of index $k$.

Thus, the supervised contrastive loss is [41]:

$$L_{SupClr} = \sum_{k=1}^{2K} \frac{-1}{|\pi(k)|} \sum_{k' \in \pi(k)} \log \frac{\exp(\sigma(\mathbf{z}_k^s, \mathbf{z}_{k'}^s)/\tau)}{\sum_{n \in \nu(k)} \exp(\sigma(\mathbf{z}_k^s, \mathbf{z}_n^s)/\tau)} , \tag{5.1}$$

where $\tau \in \mathbb{R}^+$ is the temperature, and $\sigma(\cdot,\cdot)$ is the cosine similarity.

**HyMOS Source-Source Class-Wise Domain Alignment.** The supervised contrastive loss aims at learning compact class clusters with large margins. This ability can be exploited to perform source-source class-wise domain alignment by composing each training mini-batch with samples coming from different visual domains. We divide each batch evenly among all $|C_s|$ classes, and for each class, we choose the same number of samples from all $L$ source domains. The loss function in Eq. (5.1) ensures that samples of the same class are located in the same area regardless of their domain, while samples from different classes are separated from each other.

**HyMOS Source-Target Style Invariance.** Contrastive learning uses image transformations to help the model learn the core semantic information while being insensitive to irrelevant cues. When working with data from different domains, it is essential to have a representation that can disregard significant differences in visual appearance, beyond simple grayscale or color jittering. This requires the use of specific image transformations that preserve the semantics of the image. We propose here to use image augmentation based on style transfer changing the global texture of the image while preserving its content. Specifically, we use the AdaIN model [265] that is trained on both source and target data to transfer the style of the target domain to source images. Since this augmentation is applied randomly, the loss function compares the original source images with target-like images, and the model learns to disregard the style difference. It's worth noting that by doing that we obtain a style-invariant features space avoiding the risk of negative transfer which is a significant challenge in open-set domain adaptation. Previous methods such as [19, 262, 263, 258, 24] attempt to address this problem by either excluding unknown samples or reducing their impact during the adaptation process. This requires implementing complex strategies to identify unknown samples before learning the

domain-invariant model. By using style transfer, we instead create a domain-agnostic representation by disregarding the semantic content of the target. This allows us to extract the style from samples belonging to unknown categories without incurring in negative transfer.

**HyMOS Adaptation Refinement via Self-Training.** Achieving perfect alignment between the source and target domains would be possible if we could include target data as an additional source domain during the training of the supervised contrastive model. Of course, this is not possible as the class labels for target samples are not available. However, once the model trained on source data including target style invariance is robust enough, it can be used to produce pseudo-labels for target data by simply using its predictions. Indeed, after the initial training on the source data, we gradually include target samples in the learning objective by going through evaluation stages that we refer to as *self-training breakpoints*. This enables us to select target samples that we are confident are *known*. Through this iterative process, we transfer label knowledge from the source to the target data, enhancing the compactness of the class clusters, and leaving *unknown* target data in sparse regions of the hyperspherical feature space.

**HyMOS Known-Unknown Separation and Classification on the Hypersphere.** The resulting embedding, with well-clustered known categories separated by large margins and unknown samples in isolated areas, is ideal for distance-based classification. Unlike previous approaches [41, 169] which use the contrastive models as a pre-training step and discards the projection head in favor of a traditional cross-entropy loss, we continue to use the hypersphere for making predictions. We compute the prototype of each source class $y^s$ by finding the average feature $h_{y^s} = \frac{1}{N_{y^s}} \sum_{k \in y^s} \mathbf{z}_k^s$ which is then re-projected into the unit hypersphere. For any target sample $\mathbf{z}^t$, we compute the cosine similarity with each source class prototype and scale it to be in the range $[0,1]$ defining the distance $d_{h_{y^s}}(\mathbf{z}^t) = \{1 - \sigma_{[0,1]}(\mathbf{z}^t, h_{y^s})\}$ for $y^s \in 1, \ldots, |C_s|$ which is used as a measure of confidence for label assignment. To determine if a sample belongs to a *known* category, we need to set a threshold for the distance from the *known* class prototypes. The issue of how to define this threshold is a widely debated topic in Open Set literature, with many methods choosing values a priori and keeping them fixed while training [262, 263]. We propose to derive the threshold directly from the observed data distribution, resulting in a value that is updated during the learning process. Specifically we introduce two metrics the *class sparsity*:

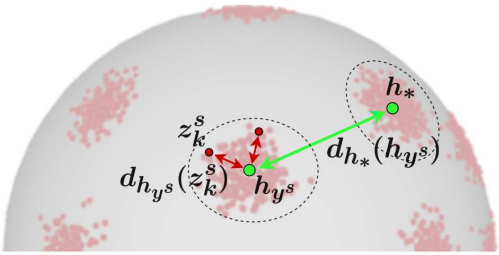



Fig. 5.3 Illustration of the distances used to set the class prediction and the self-training procedure.

$$\theta = \frac{1}{|C_s|} \sum_{y^s \in C_s} d_{h_*}(h_{y^s}), \tag{5.2}$$

where $h_*$ is the closest prototype to each $h_{y^s}$, and the *class compactness*:

$$\phi = \frac{1}{|C_s|} \sum_{y^s \in C_s} \left\{ \frac{1}{N_{y^s}} \sum_{k \in y^s} d_{h_{y^s}}(z_k^s) \right\}. \tag{5.3}$$

The first calculates the minimum distance between each class prototype and provides a measure of how distinct the classes are from each other. The second checks if the samples within each class are closely grouped around their corresponding prototype (see Figure 5.3). When there are many classes with little variation within each class, it results in a feature scenario with high compactness but low sparsity, for which a low threshold is needed. In contrast, when there are few classes with a lot of variation within each class, it results in a low compactness and high sparsity condition for which we can allow a higher threshold. We compute our threshold by:

$$\alpha = \phi \cdot \left[ log\left(\frac{\theta}{2\phi}\right) + 1 \right], \tag{5.4}$$

where $\theta/2\phi$ estimates the average ratio between the distance of two adjacent prototypes and the radii of the respective clusters. The use of the threshold during inference is straightforward:

$$\hat{y}^t = \begin{cases} \text{argmin}_{y^s}(d_{h_{y^s}}(\mathbf{z}^t)) & \text{if } \min_{y^s}(d_{h_{y^s}}(z^t)) < \alpha \\ \texttt{unknown} & \text{if } \min_{y^s}(d_{h_{y^s}}(\mathbf{z}^t)) \geq \alpha\,. \end{cases} \tag{5.5}$$

We use the same threshold for the self-training break-points mentioned earlier, however, we are more conservative in this stage. So, we have an additional factor $\alpha_m$ that enables us to maintain a more cautious threshold: $\alpha_c = \alpha_m \cdot \alpha$. The multiplier is fixed at 0.5 and it's the only adjustable parameter in HyMOS.

### 5.1.2 Experiments

We implemented HyMOS with a ResNet-50 [219] backbone and two fully connected layers for the contrastive head. All the technical details can be found in the Appendix.

**Results.** We compare HyMOS with several state-of-the-art baselines proposed for single-source Open-Set (Inheritable [134], ROS [19], PGL [135]), multi-source Open-Set (MOSDANET [138]) and universal domain adaptation (CMU [262], DANCE [263]). For all the approaches not designed to handle multiple sources, we employ the *Source Combine* strategy [1], which involves combining all source data into a single domain. We considered the $HOS$ metric, defined in [19, 262] to fairly evaluate the Open-Set approaches: it is the harmonic mean between the average class accuracy over the known classes $OS^*$ and the accuracy over the unknown class $UNK$: $HOS = 2\frac{\text{OS}^* \times \text{UNK}}{\text{OS}^* + \text{UNK}}$. Table 5.2 presents the results of our study, which demonstrate that HyMOS outperforms all the competitors. The margin of improvement of HyMOS over its closest competitor, ROS, ranges from 1.9% on OfficeHome to 10.8% on DomainNet. Besides being simpler than the other methods, HyMOS also demonstrates robustness across the various scenarios represented by the three datasets, including the number of shared and private classes and the nature and extent of domain gaps. These unique qualities make HyMOS the most suitable approach for a wide range of real-world applications. We also analyze HyMOS considering the *AUROC* (Area under the Receiver Operating Characteristic curve) which is not affected by a threshold. In HyMOS, the *normality score* used to determine if a sample is *known* or *unknown* is based on its distance from the nearest source class prototype, while ROS, its best competitor, uses a combination of entropy and probability output from an auxiliary rotation recognition classifier. Even in this case, HyMOS outperforms ROS, which is consistent with the results obtained with HOS. This confirms that the *known* and *unknown* samples are well separated in the learned hyperspherical embedding space.

| | Method | | DomainNet | | | Office31 | | | | Office-Home | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | → S | → C | **Avg.** | → W | → D | → A | **Avg.** | → Rw | → Cl | → Ar | → Pr | **Avg.** |
| HOS | Source Combine | Inheritable [134] | 34.8 | 44.0 | 39.4 | 76.6 | 79.5 | 70.0 | 75.4 | 63.2 | 52.6 | 48.7 | 60.7 | 56.3 |
| | | ROS [19] | 44.5 | 52.4 | 48.5 | 81.8 | 80.1 | 64.7 | 75.5 | **73.0** | 57.3 | 61.6 | 69.1 | 65.3 |
| | | CMU [262] | 38.1 | 35.5 | 36.8 | 61.4 | 64.0 | 56.4 | 60.6 | 70.8 | 50.0 | 58.1 | 69.3 | 62.1 |
| | | DANCE [263] | 30.0 | 37.6 | 33.8 | 38.5 | 59.7 | 58.0 | 52.0 | 12.4 | 16.1 | 18.6 | 22.9 | 17.5 |
| | | PGL [135] | 18.5 | 19.4 | 19.0 | 43.3 | 37.7 | 35.6 | 38.9 | 40.0 | 31.5 | 31.8 | 42.2 | 36.4 |
| | Multi-Source | MOSDANET [138] | 40.0 | 39.3 | 39.6 | 60.5 | 71.5 | **73.9** | 68.6 | 65.0 | 51.1 | 54.3 | 65.9 | 59.1 |
| | | **HyMOS** | **57.5** | **61.0** | **59.3** | **90.2** | **89.9** | 60.8 | **80.3** | 71.0 | **64.6** | **62.2** | **71.1** | **67.2** |
| AUROC | Source Combine | ROS [19] | 63.9 | 68.0 | 66.0 | 93.9 | 95.2 | **73.5** | 87.5 | 80.8 | 69.6 | 73.7 | 79.4 | 75.9 |
| | Multi-Source | **HyMOS** | **71.9** | **75.8** | **73.9** | **96.9** | **96.1** | 71.0 | **88.0** | **81.1** | **76.4** | **75.3** | **79.6** | **78.1** |

Table 5.2 Results averaged over three runs for each method on the DomainNet, Office31, and Office-Home datasets.

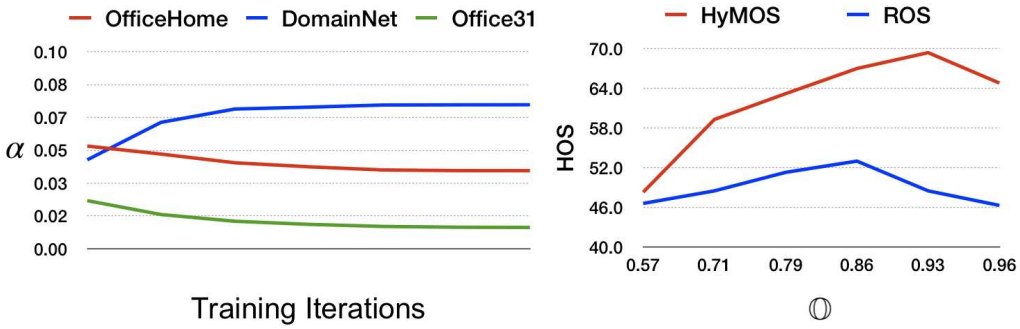



Fig. 5.4 Left: analysis on the dynamic threshold $\alpha$ at different training iterations. Right: performance of HyMOS and ROS [19] at different openness ($\mathbb{O}$) levels.

**Analysis on the Threshold.** HyMOS is designed with a self-paced procedure that adapts the dynamic threshold $\alpha$ to the data distribution. Figure 5.4 (left) shows the change in $\alpha$ during training. For Office31 and Office-Home datasets, the threshold gradually decreases over time, while for DomainNet it increases. The changes in $\alpha$ demonstrate how the data clusters change over time. As the training progresses, the clusters become more compact, and the distance between them increases, resulting in a more sparse distribution of classes on the hypersphere. For DomainNet, the different trend is associated with the class cardinality, which is higher compared to the other datasets. In all cases, the threshold ultimately reaches a stable value. HyMOS has only one adjustable parameter, the $\alpha_m$ multiplier, which is used to compute a more conservative threshold during training. By varying this parameter, one can choose to prioritize recognition of known classes over unknown classes. The results in Table 5.3 indicate that a value of 0.5 for $\alpha_m$ is a safe choice for all datasets. Additionally, by adjusting this multiplier, the HOS performance of HyMOS remains competitive with ROS, and in some cases, such as DomainNet with $\alpha_m$=1, it can even improve.

| Method | | **DomainNet** | **Office31** | **Office-Home** |
|---|---|---|---|---|
| HyMOS | $\alpha_m = 0.3$ | 55.1 | 79.2 | 65.8 |
| | $\alpha_m = 0.5$ | 59.3 | **80.3** | **67.2** |
| | $\alpha_m = 0.7$ | 60.8 | 78.2 | 66.8 |
| | $\alpha_m = 1.0$ | **61.4** | 74.1 | 65.8 |
| ROS [19] | | 48.5 | 75.7 | 65.3 |

Table 5.3 Average performance (HOS) when changing the train-time multiplier $\alpha_m$ to the self-paced threshold $\alpha$.

| Method | **Office-Home** | | | | |
|---|---|---|---|---|---|
| | $\rightarrow$ Rw | $\rightarrow$ Cl | $\rightarrow$ Ar | $\rightarrow$ Pr | **Avg.** |
| **HyMOS** | 71.0 | 64.6 | 62.2 | 71.1 | **67.2** |
| w/o Source Balance | 69.2 | 58.4 | 60.6 | 70.2 | 64.6 |
| Style Tr. Target Known (Oracle) | 70.7 | 63.7 | 62.5 | 71.2 | 67.0 |
| w/o Style Transfer | 69.5 | 56.4 | 60.0 | 68.3 | 63.6 |
| w/o Self-Training | 72.2 | 55.0 | 58.6 | 71.5 | 64.3 |
| Improved Cross-Entropy | 61.5 | 61.2 | 58.1 | 57.1 | 59.5 |
| ROS [19] | 73.0 | 57.3 | 61.6 | 69.1 | 65.3 |
| + Source Balance | 75.2 | 55.5 | 62.6 | 66.9 | 65.0 |
| + Style Transfer | 62.6 | 46.3 | 52.0 | 60.1 | 55.2 |
| + Self-Training | 69.6 | 59.1 | 61.5 | 60.5 | 62.7 |
| + S. Balance, Style Tr., Self-Train. | 62.0 | 40.4 | 52.2 | 62.4 | 54.3 |

Table 5.4 Ablation Study, HOS results.

**Increasing the Openness Level.** In realistic situations, it is challenging to have direct control over the number of *unknown* classes in the unlabeled target data, and it is common to have more unknown categories than known ones. To evaluate how HyMOS performs with a different level of openness we consider the DomainNet dataset, which has a large number of classes. The plot in Figure 5.4 (right) shows that HyMOS surpasses its closest competitor ROS for the openness levels in range $\mathbb{O} \in 0.5, 1$.

### 5.1.3 Ablation Analysis

We designed HyMOS to be simple while taking into account all the challenges of multi-source Open-Set domain adaptation. In the following, we will provide a thorough analysis of each challenge, by providing a detailed ablation that sheds light on the inner functioning of our method. All the results are reported in Table 5.4.

| Multi-Source Closed-Set | | | | | | | | Multi-Source Universal | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Method | → clp | → inf | → pnt | → qdr | → rel | → skt | **Avg.** | Method | → S | → C | **Avg.** |
| Source Only [266] | 52.1 | 23.4 | 47.7 | 13.0 | 60.7 | 46.5 | 40.6 | CMU [262] | 38.9 | 31.2 | 35.1 |
| LtC-MSDA [267] | 63.1 | 28.7 | 56.1 | 16.3 | 66.1 | 53.8 | 47.4 | DANCE [263] | 44.5 | 49.9 | 47.2 |
| DRT [266] | 71.0 | 31.6 | **61.0** | 12.3 | 71.4 | 60.7 | 51.3 | ROS [19] | 39.7 | 46.0 | 42.9 |
| **HyMOS** | **71.5** | **41.8** | 60.8 | **34.5** | **74.2** | **66.6** | **58.2** | **HyMOS** | **54.6** | **57.1** | **55.9** |

Table 5.5 Multi-Source Closed-Set (Accuracy) and Universal Domain Adaptation (HOS) performance on DomainNet.

**Source-Source Alignment.** Aligning the source domains improves the model's ability to generalize. This idea is widely discussed in the literature on multi-source Closed-Set domain adaptation [94, 98]. The only existing multi-source Open-Set method, MOSDANET, also includes a specific component for aligning the sources. HyMOS achieves cross-source adaptation by combining a supervised contrastive learning loss with a carefully crafted batch sampling strategy: each training mini-batch includes one sample per class and per domain. The supervised contrastive loss creates a robust alignment between classes by bringing together samples of the same class and separating samples of different classes, regardless of the domain. Our results show that this balancing strategy increases the performance of HyMOS by 2.6% compared to the version without source balancing (as shown in the row *w/o Source Balance*).

**Source-Target Adaptation.** HyMOS uses both style transfer augmentation and an adaptive self-training process to bring the source and target domains closer without the risk of negative transfer. By adding target style transfer as one of the source augmentations, the model is trained to identify visual characteristics that are not specific to the target domain. We compare the performance of HyMOS extracting the target style only from the known categories (*Style Tr. Target Known (Oracle)*). Our results indicate that HyMOS is not negatively affected by using the entire target for this adaptation step. Additionally, we also evaluate the performance of HyMOS without style transfer (*w/o Style Transfer*), which results in a 3.6% drop in performance, demonstrating its crucial role of style in HyMOS. Finally, the self-training process is responsible for creating a robust alignment between source and target features, by selecting samples from the target that are confidently identified as known (those closest to the source class prototypes) and incorporating them into the learning objective. The performance of HyMOS improves by 2.9% when compared to its version without this strategy (*w/o Self-Training*).

**Comparison with an Improved Cross-Entropy Baseline.** Source balance, style transfer, and self-training are relatively straightforward techniques that can be used

with any supervised learning model to enhance its performance in multi-source Open-Set scenarios. However, we believe that the use of supervised contrastive learning and its associated hyperspherical embedding is essential for this task. To demonstrate this point, we replaced the contrastive loss in HyMOS with the standard cross-entropy loss. The results of this experiment (*Improved cross-entropy*) show that this simple baseline approach performs significantly worse than HyMOS.

**Comparison with an improved version of ROS [19].** We also enhanced our closest competitor, ROS, by incorporating the same techniques used in HyMOS: source balancing, style transfer, and self-training. The results in the lower portion of Table 5.4 show that organizing the training data to include equal amounts of categories and source domains from different datasets in each mini-batch (+ *Source Balance*) does not lead to an improvement in the performance of the standard version of ROS. This is because, unlike contrastive learning, the cross-entropy loss does not inherently promote clustering and adaptation between sources. When ROS is augmented with style transfer (+ *Style Transfer*) it performs poorly: we observed a slight improvement in the recognition accuracy of known classes, but a significant decline in the accuracy of unknown class, resulting in a decrease in the overall performance. We also tried to extend ROS with self-training (+ *Self-Training*), following the approach used in [138] but this also leads to a drop in performance: the self-training process tends to propagate recognition errors caused by the overconfidence due to the use of cross-entropy loss. While self-training can introduce dangerous model drift, recent research has shown that it can be effective and safe when sample selection is performed using a self-paced strategy based on the distribution of unlabeled samples, similar to the approach used in HyMOS [268]. Finally, when all the strategies are used together, the results are comparable to those obtained when using style transfer alone. This last technique clearly steered the whole method towards a low performance.

### 5.1.4 Extension to Closed-Set and Universal

HyMOS is versatile and can be applied to various domain adaptation settings including the simpler multi-source Closed-Set scenario where there is a complete overlap between the source and target classes and the more complex multi-source Universal case where both the sources and target can have their own unique classes. To evaluate these settings, we use the DomainNet dataset following the experimental protocols outlined in previous literature, such as [266] for Closed-Set and [262] for

Universal. In the latter case, we used the first 150 classes in alphabetical order as shared classes between sources and target, the next 50 categories as unique classes of the sources, and the remaining classes as the unknown categories in the target. For the Closed-Set scenario, we compare with LtC-MSDA [267] and DRT [266] which leverage respectively on a graph connecting domain prototypes, and on a dynamic transfer that updates the model parameters on a per-sample basis. The results of these methods are summarized in Table 5.5 and show that HyMOS performs very well compared to several state-of-the-art methods also in these two scenarios.

### 5.1.5 Computational Cost

In Table 5.6 we show the computational cost of our proposed approach and its best competitor ROS [19] for the experiments in Table 5.2. Our analysis shows the total number of FLOPs (Floating Point Operations) required for a single forward pass of the network, the training, and the inference time (in milliseconds). HyMOS shows a significantly higher number of FLOPs and training time than ROS since it uses two different networks: one for the main contrastive learning task and one to produce target-like source images. However, HyMOS has the best performance with the same inference time as ROS so it could be a preferable choice when priority is given to high accuracy even at the cost of longer training.

Table 5.6 Cost analysis on Office-31 with ResNet-50 in MOSDA setting. Hardware - CPU: Intel(R) Core(TM) i7-9800X CPU @ 3.80GHz, GPU (x1): Nvidia TITAN Xp.

| | **Cost analysis** | | | |
|---|---|---|---|---|
| | FLOPs | Training Time (ms) | Inference Time (ms) | HOS (Avg.) |
| HyMOS | $7.68 \times 10^{10}$ | $8.89 \times 10^{7}$ | 3.31 | **80.30** |
| ROS [19] | $4.19 \times 10^{9}$ | $4.53 \times 10^{7}$ | 3.51 | 75.50 |

### 5.1.6 Conclusions

In this section, we presented HyMOS, a novel approach for multi-source Open-Set domain adaptation. By utilizing supervised contrastive learning and the inherent properties of the hyperspherical feature space, it overcomes the limitations of current approaches. HyMOS includes a tailored data balancing to enforce cross-source alignment and introduces style transfer among the instance transformations for

source-target adaptation, keeping away from the risk of negative transfer. Finally, a self-training strategy refines the model without the need for manually set thresholds Through extensive experimentation, we demonstrate that HyMOS outperforms current state-of-the-art methods on three benchmarks providing a detailed analysis of its inner workings. Additionally, by testing the approach in the multi-source closed-set and universal scenarios, we confirmed its effectiveness proposing a valuable tool for lifelong learning in real-world applications.

## 5.2 ReSeND: Semantic Novelty Detection via Relational Reasoning

*Reproduced with permission from Springer Nature: Borlino, F. C., Bucci, S., & Tommasi, T. Semantic Novelty Detection via Relational Reasoning. European Conference on Computer Vision (pp. 183-200). Springer, Cham (ECCV 2022)*

In safety-critical fields like autonomous driving and healthcare, the ability to detect previously unseen categories as *unknown* is crucial for reliable performance. Researchers have proposed various approaches to improve deep learning models' ability to handle novel categories: by calibrating the softmax output of the classifier [145, 144, 149], by using generative approaches to synthesize outlier examples or by incorporating energy-based models to the learning process [157, 158, 154–156]. Despite the effectiveness of these techniques, they have a notable limitation: they all require a reasonably large set of reference data to learn what is *known*. When data access is limited due to privacy concerns or when dealing with computational and memory constraints (e.g. edge computing), these strategies may not be feasible. In this section, we focus on the pre-training stage: instead of using traditional cross-entropy-based approaches [219] or self-supervised contrastive learning strategies [40, 172], we can use ImageNet [214] to optimize a relational reasoning objective to obtain a more reliable embedding for novelty detection (see Fig. 5.5). Our goal is to achieve a semantic similarity measure that can accurately determine whether two samples belong to the same class or different ones. As a result, we focus on building a representation that is specifically designed for semantic comparison, which doesn't require additional fine-tuning steps on the task-specific annotated data. This representation will be able to differentiate known and unknown categories by comparing each test sample to the reference class prototypes. Not only is our method efficient, but it also provides a simple solution for converting closed-set models to open-set ones by adding a plug-and-play rejection option for unknown classes. This allows for easy integration and implementation in various applications.

In summary, our focus is on Semantic Novelty Detection (SeND) and we propose ReSeND, a representation learning approach, based on Relational Reasoning, that can be directly applied to real-world applications without the need for fine-tuning. In particular, our contributions are:

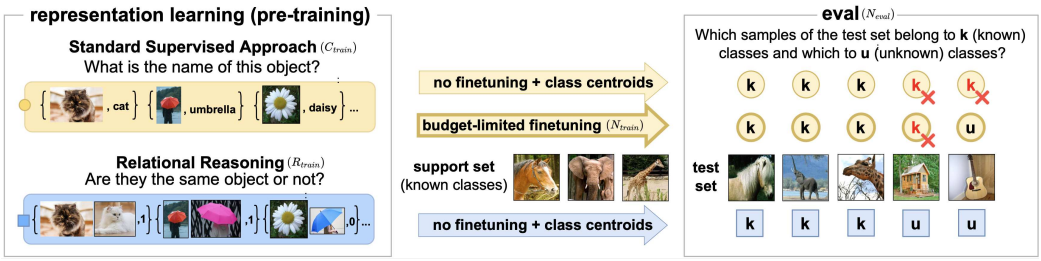



Fig. 5.5 Comparison between standard supervised learning and relational reasoning representation learning. Standard supervised learning focuses on recognizing known object classes, while relational reasoning representation learning aims to learn a measure of semantic similarity among image pairs. We show that relational reasoning is particularly well-suited for semantic novelty detection tasks. Our pre-trained large-scale relational model can be applied to these tasks without the need for a fine-tuning phase on the known classes specific to the task.

- we conduct a comprehensive experimental analysis on the capability of various representation learning approaches to handle the task of SeND, examining their potential and limitations;
- we propose ReSeND and evaluate it in multiple scenarios, including both *intra-* and *cross*-domain settings varying ratios of unknown classes in the test data. Our results, obtained through a comprehensive benchmark with various competitors, demonstrate the effectiveness and efficiency of our approach;
- we demonstrate how ReSeND can be used as a plug-and-play module in closed-set domain generalization approaches, effectively converting them into open-set domain generalization strategies and setting a new state-of-the-art.

### 5.2.1 Method

**Notation and background.** In Semantic Novelty Detection, we have two datasets: a *Support Set* containing labeled samples $S = \{\mathbf{x}^s, y^s\}_{k=1}^{K}$ drawn from the distribution $p_S$, and a *Test Set* containing unlabeled samples $T = \{\mathbf{x}^t\}_{h=1}^{H}$ drawn from the distribution $p_T$. The main difference between $p_S$ and $p_T$ is a semantic shift, with $y^s \in Y_s$, $y^t \in Y_t$ and $Y_s \neq Y_t$. The two sets of classes can be completely disjoint $Y_s \cap Y_t = \emptyset$, or partially overlapping $Y_s \subset Y_t$. In the following, we will refer to $Y_s$ as the *known* classes, and use the term *unknown* to refer to the test classes $Y_{t \setminus s}$ that are not present in the support set. The distribution difference between support and test

sets can be also due to variations in the appearance of the samples, but the class content remains unchanged. A robust semantic novelty detector should be able to distinguish between known and unknown samples in the test set, despite the domain shift. Given a test sample $\mathbf{x}^t$, the detector $D$ should be able to output a *score* in the range of [0,1] that indicates whether the sample is known (high score) or unknown (low score). Following the traditional approach, the detector can be formulated as $D : \{C_{train}(I), N_{train}(S), N_{eval}(\mathbf{x}^t)\}$. In this process, at first, a robust representation is obtained by training a classification model $C$ on a large-scale dataset such as ImageNet1k [214] using the samples $(\mathbf{x}_i, y_i)_{i=1}^{I}$. This representation is then used by the model $N$, which is further fine-tuned with the support set to gather the definition of *normality* from the data. The training objective is typically a simple classification task, and the final evaluation of $N$ on the test set is done using a Maximum Softmax Probability (MSP) approach, where $score = \max_{c \in Y_s} p(y = c|\mathbf{x}^t)$. We want to highlight that the fine-tuning process requires a computational cost that may not be feasible on edge devices. Moreover, in the long term, its catastrophic forgetting effect reduces the original large-scale knowledge, as well as the ability to anticipate potential semantic anomalies [269]. Thus, carefully designing the representation learning approach and choosing how the pre-trained model should be applied for the downstream task is crucial. We propose to change the learning paradigm for the semantic novelty detector so that it can be written as $D : \{R_{train}(I), N_{eval}(S, \mathbf{x}^t)\}$. The first component, $R$, is a representation learning model that uses relational reasoning and is trained on the ImageNet1k dataset. The embedding obtained from this model is then directly used by the evaluation system to compare each test sample with the support set, determining its normality score.

**Representation Learning via Relational Reasoning.** In our proposed approach $R$ is made of a feature extractor $f_\theta$ and a relational module $r_\gamma$. When a pair of samples $(\mathbf{x}_i, \mathbf{x}_j)$ from the dataset $I$ is fed through the model, it first goes through the feature extractor, obtaining the features $(\mathbf{z}_i = f_\theta(\mathbf{x}_i), \mathbf{z}_j = f_\theta(\mathbf{x}_j))$ then passed to the relational module $r_\gamma$. The output of this module serves as the input for the semantic similarity head $c_\delta$, which is a fully connected layer. It returns $\sigma_{ij} = c_\delta(r_\gamma(\mathbf{z}_i, \mathbf{z}_j)) \in [0, 1]$, representing a semantic similarity measure and can be interpreted as the likelihood of the two input samples belonging to the same category. The entire representation learning model is trained with a regression objective. Specifically, for each data pair, we assign the label $l_{ij} = 0$ if $y_i \neq y_j$ and $l_{ij} = 1$ otherwise, and the

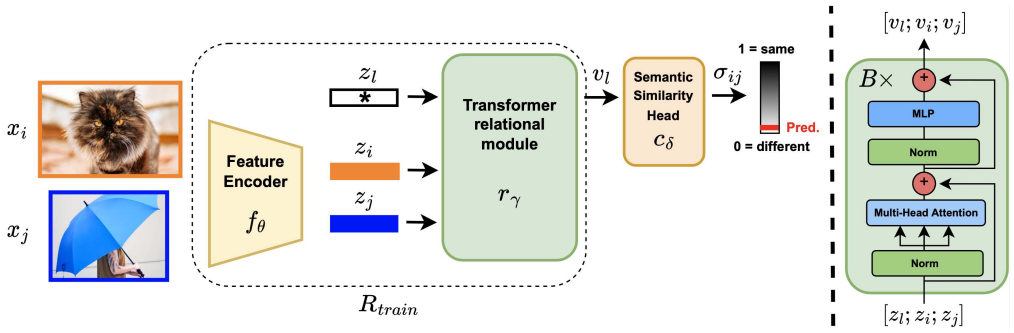



Fig. 5.6 Schematic illustration of the training phase of ReSeND. The features extracted from a pair of images are provided as input to our relational module. It consists of a transformer encoder that elaborates over a tuple composed of the sample pair and of a learnable label token. The output corresponding to this last token is finally provided as input to a semantic similarity head that predicts the sample resemblance.

mean squared error (MSE) loss is minimized:

$$\underset{\theta,\gamma,\delta}{\operatorname{argmin}} \sum_{m=1}^{M} (\sigma_m - l_m)^2 \,, \tag{5.6}$$

where the index $m$ specifies the pairs $(\mathbf{x}_i, \mathbf{x}_j)$ with $i \neq j$ and $x_i, x_j \in I$.

Despite the ground truth supervision being only at the extremes of the prediction interval, our goal is to learn a semantic similarity measure in the continuous range of $[0,1]$. Given that, the regression loss is well-suited for the task, but the problem could also be cast as binary classification. In the experimental section, we compare these two approaches and provide empirical evidence of the benefits of using regression.

**Evaluation Process.** Starting from the learned embedding, the component $N_{eval}$ has a straightforward role of comparing each test sample with the reference support set, without any additional training phase. $N_{eval}$ uses the relational module and inputs data pairs made by the features of each test sample $\mathbf{z}^t = f_\theta(\mathbf{x}^t)$, and the set of per-class prototypes $\overline{\mathbf{z}}^s_{y^s} \; \forall \, y^s \in Y_s$ which are obtained as the average of the samples of each class in the support set. We compute a vector of similarities between the test sample and each known class prototype by passing the pair $(\mathbf{z}^t, \overline{\mathbf{z}}^s_{y^s})$ through the relational module and the semantic similarity head. The resulting vector $\mathbf{u}$ has $|Y_s|$ elements, each representing the similarity of the test sample to a known class. This vector is then passed through a softmax function to normalize the similarities and the maximum value (MSP) is selected as the final normality score: $score = \max(softmax(\mathbf{u}))$.

**Relational module.** With respect to other standard components of deep neural networks that process single samples, the peculiarity of the relational module is that

it receives pairs of inputs and gives information about their similarity. Additionally, the relational module should be symmetric: the order of the input samples should not affect the output of the network. Given its inherent permutation-invariance and its ability to compare multiple inputs, we choose to implement our relational module using a simple transformer encoder. It comprises $B$ identical blocks, each consisting of Multi-Head Self-Attention (MSA) and Multi-Layer Perceptron (MLP), both preceded by Layer-Norm (LN) modules and connected by residual skip connections, as illustrated in the right part of Fig. 5.6. In the architecture of the relational module, the input feature vectors pair is combined with a learnable label token to make the tuple $[\mathbf{z}_l, \mathbf{z}_i, \mathbf{z}_j]$, which is then fed into the transformer encoder. It processes this input and produces the output sequence $[\mathbf{v}_l, \mathbf{v}_i, \mathbf{v}_j]$. It's important to note that in this architecture, each image is considered as a single input token to the transformer, similar to the approach used in previous works like [270]. To maintain permutation invariance, we do not include positional embeddings in our transformer encoder. In our implementation, we use a ResNet18-based backbone for the feature extractor $f_\theta$ selecting $\mathbf{v}_l$ as the output of the relational module, then passed through the similarity head $c_\delta$ producing a semantic similarity score $\sigma_{ij}$. Alternative architectures for the relational module are also evaluated in the experimental section.

### 5.2.2 Experimental Setup

ReSeND is a new approach for OOD fully based on representation learning. We claim that the embedding space learned through relational reasoning is particularly suitable for detecting novel classes by comparing test samples with the support set, which represents the reference *normal* condition. Since this logic substantially differs from that of previous works in OOD, there are several questions that we need to answer with experimental validations

*Are existing representation learning approaches effective for the SeND task?* (see Sec. 5.2.3) Our research focuses on the data representation obtained through a model pre-trained on ImageNet1k. We consider several advanced learning methods applying the same prototype-based evaluation approach used in ReSeND. The method involves identifying each class prototype of the support set by computing the average of its sample's feature representation. Additionally, we determine the normality score for each test instance by measuring its similarity to the closest known class centroid.

Our study analyzes two families of approaches. Within the cross-entropy-based classifiers, we consider this loss applied to ResNet [219] and ViT [146] architectures, as well as the data augmentation-based approach CutMix [166]. The second group of approaches is made by contrastive learning-based techniques, specifically, we consider the self-supervised approaches SimCLR [40] and CSI [169], as well as their supervised counterparts SupCLR [41] and SupCSI [169]. To determine the relationship between each test sample and the class prototypes, we use two different similarity measures: the Euclidean similarity (which is the inverse of the Euclidean distance [271]) and the cosine similarity. These measures are applied, respectively, to the cross-entropy and contrastive learning-based approaches. It is worth noting that these methods have been previously applied for anomaly and novelty detection [167, 272, 273]. However, in these previous works, a training phase on the support set was always necessary. In contrast, our approach only uses ImageNet1k to obtain the learned representation. It's important to note that the various approaches used in our study have different learning objectives but all of them use ResNet101 [219] architecture as their backbone. The ResNet101 has 44 million learnable parameters, which is comparable to ReSeND's 40 million (11 million for $f_\theta$, 29 million for $r_\gamma + c_\delta$). The only exception is made by ViT, which is an example of a Vision Transformer suggested for OOD in [272]. In that case, we use the Vit-Base (86 million parameters) implementation from [274].

*Is the learned embedding robust to domain variations?* (See Sec. 5.2.3) ImageNet1k is made by images of real-world objects and it's crucial to examine if the relationships encoded in the learned embedding are still applicable when the ultimate goal is to recognize new classes in vastly different contexts, such as texture images or sketches. We evaluate the performance of our method under two levels of difficulty. The first level is made by a domain shift between pre-training and the downstream task, which means that the support and test sets come from a different domain than that of ImageNet1k. The second level of difficulty is characterized by a domain shift between the support and test sets. The support set can be made up of data from one or multiple source domains, while the test set is from an even different target domain. We use multiple datasets to conduct an extensive analysis of the performance of our method.

*How does ReSeND compare with state-of-the-art OOD methods?* (See Sec. 5.2.3) Given that ReSeND does not require access to the support set during training, but

instead relies on it during evaluation, we can measure the amount of time and computational resources it uses in the evaluation stage and apply the same resources to the training phase of existing state-of-the-art OOD approaches. We evaluate ReSeND against the following baselines: MSP [145] which uses the standard maximum softmax probability, ODIN [144] a straightforward approach that uses input perturbation and temperature scaling, Energy [149] which uses an energy score for OOD uncertainty estimation, GradNorm [150] which is based on test-time extracted gradients to detect out-of-distribution samples, the ViT-based approach OODFormer [272] and two approaches that use tailored metric estimation: Mahalanobis [163] and Gram [164].

*Can ReSeND provide unknown detection abilities to closed-set approaches?* (See Sec. 5.2.3) ReSeND is a plug-and-play module that can be used to enhance existing close-set approaches, allowing them to work in open-set conditions without the need to train on the support set. We investigate the open-set domain generalization (Open-DG) scenario presented in [140] and demonstrate how ReSeND can improve the performance of existing methods. In addition to DAML, which was introduced in [140], we evaluate the state-of-the-art multi-source closed-set DG method SWAD [101], which aims to find flat minima in the learning objective function, as well as two single-source closed-set methods: SagNet [275], which separates shape and style in image features to decrease style bias, and Diversify [104], which generates images with unseen styles.

### 5.2.3 Experiments

In this section, we present and discuss the results of our experimental analysis. We evaluate the performance of ReSeND using two metrics: AUROC (Area Under the Receiver Operating Characteristic curve) obtained by varying the normality decision threshold, and FPR95 false positive rate of out-of-distribution examples when the true positive rate of in-distribution examples is at 95%. In addition, for the open-set DG experiments, following [140], we also consider the overall accuracy on known samples (Acc) and the harmonic mean between accuracy on known classes and unknown detection accuracy (H-score). Implementation[1] details and

[1]The code is available at https://github.com/FrancescoCappio/ReSeND

Table 5.7 Intra-Domain analysis. Best result in bold and second best underlined.

| Rep. Learning | Network | Texture | | Real | | Sketch | | Painting | |
|---|---|---|---|---|---|---|---|---|---|
| | | AUROC ↑ | FPR95 ↓ | AUROC ↑ | FPR95 ↓ | AUROC ↑ | FPR95 ↓ | AUROC ↑ | FPR95 ↓ |
| Cross Entropy | ResNet [219] | <u>0.678</u> | <u>0.892</u> | 0.710 | 0.860 | 0.553 | <u>0.936</u> | 0.651 | 0.926 |
| Cross Entropy | ViT [146] | 0.562 | 0.919 | 0.696 | <u>0.833</u> | <u>0.554</u> | 0.952 | <u>0.681</u> | <u>0.850</u> |
| CutMix [166] | ResNet | 0.619 | 0.922 | <u>0.721</u> | 0.877 | 0.542 | 0.943 | 0.629 | 0.927 |
| SimCLR [40] | ResNet | 0.529 | 0.942 | 0.481 | 0.944 | 0.502 | 0.956 | 0.510 | 0.956 |
| SupCLR [41] | ResNet | 0.534 | 0.947 | 0.561 | 0.899 | 0.532 | 0.946 | 0.532 | 0.933 |
| CSI [169] | ResNet | 0.651 | 0.906 | 0.663 | 0.887 | 0.514 | 0.955 | 0.621 | 0.910 |
| SupCSI [169] | ResNet | 0.652 | 0.903 | 0.695 | 0.875 | 0.535 | 0.953 | 0.652 | 0.909 |
| **ReSeND** | | **0.691** | **0.859** | **0.780** | **0.805** | **0.623** | **0.917** | **0.735** | **0.829** |

additional experimental analyses can be found in Appendices B.2.1 and B.2.2. All the experimental results are averaged over three runs.

**Intra-Domain analysis.** For the intra-domain analysis, we examine the scenario where the support and test sets come from the same visual distribution but have significant differences from ImageNet1k. Specifically, the testbeds were designed to not have any semantic overlaps with ImageNet1k, meaning that neither the known nor unknown classes appear in its label set. The texture benchmark, which was introduced in [15], and has already been used in [276], covers a completely different data type compared to ImageNet1k (objects vs textures). The Real, Sketch, and Painting domains are taken from the DomainNet dataset [1], and unlike Texture, they share the same data type (objects) as ImageNet1k but covering different visual domains. As shown in Table 5.7, ReSeND achieves the best results, demonstrating its ability to effectively transfer knowledge. On the Texture benchmark, the second and third best performing methods are Cross Entropy on ResNet and SupCSI respectively, however, this ranking is not consistent across all settings and the performance gap in comparison to ReSeND remains significant, particularly in the case of Sketch and Painting.

**Cross-Domain analysis.** In many real-world scenarios, it's not possible to avoid a visual domain shift between the training and test data, making the task more challenging. An efficient semantic novelty detection approach should be able to handle the domain shift between the support and test sets focusing only on the semantic content of the data. We evaluate ReSeND against the same baselines as the previous section considering two different benchmarks built from the PACS dataset [12]. In this experiment, the images used in the support set come from a single visual domain, while the test set is composed of images from a different domain.

Table 5.8 Cross-domain analysis. Top: single-source results, Bottom: multi-source results. We consider the PACS dataset with all the possible combinations of source/target as support/test sets. Best result in bold and second best underlined.

| Rep. Learning | Network | PACS **Single-Source** | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | ArtPainting | | Sketch | | Cartoon | | **Avg** | |
| | | AUROC ↑ | FPR95 ↓ | AUROC ↑ | FPR95 ↓ | AUROC ↑ | FPR95 ↓ | AUROC ↑ | FPR95 ↓ |
| Cross Entropy | ResNet [219] | 0.655 | 0.940 | 0.519 | 0.969 | 0.546 | 0.958 | 0.573 | 0.956 |
| Cross Entropy | ViT [146] | 0.593 | 0.895 | 0.595 | 0.881 | 0.500 | 0.953 | 0.562 | 0.910 |
| CutMix [166] | ResNet | 0.663 | 0.949 | 0.372 | 0.981 | 0.419 | 0.980 | 0.485 | 0.970 |
| SimCLR [40] | ResNet | 0.444 | 0.984 | **0.945** | **0.400** | 0.401 | 0.988 | 0.597 | **0.791** |
| SupCLR [41] | ResNet | 0.500 | 0.909 | 0.176 | 1.000 | 0.469 | 0.919 | 0.381 | 0.942 |
| CSI [169] | ResNet | 0.495 | 0.987 | 0.591 | 0.881 | 0.433 | 0.978 | 0.506 | 0.949 |
| SupCSI [169] | ResNet | 0.546 | 0.976 | 0.655 | 0.819 | 0.567 | 0.909 | 0.589 | 0.901 |
| **ReSeND** | | **0.828** | **0.668** | 0.576 | 0.981 | **0.651** | **0.891** | **0.685** | 0.847 |

| Rep. Learning | Network | PACS **Multi-Source** | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | ArtPainting | | Sketch | | Cartoon | | Photo | | **Avg** | |
| | | AUROC ↑ | FPR95 ↓ | AUROC ↑ | FPR95 ↓ | AUROC ↑ | FPR95 ↓ | AUROC ↑ | FPR95 ↓ | AUROC ↑ | FPR95 ↓ |
| Cross Entropy | ResNet [219] | 0.575 | 0.947 | 0.451 | 1.000 | 0.547 | 0.943 | 0.361 | 0.991 | 0.484 | 0.970 |
| Cross Entropy | ViT [146] | 0.611 | 0.837 | 0.566 | 0.944 | 0.539 | 0.904 | 0.932 | 0.403 | 0.662 | 0.772 |
| CutMix [166] | ResNet | 0.604 | 0.895 | 0.411 | 1.000 | 0.407 | 0.975 | 0.655 | 0.942 | 0.519 | 0.953 |
| SimCLR [40] | ResNet | 0.461 | 0.953 | **0.933** | **0.663** | 0.368 | 0.995 | 0.739 | 0.854 | 0.625 | 0.866 |
| SupCLR [41] | ResNet | 0.581 | 0.898 | 0.100 | 1.000 | 0.499 | 0.909 | 0.467 | 0.995 | 0.412 | 0.951 |
| CSI [169] | ResNet | 0.474 | 0.984 | 0.702 | 0.800 | 0.560 | 0.977 | 0.524 | 0.946 | 0.565 | 0.927 |
| SupCSI [169] | ResNet | 0.417 | 0.984 | 0.660 | 0.869 | 0.323 | 1.000 | 0.601 | 0.946 | 0.500 | 0.950 |
| **ReSeND** | | **0.750** | **0.820** | 0.685 | 0.894 | **0.660** | **0.854** | **0.963** | **0.181** | **0.765** | **0.687** |

In the single-source case (Table 5.8 top), the support set is always drawn from the Photo domain, while the test set is taken from the remaining three domains. The multi-source benchmark (Table 5.8 bottom) is inherited from [140]: each of the four domains is used as the test set in turn, with the added complexity that the support set is composed of images from multiple domains that have partial class overlap (as shown in Fig. 5.7). From our results, we observe that SimCLR performs particularly well when the test domain is Sketch, but it is outperformed by other approaches in the other settings. On the other hand, ReSeND consistently achieves top results across all benchmarks, indicating its robustness to domain shift without any specific strategy to address it.

**OOD with budget-limited finetuning.** As previously mentioned, ReSeND does not require fine-tuning on the support set to perform semantic novelty detection. Therefore, it is not straightforward to make a fair comparison with existing OOD approaches which instead require a learning phase on the support set. However, we believe that it's important to contextualize ReSeND in the current literature to provide a clearer overview of its performance. To accomplish this, we focus on the challenging PACS multi-source setting and compare ReSeND to several

Table 5.9 Comparison with finetuning-based state-of-the-art OOD methods. Best result in bold and second best underlined.

| OOD Methods | PACS **Multi-Source** | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Fine-Tun. | Eval. | ArtPainting | | Sketch | | Cartoon | | Photo | | **Avg** | |
| | | | AUROC ↑ | FPR95 ↓ | AUROC ↑ | FPR95 ↓ | AUROC ↑ | FPR95 ↓ | AUROC ↑ | FPR95 ↓ | AUROC ↑ | FPR95 ↓ |
| MSP [145] | ✓ | ✓ | 0.617 | 0.973 | 0.412 | 0.998 | <u>0.781</u> | **0.767** | 0.752 | 0.905 | 0.640 | 0.911 |
| ODIN [147] | ✓ | ✓ | 0.602 | 0.977 | 0.425 | 0.998 | **0.785** | <u>0.774</u> | 0.782 | 0.912 | 0.649 | 0.915 |
| Energy [149] | ✓ | ✓ | 0.583 | 0.987 | 0.543 | 0.996 | 0.687 | 0.802 | 0.845 | 0.924 | 0.665 | 0.927 |
| GradNorm [150] | ✓ | ✓ | 0.637 | 0.954 | 0.514 | 1.000 | 0.762 | **0.767** | 0.851 | 0.861 | 0.691 | 0.896 |
| OODformer [272] | ✓ | ✓ | <u>0.703</u> | <u>0.929</u> | 0.610 | 0.973 | 0.776 | 0.802 | 0.732 | 0.773 | <u>0.705</u> | <u>0.869</u> |
| Mahalanobis [163] | ✓ | ✓ | 0.596 | 0.976 | 0.559 | 0.933 | 0.682 | 0.909 | <u>0.861</u> | 0.849 | 0.665 | 0.916 |
| Gram [164] | ✓ | ✓ | 0.448 | 0.962 | **0.885** | **0.713** | 0.536 | 0.946 | 0.838 | <u>0.579</u> | 0.677 | 0.800 |
| Mahalanobis [163] | × | ✓ | 0.596 | 0.976 | 0.466 | 0.981 | 0.593 | 0.926 | 0.808 | 0.935 | 0.616 | 0.954 |
| Gram [164] | × | ✓ | 0.494 | 0.960 | <u>0.840</u> | <u>0.844</u> | 0.494 | 0.954 | 0.797 | 0.981 | 0.656 | 0.935 |
| **ReSeND** | × | ✓ | **0.750** | **0.820** | 0.685 | 0.894 | 0.660 | 0.854 | **0.963** | **0.181** | **0.765** | **0.687** |

standards and state-of-the-art OOD methods. To ensure a fair comparison, we allow these methods to fine-tune (refine the original ImageNet1k pre-trained model) on the support set for the same duration and using the same computational resources as ReSeND during its prediction phase ($\sim$ 30s on 1 GPU). Mahalanobis [163] and Gram [164] are both metric-based approaches and their distance between test samples and the support set can be computed using a non-fine-tuned model (although this was not the strategy proposed by the authors). For them, we evaluated both the fine-tuned and non-fine-tuned versions. The results in Table 5.9 demonstrate that ReSeND outperforms all the other methods, which would need more time or resources to achieve similar performance. This highlights the effectiveness of ReSeND as a powerful tool for semantic novelty detection in situations with limited budget constraints.

**Open-set Domain Generalization.** The good results obtained by ReSeND in the evaluated settings suggest that it can be applied to a variety of real-world tasks. We focus on the challenging open-set domain generalization problem which was introduced in [140] (Fig. 5.7). In this setting, Multiple source domains are combined together and their different label sets cause some classes to exist in many more domains than other classes. The target domain is drawn from a different distribution with a significant shift in terms of style and semantic content, and it contains more classes than the source, which should be identified as unknown during test time. Existing closed-set DG methods are able to learn classification models that generalize to the unseen target domain containing the same categories of the source. One common approach to identifying new classes is using a threshold on the Maximum Softmax Probability (MSP) and classifying samples with uncertain predictions as unknown, as is done in DAML. This same technique can also be applied to SagNet,

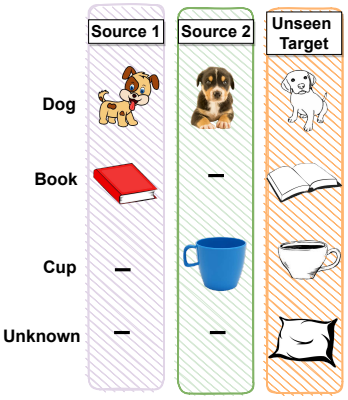



Fig. 5.7 Open-Set DG setting

Table 5.10 Open-Set DG experiments.

| | Single-Source | | | | | |
|---|---|---|---|---|---|---|
| | PACS | | | Office-Home | | |
| | AUROC | Acc | H-Score | AUROC | Acc | H-Score |
| **ReSeND** | 0.685 | - | - | 0.685 | - | - |
| SagNet [275] + MSP | 0.643 | 55.85 | 48.64 | 0.699 | 67.58 | 59.92 |
| SagNet+ **ReSeND** | **0.700** | 55.85 | **52.17** | **0.714** | 67.58 | **61.01** |
| Diversify [104] + MSP | 0.643 | 52.06 | 48.12 | 0.696 | 70.49 | 60.03 |
| Diversify+ **ReSeND** | **0.691** | 52.06 | **51.19** | **0.707** | 70.49 | **60.77** |

| | Multi-Source | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | PACS | | | Office-Home | | | Multi-Datasets | | |
| | AUROC | Acc | H-Score | AUROC | Acc | H-Score | AUROC | Acc | H-Score |
| **ReSeND** | 0.765 | - | - | 0.674 | - | - | 0.686 | - | - |
| DAML [140] + MSP | 0.657 | 62.85 | 52.99 | 0.651 | 55.28 | 52.37 | 0.695 | 45.90 | 47.88 |
| DAML+**ReSeND** | **0.722** | 62.85 | **57.93** | **0.683** | 55.28 | **54.13** | **0.720** | 45.90 | **49.96** |
| Swad [101] + MSP | 0.570 | 60.52 | 42.85 | 0.661 | 53.49 | 51.06 | 0.661 | 47.90 | 49.10 |
| Swad+**ReSeND** | **0.700** | 60.52 | **57.05** | **0.682** | 53.49 | **52.92** | **0.682** | 47.90 | **50.73** |

Diversify, and SWAD. However, the results can be further improved by using a method that is better suited for identifying semantic novelties across domains, such as ReSeND. We use the source domains as the support set and the target as the test set. The ReSeND evaluation procedure is applied to the target samples to obtain a normality score for each of them. These scores are then combined with the Maximum Softmax Probability (MSP) of reference methods through a simple ensemble strategy. The ensemble strategy is used because the normality scores and the MSP results are based on different input features, and combining them can maximize the final accuracy for unknown class rejection. The obtained results are presented in Table 5.10. Combining ReSeND with other methods consistently improves both the AUROC and the H-score, with the accuracy on known classes remaining unchanged, as ReSeND does not affect predictions on known classes.

### 5.2.4 Further analysis and discussions

**Learnable Relational Module.** To evaluate the impact of the design choices for the relational module in ReSeND, we examine alternative methods for combining the features of sample pairs. We investigate the effect of replacing our transformer-based relational module in ReSeND with hand-designed aggregation functions (*Max*/*Sum*/*Concat*). This approach uses an MLP, with a similar number of learnable parameters as the transformer-based module, whose output is fed to the final semantic similarity head. For the *Concat* case, we use feature concatenation as done in [42]: it's important to note that the *permutation invariance property* of our transformer is lost when using feature concatenation as the order of the images in the pair impacts

Table 5.11 Results obtained with different configurations of the relational module. We compare ReSeND with handcrafted feature aggregation strategies for sample pairs.

| | | PACS - **Multi-Source** | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | ArtPainting | | Sketch | | Cartoon | | Photo | | **Avg.** | |
| | | AUROC ↑ | FPR95 ↓ | AUROC ↑ | FPR95 ↓ | AUROC ↑ | FPR95 ↓ | AUROC ↑ | FPR95 ↓ | AUROC ↑ | FPR95 ↓ |
| **ReSeND** | | 0.750 | 0.820 | 0.685 | 0.894 | 0.660 | 0.854 | 0.963 | 0.181 | **0.765** | **0.687** |
| Aggreg. | Max | 0.676 | 0.899 | 0.785 | 0.742 | 0.616 | 0.940 | 0.827 | 0.786 | 0.726 | 0.842 |
| | Sum | 0.583 | 0.976 | 0.446 | 0.988 | 0.514 | 0.996 | 0.575 | 1.000 | 0.530 | 0.990 |
| | Concat | 0.676 | 0.842 | 0.710 | 0.790 | 0.635 | 0.902 | 0.921 | 0.438 | 0.736 | 0.743 |

the final predictions. Table 5.11 shows the results of this analysis on the PACS multi-source setting. We argue that the superior performance of ReSeND originates from having learned the feature aggregation function rather than relying on a fixed approach imposed a priori. However, the *Max* and *Concat* methods still achieve good results, better than the second-best method in Table 5.9, OODFormer ($\text{Avg}_{AUROC}$: 0.705, $\text{Avg}_{FPR95}$: 0.869). This further supports the effectiveness of the relational reasoning approach for semantic novelty detection. We want to point out that one of the key features of ReSeND is its capability to simultaneously learn the feature embedding and the semantic similarity metric through an end-to-end training process. As noted by Sung et al. [209], this approach has an advantage over other methods that only learn the feature embedding using a fixed similarity measure (e.g. Euclidean) [277] or methods that learn a similarity measure using a fixed feature representation [278, 279].

**Regression vs Classification.** Sec. 5.2.1 discusses how the relational reasoning learning paradigm can be approached in two ways: as a binary classification or a regression problem. We argue that using regression, which results in a continuous value for semantic similarity, is more appropriate. The main difference between these two approaches is the behavior of the loss function. Fig. 5.8 shows the trend of the loss for the two different approaches: classification cross entropy (CE) and regression mean squared error (MSE). Both methods assign a high loss to a low probability ($p \approx 0$) and vice versa. However, when the probability values are very small, CE is higher than MSE. While the MSE gives more importance through higher loss values to hard samples belonging to the intermediate probability region, the CE focuses more on easy samples ($p > 0.75$) pushing their already high probability values to the same even higher output. The final result of using CE is that the difference among samples is minimized, which is not desirable when trying to use confidence as a semantic similarity metric. In Fig. 5.9, we evaluate the performance of ReSeND using two different loss functions. We use the same dataset benchmarks

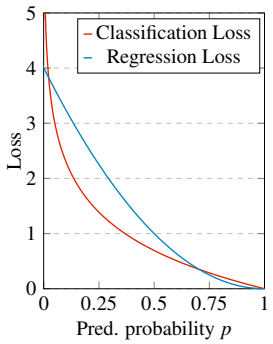



Fig. 5.8 Loss trend for the probability of the correct class.

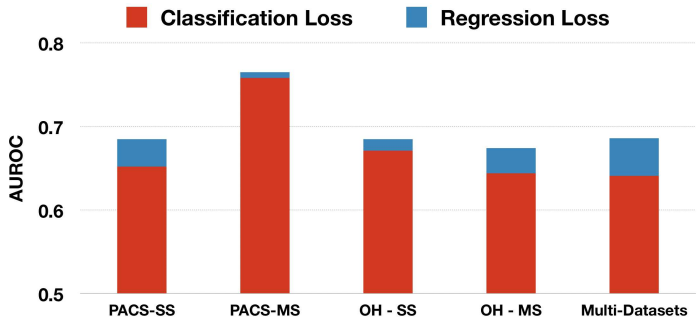



Fig. 5.9 AUROC comparison with ReSeND trained for classification through Cross Entropy Loss or for Regression via MSE. OH stands for Office-Home. SS and MS indicate respectively the Single- and Multi-Source settings.

that were previously used for open-set DG analysis, and we observe that while both loss functions produce good results, the regression loss consistently outperforms the classification loss across all benchmarks.

### 5.2.5 Computational Cost

In Table 5.12 we show the computational cost of our proposed approach and its best competitor SimCLR [40] for the experiments in Table 5.8. Our analysis shows the total number of FLOPs (Floating Point Operations) required for a single forward pass of the network, the training, and the inference time (in milliseconds). The results show that ReSeND, despite a significantly lower number of FLOPs and faster training time than SimCLR, has better performance (in terms of AUROC) and faster inference time. This confirms that ReSeNd represents a great solution in real-world applications where fine-tuning on the in-distribution is unfeasible.

Table 5.12 Cost analysis on PACS Single-Source experiment. Hardware: i) Inference: CPU: Intel(R) Core(TM) i7-9800X CPU @ 3.80GHz, GPU (x1): Nvidia TITAN Xp ii) Training 16 units: CPU: Intel(R) Xeon(R) Silver 4210 CPU @ 2.20GHz, GPU (x1): Tesla V100 SXM2 16GB

| | **Cost analysis** | | | | |
|---|---|---|---|---|---|
| | FLOPs | Training Time (ms) | Inference Time (ms) | AUROC (Avg.) | FPR95 (Avg.) |
| ReSeND | $3.74 \times 10^9$ | $7.20 \times 10^7$ | 3376.67 | **0.685** | 0.847 |
| SimCLR [40] | $7.84 \times 10^9$ | $1.98 \times 10^8$ | 4060.46 | 0.597 | **0.791** |

### 5.2.6 Conclusions

In this section, we explored the problem of semantic novelty detection by studying how conventional representation learning methods can be applied to it. Additionally, we presented ReSeND, a representation learning method that uses relational reasoning to model semantic similarity between pairs of samples. ReSeND is based on a basic transformer architecture, and after being trained on ImageNet1k, it can determine if a test sample belongs to a known or an unknown category by comparing it to a reference support set without requiring any fine-tuning. Our comprehensive experimental evaluation demonstrates the efficiency of ReSeND in both intra- and cross-domain scenarios and its ability to serve as a plug-and-play module, converting traditional closed-set domain generalization methods into robust open-set approaches with outstanding results. The ability to accurately identify unknown categories without any training time latency, and thus preventing incorrect annotations, is crucial for many real-world applications. We believe that our research can open up opportunities for future studies in this area, specifically focusing on new paradigms or more sophisticated architectures for relational reasoning.

**Algorithm 1** HyMOS training procedure

**Input:** $\{\mathbf{x}^s, y^s\} \in \mathscr{S}$, $\mathbf{x}^t \in \mathscr{T}$, $\alpha_m$, AdaIN model
**Output:** $Enc$ and $Proj$

**procedure** TRANSFORM($\mathbf{x}$)
  $styleAugment = random(True, False)$
  $\mathbf{x}' = randomCrop(\mathbf{x})$
  **if** $styleAugment$ **then**
    **return** $styleTransf(\mathbf{x}')$ ▷ target style
  **else**
    **return** $grayScale(jitter((\mathbf{x}')))$

**procedure** CREATEBATCH($D$) ▷ $D$: set of domains
  $batch = [\,]$ ▷ balance domains and categories
  **for each** $y^s$ **in** $\{1, ..., |\mathscr{C}_s|\}$ **do**
    **for each** $D_i$ **with** $i$ **in** $\{1, ..., |D|\}$ **do**
      $\mathbf{x}'_{(y^s, D_i)} = transform(\mathbf{x}_{(y^s, D_i)})$
      $batch.append(\mathbf{x}_{(y^s, D_i)}, \mathbf{x}'_{(y^s, D_i)})$
  **return** $batch$ ▷ $len(batch) = |\mathscr{C}_s| \times |D| \times 2$

**procedure** MAIN()
  $\hat{T} = [\,]$
  **for** $it$ **in** $range(0, end)$ **do**
    **if** $it$ **in** $break\text{-}points$ **then**
      $\hat{T} = [\,]$
      $\alpha \leftarrow$ (Eq. 5.4) ; $\alpha_c = \alpha_m \cdot \alpha$
      **for** $x_t$ **in** $\mathscr{T}$ **do**
        $\mathbf{z}^t = Proj(Enc(\mathbf{x}^t))$
        $h_{y^s} \leftarrow$ `closest prototype to` $\mathbf{z}^t$
        **if** $d_{h_{y^s}}(\mathbf{z}^t) < \alpha_c$ **then** ▷ self-training
          $\hat{y}^t = y^s$ ; $\hat{T}.append(\mathbf{x}^t, \hat{y}^t)$
    $B = createBatch(\mathscr{S}.concat(\hat{T}))$
    $\mathbf{z} = Proj(Enc(B))$
    $loss = SupClr(\mathbf{z})$ (Eq. 5.1)
    Update $Enc, Proj \leftarrow \nabla loss$

# Chapter 6

# Conclusions

*This Chapter summarizes the main contributions and results of this thesis discussing open issues and future research directions.*

## 6.1 Summary

Despite the impressive performance that the newest cutting-edge deep architectures have in visual recognition, they still need a huge amount of data to reach a sufficiently high level of knowledge generalization. As opposed to humans who with small effort and a few examples are able to learn general and robust knowledge, these models struggle to maintain equally good performances on real-world applications where, typically, there is no guarantee in terms of data distribution.

In this thesis, we tackled this major problem by proposing a new perspective. Taking inspiration from how humans learn, we proposed to include a self-supervised objective in the supervised training process. We demonstrated how the joint use of supervised and self-supervised learning boosts the generalization ability of the models that learn domain-agnostic regularities of the objects (e.g. shape) moving the focus from the domain-specific parts (e.g. texture). More specifically the first part of this thesis (Chapter 3) considered the scenario where only the covariate distribution shift between the training and test set holds. We show how by using as auxiliary objectives the jigsaw puzzle/rotation recognition tasks (JiGen, Section 3.1) or RGB-D reconstruction in multi-modal learning (Tran-Adapt, Section 3.2), it is possible to considerably boost the recognition performance of the model on the unseen test set. In the second part of the thesis (Chapter 4), with the extra challenge of having both data distribution and category shifts, we still exploit the properties of self-supervision to mitigate the domain gap between train and test data (Section 4.1), while identifying samples belonging to unknown classes in the test set (ROS, Section 4.2). Finally, in Chapter 5 we show how relation-based self-supervised approaches allow for straightforward integration of supervision resulting in robust models for open-set recognition (HyMOS, Chapter 5.1) and semantic novelty detection (ReSeND Section 5.2).

In conclusion, this thesis demonstrates that exploiting self-supervision jointly with supervision during the learning process leads the model to focus on the most relevant information reaching a robust and general knowledge easily applicable to the open world. Considering the current advancements in edge computing, we further include a discussion about the computational cost for each proposed approach. Despite being often overlooked, especially in the current technological revolution, it represents an essential indicator to assess the limits and potentials of deep models.

## 6.2 Open issues and Future Works

While the focus of this thesis is primarily on object recognition, the proposed supervised and self-supervised learning integration can yield benefits in various other computer vision tasks as detection or segmentation which are essential for robotics applications. This would bridge the gap between static AI algorithms used in virtual environments and real AI agents that can interact with the physical world, paving the way for more dynamic and adaptive AI systems. In addition, this thesis explores relation-based self-supervised tasks in an open-world setting, primarily for unknown detection. However, there is potential for extending the proposed approaches to lifelong learning, for the continual discovery of novel categories and progressive knowledge growth. To pursue these new research directions, it may be necessary to supplement vision-based approaches with natural language also facilitating human-machine interaction.

Indeed, besides being popular in computer vision, self-supervised objectives are also largely employed to build large language models (e.g. GPT-4 [280]) which are pioneering the current AI revolution. As extensively discussed in this thesis, the major limitation of vision models is their inability to manage changes in domain and image semantic content. In this respect, natural language could represent the extra modality needed to close this knowledge gap. Generative multi-modal models like DALL·E 2 [281] could be used to synthesize new images with the right style or semantic content from a simple text prompt. The use of natural language across domains has some examples in the recent literature [282, 283], however, future works may investigate a broader perspective, considering the design of a totally autonomous system able to merge vision and language without any priors about the challenges that it is going to meet [258]. The embedding space resulting from the combination of vision and language has the potential to capture common sense knowledge producing image features agnostic to the specific visual domain [284, 285]. Moreover, the learned representations are highly effective in capturing the notion of similarity thus mitigating the out-of-distribution (OOD) problem [286]. Considering the complexity and expressive potential of language, it is expected that this type of research will have significant impacts also on more challenging tasks like those involving videos [287, 288], 3D learning [289] or robotic control [290, 291].

Furthermore, language could allow easier interactions with humans promoting explainability and model transparency [292–294]: if the decision of an AI agent is

explained in words, a human could provide auxiliary information to correct the cause of possible errors improving the trustworthiness and fairness [295] of AI systems.

Overall, adaptability, generalization, and being ready to adapt to novelty are key aspects of developing reliable AI systems, so we believe that the topics and methods discussed in this thesis will remain relevant references for future research.

# References


[1] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *ICCV*, 2019.

[2] Shuran Song, Samuel P. Lichtenberg, and Jianxiong Xiao. SUN RGB-D: A RGB-D scene understanding benchmark suite. In *CVPR*, 2015.

[3] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016.

[4] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *CVPR*, 2017.

[5] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European Conference on Computer Vision (ECCV)*, 2016.

[6] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. In *ICLR*, 2018.

[7] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *ECCV*, 2010.

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

[9] Bryan C Russell, Antonio Torralba, Kevin P Murphy, and William T Freeman. Labelme: a database and web-based tool for image annotation. *International journal of computer vision*, 77(1):157–173, 2008.

[10] Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *2004 conference on computer vision and pattern recognition workshop*, pages 178–178. IEEE, 2004.

[11] Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 3485–3492. IEEE, 2010.

[12] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M. Hospedales. Deeper, broader and artier domain generalization. In *International Conference on Computer Vision (ICCV)*, 2017.

[13] Xingchao Peng, Ben Usman, Neela Kaushik, Judy Hoffman, Dequan Wang, and Kate Saenko. Visda: The visual domain adaptation challenge. *arXiv preprint arXiv:1710.06924*, 2017.

[14] Laurens van der Maaten and Geoffrey E. Hinton. Visualizing high-dimensional data using t-sne. *JMLR*, 9:2579–2605, 2008.

[15] Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *CVPR*, 2014.

[16] Saurabh Gupta, Ross B. Girshick, Pablo Andrés Arbeláez, and Jitendra Malik. Learning Rich Features from RGB-D Images for Object Detection and Segmentation. In *ECCV*, 2014.

[17] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. In *ICLR*, 2015.

[18] Mohammad Reza Loghmani, Luca Robbiano, Mirco Planamente, Kiru Park, Barbara Caputo, and Markus Vincze. Unsupervised domain adaptation through inter-modal rotation for RGB-D object recognition. *RA-L*, 5(4):6631–6638, 2020.

[19] Silvia Bucci, Mohammad Reza Loghmani, and Tatiana Tommasi. On the effectiveness of image rotation for open set domain adaptation. In *ECCV*, 2020.

[20] Gabriela Csurka. *Domain Adaptation in Computer Vision Applications*. Springer Publishing Company, Incorporated, 1st edition, 2017.

[21] https://en.wikipedia.org/wiki/Intelligence.

[22] Shane Legg, Marcus Hutter, et al. A collection of definitions of intelligence. *Frontiers in Artificial Intelligence and applications*, 157:17, 2007.

[23] Kimin Lee, Honglak Lee, Kibok Lee, and Jinwoo Shin. Training confidence-calibrated classifiers for detecting out-of-distribution samples. *arXiv preprint arXiv:1711.09325*, 2017.

[24] Hong Liu, Zhangjie Cao, Mingsheng Long, Jianmin Wang, and Qiang Yang. Separate to adapt: Open set domain adaptation via progressive separation. In *CVPR*, 2019.

[25] Pulkit Agrawal, Joao Carreira, and Jitendra Malik. Learning to see by moving. In *Proceedings of the IEEE international conference on computer vision*, pages 37–45, 2015.

[26] Carl Doersch, Abhinav Gupta, and Alexei A. Efros. Unsupervised visual representation learning by context prediction. In *ICCV*, 2015.

[27] Xiaolong Wang and Abhinav Gupta. Unsupervised learning of visual representations using videos. In *International Conference on Computer Vision (ICCV)*, 2015.

[28] Rodrigo Santa Cruz, Basura Fernando, Anoop Cherian, and Stephen Gould. Visual permutation learning. In *Computer Vision and Pattern Recognition (CVPR)*, 2017.

[29] Mehdi Noroozi, Ananth Vinjimoor, Paolo Favaro, and Hamed Pirsiavash. Boosting self-supervised learning via knowledge transfer. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[30] Mehdi Noroozi, Hamed Pirsiavash, and Paolo Favaro. Representation learning by learning to count. In *ICCV*, 2017.

[31] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *ECCV*, 2016.

[32] Ishan Misra, C. Lawrence Zitnick, and Martial Hebert. Shuffle and Learn: Unsupervised Learning using Temporal Order Verification. In *European Conference on Computer Vision (ECCV)*, 2016.

[33] Fabio M Carlucci, Antonio D'Innocente, Silvia Bucci, Barbara Caputo, and Tatiana Tommasi. Domain generalization by solving jigsaw puzzles. In *CVPR*, 2019.

[34] Silvia Bucci, Antonio D'Innocente, Yujun Liao, Fabio Maria Carlucci, Barbara Caputo, and Tatiana Tommasi. Self-supervised learning across domains. *IEEE TPAMI*, 2021.

[35] Andrea Ferreri, Silvia Bucci, and Tatiana Tommasi. Multi-modal rgb-d scene recognition across domains. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2199–2208, 2021.

[36] Dapeng Du, Limin Wang, Huiling Wang, Kai Zhao, and Gangshan Wu. Translate-to-recognize networks for rgb-d scene recognition. In *CVPR*, 2019.

[37] Silvia Bucci, Francesco Cappio Borlino, Barbara Caputo, and Tatiana Tommasi. Distance-based hyperspherical classification for multi-source open-set domain adaptation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1119–1128, 2022.

[38] Francesco Cappio Borlino, Silvia Bucci, and Tatiana Tommasi. Semantic novelty detection via relational reasoning. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXV*, pages 183–200. Springer, 2022.

[39] Mert Bulent Sariyildiz, Yannis Kalantidis, Karteek Alahari, and Diane Larlus. No reason for no supervision: Improved generalization in supervised models. In *ICLR 2023-International Conference on Learning Representations*, 2023.

[40] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020.

[41] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. In *NeurIPS*, 2020.

[42] Massimiliano Patacchiola and Amos Storkey. Self-supervised relational reasoning for representation learning. In *NeurIPS*, 2020.

[43] Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474*, 2014.

[44] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning transferable features with deep adaptation networks. In *ICML*, 2015.

[45] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I. Jordan. Deep transfer learning with joint adaptation networks. In *ICML*, 2017.

[46] Hongliang Yan, Yukang Ding, Peihua Li, Qilong Wang, Yong Xu, and Wangmeng Zuo. Mind the class weight bias: Weighted maximum mean discrepancy for unsupervised domain adaptation. In *CVPR*, 2017.

[47] Zhikang Zou, Xiaoye Qu, Pan Zhou, Shuangjie Xu, Xiaoqing Ye, Wenhao Wu, and Jin Ye. Coarse to fine: Domain adaptive crowd counting via adversarial scoring network. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 2185–2194, 2021.

[48] Bharath Bhushan Damodaran, Benjamin Kellenberger, Remi Flamary, Devis Tuia, and Nicolas Courty. DeepJDOT: Deep Joint Distribution Optimal Transport for Unsupervised Domain Adaptation. In *ECCV*, 2018.

[49] Chen-Yu Lee, Tanmay Batra, Mohammad Haris Baig, and Daniel Ulbricht. Sliced wasserstein discrepancy for unsupervised domain adaptation. In *CVPR*, 2019.

[50] Werner Zellinger, Thomas Grubinger, Edwin Lughofer, Thomas Natschläger, and Susanne Saminger-Platz. Central moment discrepancy (CMD) for domain-invariant representation learning. In *ICLR*, 2017.

[51] Pietro Morerio, Jacopo Cavazza, and Vittorio Murino. Minimal-entropy correlation alignment for unsupervised deep domain adaptation. In *ICLR*, 2017.

[52] Baochen Sun, Jiashi Feng, and Kate Saenko. Return of frustratingly easy domain adaptation. In *AAAI*, 2016.

[53] Ruijia Xu, Guanbin Li, Jihan Yang, and Liang Lin. Larger norm more transferable: An adaptive feature norm approach for unsupervised domain adaptation. In *ICCV*, 2019.

[54] Fabio Maria Carlucci, Lorenzo Porzi, Barbara Caputo, Elisa Ricci, and Samuel Rota Bulò. Autodial: Automatic domain alignment layers. In *International Conference on Computer Vision (ICCV)*, 2017.

[55] Yanghao Li, Naiyan Wang, Jianping Shi, Jiaying Liu, and Xiaodi Hou. Revisiting batch normalization for practical domain adaptation. In *ICLR*, 2017.

[56] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030, 2016.

[57] Eric Tzeng, Judy Hoffman, Trevor Darrell, and Kate Saenko. Adversarial discriminative domain adaptation. In *CVPR*, 2017.

[58] Paolo Russo, Fabio Maria Carlucci, Tatiana Tommasi, and Barbara Caputo. From source to target and back: symmetric bi-directional adaptive gan. In *Computer Vision and Pattern Recognition (CVPR)*, 2018.

[59] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, 2017.

[60] Mingsheng Long, ZHANGJIE CAO, Jianmin Wang, and Michael I Jordan. Conditional adversarial domain adaptation. In *NeurIPS*, 2018.

[61] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

[62] Konstantinos Bousmalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. Unsupervised pixel-level domain adaptation with gans. In *Computer Vision and Pattern Recognition (CVPR)*, 2017.

[63] Swami Sankaranarayanan, Yogesh Balaji, Carlos D Castillo, and Rama Chellappa. Generate to adapt: Aligning domains using generative adversarial networks. In *Computer Vision and Pattern Recognition (CVPR)*, 2018.

[64] Rui Gong, Wen Li, Yuhua Chen, and Luc Van Gool. Dlow: Domain flow for adaptation and generalization. In *CVPR*, 2019.

[65] Yawei Luo, Ping Liu, Tao Guan, Junqing Yu, and Yi Yang. Adversarial style mining for one-shot unsupervised domain adaptation. In *NeurIPS*, 2020.

[66] Abhishek Kumar, Prasanna Sattigeri, Kahini Wadhawan, Leonid Karlinsky, Rogerio Feris, Bill Freeman, and Gregory Wornell. Co-regularized alignment for unsupervised domain adaptation. *Advances in Neural Information Processing Systems*, 31, 2018.

[67] Zhekai Du, Jingjing Li, Hongzu Su, Lei Zhu, and Ke Lu. Cross-domain gradient discrepancy minimization for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3937–3946, 2021.

[68] Guoliang Kang, Lu Jiang, Yi Yang, and Alexander G Hauptmann. Contrastive adaptation network for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4893–4902, 2019.

[69] Jaemin Na, Heechul Jung, Hyung Jin Chang, and Wonjun Hwang. Fixbi: Bridging domain spaces for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1094–1103, 2021.

[70] Astuti Sharma, Tarun Kalluri, and Manmohan Chandraker. Instance level affinity-based transfer for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5361–5371, 2021.

[71] Kuniaki Saito, Yoshitaka Ushiku, and Tatsuya Harada. Asymmetric tri-training for unsupervised domain adaptation. In *International Conference on Machine Learning, (ICML)*, 2017.

[72] Hsin-Ying Lee, Hung-Yu Tseng, Jia-Bin Huang, Maneesh Singh, and Ming-Hsuan Yang. Diverse image-to-image translation via disentangled representations. In *Proceedings of the European conference on computer vision (ECCV)*, pages 35–51, 2018.

[73] Xingchao Peng, Zijun Huang, Ximeng Sun, and Kate Saenko. Domain agnostic learning with disentangled representations. In *International Conference on Machine Learning*, pages 5102–5112. PMLR, 2019.

[74] Xingchao Peng, Yichen Li, and Kate Saenko. Domain2vec: Domain embedding for unsupervised domain adaptation. In *European conference on computer vision*, pages 756–774. Springer, 2020.

[75] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Unsupervised domain adaptation with residual transfer networks. In *NIPS*, pages 136–144, 2016.

[76] Zelun Luo, Yuliang Zou, Judy Hoffman, and Li F Fei-Fei. Label efficient learning of transferable representations acrosss domains and tasks. In *NIPS*, pages 165–177, 2017.

[77] Ying Jin, Ximei Wang, Mingsheng Long, and Jianmin Wang. Minimum class confusion for versatile domain adaptation. In *European Conference on Computer Vision*, pages 464–480. Springer, 2020.

[78] Donghyun Kim, Kuniaki Saito, Tae-Hyun Oh, Bryan A Plummer, Stan Sclaroff, and Kate Saenko. Cds: Cross-domain self-supervised pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9123–9132, 2021.

[79] Jaehoon Choi, Minki Jeong, Taekyung Kim, and Changick Kim. Pseudo-labeling curriculum for unsupervised domain adaptation. *arXiv preprint arXiv:1908.00262*, 2019.

[80] Geoffrey French, Michal Mackiewicz, and Mark Fisher. Self-ensembling for visual domain adaptation. *arXiv preprint arXiv:1706.05208*, 2017.

[81] Jiaolong Xu, Liang Xiao, and Antonio Lopez. Self-supervised domain adaptation for computer vision tasks. *IEEE ACCESS*, 7:156694–156706, 2019.

[82] Yu Mitsuzumi, Go Irie, Daiki Ikami, and Takashi Shibata. Generalized domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1084–1093, 2021.

[83] Luciano Spinello and Kai Oliver Arras. Leveraging RGB-D data: Adaptive fusion and domain adaptation for object detection. In *ICRA*, 2012.

[84] Judy Hoffman, Saurabh Gupta, Jian Leong, Sergio Guadarrama, and Trevor Darrell. Cross-modal adaptation for rgb-d detection. In *ICRA*, 2016.

[85] Xiao Li, Min Fang, Ju-Jie Zhang, and Jinqiao Wu. Domain adaptation from rgb-d to rgb images. *Signal Process.*, 131:27–35, 2017.

[86] Jing Wang and Kuangen Zhang. Unsupervised domain adaptation learning algorithm for rgb-d staircase recognition. *arXiv:1903.01212*, 2019.

[87] Massimiliano Mancini, Lorenzo Porzi, Samuel Rota Bulo, Barbara Caputo, and Elisa Ricci. Boosting domain adaptation by discovering latent domains. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3771–3780, 2018.

[88] Ruijia Xu, Ziliang Chen, Wangmeng Zuo, Junjie Yan, and Liang Lin. Deep cocktail network: Multi-source unsupervised domain adaptation with category shift. In *CVPR*, 2018.

[89] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Vaughan. A theory of learning from different domains. *Machine Learning*, 79:151–175, 2010.

[90] John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman. Learning bounds for domain adaptation. *Advances in neural information processing systems*, 20, 2007.

[91] Judy Hoffman, Mehryar Mohri, and Ningshan Zhang. Algorithms and theory for multiple-source adaptation. In *NIPS*, 2018.

[92] Yitong Li, David E Carlson, et al. Extracting relationships by multi-domain matching. *Advances in Neural Information Processing Systems*, 31, 2018.

[93] Kuniaki Saito, Kohei Watanabe, Yoshitaka Ushiku, and Tatsuya Harada. Maximum classifier discrepancy for unsupervised domain adaptation. *CVPR*, 2018.

[94] Luyu Yang, Yogesh Balaji, Ser-Nam Lim, and Abhinav Shrivastava. Curriculum manager for source selection in multi-source domain adaptation. In *ECCV*, 2020.

[95] Jiang Guo, Darsh J Shah, and Regina Barzilay. Multi-source domain adaptation with mixture of experts. *arXiv preprint arXiv:1809.02256*, 2018.

[96] Sayan Rakshit, Biplab Banerjee, Gemma Roig, and Subhasis Chaudhuri. Unsupervised multi-source domain adaptation driven by deep adversarial ensemble learning. In *German Conference on Pattern Recognition*, pages 485–498. Springer, 2019.

[97] Yingwei Pan, Ting Yao, Yehao Li, Yu Wang, Chong-Wah Ngo, and Tao Mei. Transferrable prototypical networks for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2239–2247, 2019.

[98] Hang Wang, Minghao Xu, Bingbing Ni, and Wenjun Zhang. Learning to combine: Knowledge aggregation for multi-source domain adaptation. In *ECCV*, 2020.

[99] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[100] Zeyi Huang, Haohan Wang, Eric P Xing, and Dong Huang. Self-challenging improves cross-lization. In *European Conference on Computer Vision*, pages 124–140. Springer, 2020.

[101] Junbum Cha, Sanghyuk Chun, Kyungjae Lee, Han-Cheol Cho, Seunghyun Park, Yunsung Lee, and Sungrae Park. Swad: Domain generalization by seeking flat minima. In *NeurIPS*, 2021.

[102] Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*, 2018.

[103] Riccardo Volpi, Hongseok Namkoong, Ozan Sener, John Duchi, Vittorio Murino, and Silvio Savarese. Generalizing to unseen domains via adversarial data augmentation. In *NeurIPS*, 2018.

[104] Zijian Wang, Yadan Luo, Ruihong Qiu, Zi Huang, and Mahsa Baktashmotlagh. Learning to diversify for single domain generalization. In *ICCV*, 2021.

[105] Aditya Khosla, Tinghui Zhou, Tomasz Malisiewicz, Alexei Efros, and Antonio Torralba. Undoing the damage of dataset bias. In *ECCV*, 2012.

[106] Prithvijit Chattopadhyay, Yogesh Balaji, and Judy Hoffman. Learning to balance specificity and invariance for in and out of domain generalization. In *European Conference on Computer Vision*, pages 301–318. Springer, 2020.

[107] Vihari Piratla, Praneeth Netrapalli, and Sunita Sarawagi. Efficient lization via common-specific low-rank decomposition. In *International Conference on Machine Learning*, pages 7728–7738. PMLR, 2020.

[108] Antonio D'Innocente and Barbara Caputo. Domain generalization with domain-specific aggregation modules. In *German Conference on Pattern Recognition (GCPR)*, 2018. Code available at https://github.com/VeloDC/D-SAM_public.

[109] Massimiliano Mancini, Samuel Rota Bulo, Barbara Caputo, and Elisa Ricci. Robust place categorization with deep domain generalization. *IEEE RAL*, 2018.

[110] Muhammad Ghifary, W. Bastiaan Kleijn, Mengjie Zhang, and David Balduzzi. Domain generalization for object recognition with multi-task autoencoders. In *ICCV*, 2015.

[111] Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex C. Kot. Domain generalization with adversarial feature learning. In *CVPR*, 2018.

[112] Saeid Motiian, Marco Piccirilli, Donald A. Adjeroh, and Gianfranco Doretto. Unified deep supervised domain adaptation and generalization. In *ICCV*, 2017.

[113] Ya Li, Xinmei Tian, Mingming Gong, Yajing Liu, Tongliang Liu, Kun Zhang, and Dacheng Tao. Deep domain generalization via conditional invariant adversarial networks. In *The European Conference on Computer Vision (ECCV)*, September 2018.

[114] Joshua Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[115] Shiv Shankar, Vihari Piratla, Soumen Chakrabarti, Siddhartha Chaudhuri, Preethi Jyothi, and Sunita Sarawagi. Generalizing across domains via cross-gradient training. In *ICLR*, 2018.

[116] Francesco Cappio Borlino, Antonio D'Innocente, and Tatiana Tommasi. Rethinking domain generalization baselines. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 9227–9233. IEEE, 2021.

[117] Yingjun Du, Xiantong Zhen, Ling Shao, and Cees GM Snoek. Metanorm: Learning to normalize few-shot batches across domains. In *International Conference on Learning Representations*, 2020.

[118] Yuyang Zhao, Zhun Zhong, Fengxiang Yang, Zhiming Luo, Yaojin Lin, Shaozi Li, and Nicu Sebe. Learning to generalize unseen domains via memory-based multi-source meta-learning for person re-identification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6277–6286, 2021.

[119] Yogesh Balaji, Swami Sankaranarayanan, and Rama Chellappa. Metareg: Towards domain generalization using meta-regularization. *Advances in neural information processing systems*, 31, 2018.

[120] Da Li, Jianshu Zhang, Yongxin Yang, Cong Liu, Yi-Zhe Song, and Timothy M. Hospedales. Episodic training for domain generalization. *arXiv preprint:1902.00113*, 2019.

[121] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M. Hospedales. Learning to generalize: Meta-learning for domain generalization. In *AAAI*, 2018.

[122] Zhangjie Cao, Mingsheng Long, Jianmin Wang, and Michael I. Jordan. Partial transfer learning with selective adversarial networks. In *CVPR*, 2018.

[123] Zhangjie Cao, Lijia Ma, Mingsheng Long, and Jianmin Wang. Partial adversarial domain adaptation. In *Europeran Conference on Computer Vision (ECCV)*, 2018.

[124] Jing Zhang, Zewei Ding, Wanqing Li, and Philip Ogunbona. Importance weighted adversarial nets for partial domain adaptation. In *CVPR*, 2018.

[125] Toshihiko Matsuura, Kuniaki Saito, and Tatsuya Harada. Twins: Two weighted inconsistency-reduced networks for partial domain adaptation. *Preprint arXiv:1812.07405*, 2018.

[126] Zhihong Chen, Chao Chen, Zhaowei Cheng, Boyuan Jiang, Ke Fang, and Xinyu Jin. Selective transfer with reinforced transfer network for partial domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12706–12714, 2020.

[127] Jian Liang, Yunbo Wang, Dapeng Hu, Ran He, and Jiashi Feng. A balanced and uncertainty-aware approach for partial domain adaptation. In *European Conference on Computer Vision*, pages 123–140. Springer, 2020.

[128] Jian Hu, Hongya Tuo, Chao Wang, Lingfeng Qiao, Haowen Zhong, Junchi Yan, Zhongliang Jing, and Henry Leung. Discriminative partial domain adversarial network. In *European Conference on Computer Vision*, pages 632–648. Springer, 2020.

[129] Wenxiao Xiao, Zhengming Ding, and Hongfu Liu. Implicit semantic response alignment for partial domain adaptation. *Advances in Neural Information Processing Systems*, 34:13820–13833, 2021.

[130] Silvia Bucci, Antonio D'Innocente, and Tatiana Tommasi. Tackling partial domain adaptation with self-supervision. In *ICIAP*, 2019.

[131] Kuniaki Saito, Shohei Yamamoto, Yoshitaka Ushiku, and Tatsuya Harada. Open set domain adaptation by backpropagation. In *ECCV*, 2018.

[132] Qianyu Feng, Guoliang Kang, Hehe Fan, and Yi Yang. Attract or distract: Exploit the margin of open set. In *ICCV*, 2019.

[133] Yingwei Pan, Ting Yao, Yehao Li, Chong-Wah Ngo, and Tao Mei. Exploring category-agnostic clusters for open-set domain adaptation. In *CVPR*, 2020.

[134] Jogendra Nath Kundu, Naveen Venkat, Ambareesh Revanur, R Venkatesh Babu, et al. Towards inheritable models for open-set domain adaptation. In *CVPR*, 2020.

[135] Yadan Luo, Zijian Wang, Zi Huang, and Mahsa Baktashmotlagh. Progressive graph learning for open-set domain adaptation. In *International Conference on Machine Learning*, pages 6468–6478. PMLR, 2020.

[136] Jie Liu, Xiaoqing Guo, and Yixuan Yuan. Unknown-oriented learning for open set domain adaptation. In *European Conference on Computer Vision*, pages 334–350. Springer, 2022.

[137] Izhak Golan and Ran El-Yaniv. Deep anomaly detection using geometric transformations. In *NeurIPS*, 2018.

[138] Sayan Rakshit, Dipesh Tamboli, Pragati Shuddhodhan Meshram, Biplab Banerjee, Gemma Roig, and Subhasis Chaudhuri. Multi-source open-set deep adversarial domain adaptation. In *ECCV*, 2020.

[139] Jing Li, Liu Yang, and Qinghua Hu. Self-supervised vision transformer based nearest neighbor classification for multi-source open-set domain adaptation. In *Pacific Rim International Conference on Artificial Intelligence*, pages 542–554. Springer, 2022.

[140] Yang Shu, Zhangjie Cao, Chenyu Wang, Jianmin Wang, and Mingsheng Long. Open domain generalization with domain-augmented meta-learning. In *CVPR*, 2021.

[141] Poojan Oza, Hien V Nguyen, and Vishal M Patel. Multiple class novelty detection under data distribution shift. In *European Conference on Computer Vision*, pages 432–449. Springer, 2020.

[142] Arthur Zimek, Erich Schubert, and Hans-Peter Kriegel. A survey on unsupervised outlier detection in high-dimensional numerical data. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 5(5):363–387, 2012.

[143] Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. Deep one-class classification. In *ICML*, 2018.

[144] Shiyu Liang, Yixuan Li, and Rayadurgam Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. *arXiv preprint arXiv:1706.02690*, 2017.

[145] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *ICLR*, 2017.

[146] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

[147] Shiyu Liang, Yixuan Li, and R. Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. In *ICLR*, 2018.

[148] Y. C. Hsu, Y. Shen, H. Jin, and Z. Kira. Generalized odin: Detecting out-of-distribution image without learning from out-of-distribution data. In *CVPR*, 2020.

[149] Weitang Liu, Xiaoyun Wang, John Owens, and Yixuan Li. Energy-based out-of-distribution detection. *NeurIPS*, 2020.

[150] Rui Huang, Andrew Geng, and Yixuan Li. On the importance of gradients for detecting distributional shifts in the wild. *NeurIPS*, 2021.

[151] Ki Hyun Kim, Sangwoo Shim, Yongsub Lim, Jongseob Jeon, Jeongwoo Choi, Byungchan Kim, and Andre S. Yoon. Rapp: Novelty detection with reconstruction along projection pathway. In *ICLR*, 2020.

[152] Anne-Sophie Collin and Christophe De Vleeschouwer. Improved anomaly detection by training an autoencoder with skip connections on images corrupted with stain-shaped noise. In *ICPR*, 2021.

[153] Hyunjong Park, Jongyoun Noh, and Bumsub Ham. Learning memory-guided normality for anomaly detection. In *CVPR*, 2020.

[154] Murat Sensoy, Lance M. Kaplan, Federico Cerutti, and Maryam Saleki. Uncertainty-aware deep classifiers using generative models. In *AAAI*, 2020.

[155] Yingda Xia, Yi Zhang, Fengze Liu, Wei Shen, and Alan Yuille. Synthesize then compare: Detecting failures and anomalies for semantic segmentation. In *ECCV*, 2020.

[156] Eric Nalisnick, Akihiro Matsukawa, Yee Whye Teh, Dilan Gorur, and Balaji Lakshminarayanan. Do deep generative models know what they don't know? In *ICLR*, 2019.

[157] Lawrence Neal, Matthew Olson, Xiaoli Fern, Weng-Keen Wong, and Fuxin Li. Open set learning with counterfactual images. In *ECCV*, 2018.

[158] Zongyuan Ge, Sergey Demyanov, Zetao Chen, and Rahil Garnavi. Generative openmax for multi-class open set classification. In *BMVC*, 2017.

[159] Kimin Lee, Honglak Lee, Kibok Lee, and Jinwoo Shin. Training confidence-calibrated classifiers for detecting out-of-distribution samples. In *ICLR*, 2018.

[160] Dan Hendrycks, Mantas Mazeika, and Thomas Dietterich. Deep anomaly detection with outlier exposure. In *ICLR*, 2019.

[161] Aristotelis-Angelos Papadopoulos, Mohammad Reza Rajati, Nazim Shaikh, and Jiamian Wang. Outlier exposure with confidence control for out-of-distribution detection. *Neurocomputing*, 441:138–150, 2021.

[162] Jiefeng Chen, Yixuan Li, Xi Wu, Yingyu Liang, and Somesh Jha. Atom: Robustifying out-of-distribution detection using outlier mining. In *ECML*, 2021.

[163] Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *NeurIPS*, 2018.

[164] Chandramouli Shama Sastry and Sageev Oore. Detecting out-of-distribution examples with Gram matrices. In *ICML*, 2020.

[165] Simon Jenni, Hailin Jin, and Paolo Favaro. Steering self-supervised feature learning beyond local pixel statistics. In *CVPR*, 2020.

[166] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *ICCV*, 2019.

[167] Chun-Liang Li, Kihyuk Sohn, Jinsung Yoon, and Tomas Pfister. Cutpaste: Self-supervised learning for anomaly detection and localization. In *CVPR*, 2021.

[168] Liron Bergman and Yedid Hoshen. Classification-based anomaly detection for general data. In *ICLR*, 2020.

[169] Jihoon Tack, Sangwoo Mo, Jongheon Jeong, and Jinwoo Shin. Csi: Novelty detection via contrastive learning on distributionally shifted instances. In *NeurIPS*, 2020.

[170] Vikash Sehwag, Mung Chiang, and Prateek Mittal. Ssd: A unified framework for self-supervised outlier detection. In *ICLR*, 2021.

[171] Jim Winkens, Rudy Bunel, Abhijit Guha Roy, Robert Stanforth, Vivek Natarajan, Joseph R. Ledsam, Patricia MacWilliams, Pushmeet Kohli, Alan Karthikesalingam, Simon Kohl, Taylan Cemgil, S. M. Ali Eslami, and Olaf Ronneberger. Contrastive training for improved out-of-distribution detection. *arXiv:2007.05566*, 2020.

[172] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020.

[173] Ching-Yao Chuang, Joshua Robinson, Yen-Chen Lin, Antonio Torralba, and Stefanie Jegelka. Debiased contrastive learning. In *NeurIPS*, 2020.

[174] Deepak Pathak, Philipp Krähenbühl, Jeff Donahue, Trevor Darrell, and Alexei Efros. Context encoders: Feature learning by inpainting. In *CVPR*, 2016.

[175] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Colorization as a proxy task for visual understanding. In *CVPR*, 2017.

[176] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *ICML*, 2021.

[177] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *CVPR*, 2021.

[178] Alexander Kolesnikov, Xiaohua Zhai, and Lucas Beyer. Revisiting self-supervised visual representation learning. In *CVPR*, 2019.

[179] Alejandro Newell and Jia Deng. How useful is self-supervised pretraining for visual tasks? In *CVPR*, 2020.

[180] Linus Ericsson, Henry Gouk, and Timothy M. Hospedales. How Well Do Self-Supervised Models Transfer? In *CVPR*, 2021.

[181] Herbert Freeman and L. Garder. Apictorial jigsaw puzzles: The computer solution of a problem in pattern recognition. *IEEE Trans. Electronic Computers*, 13(2):118–127, 1964.

[182] David A. Kosiba, Pierre M. Devaux, Sanjay Balasubramanian, Tarak Gandhi, and Rangachar Kasturi. An automatic jigsaw puzzle solver. In *ICPR*, 1994.

[183] Taeg Sang Cho, Shai Avidan, and William T. Freeman. The patch transform. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 32:1489–1501, 2009.

[184] Dror Sholomon, Omid David, and Nathan Netanyahu. A generalized genetic algorithm-based solver for very large jigsaw puzzles of complex types. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 28, 2014.

[185] Benedict J. Brown, Corey Toler-Franklin, Diego Nehab, Michael Burns, David Dobkin, Andreas Vlachopoulos, Christos Doumas, Szymon Rusinkiewicz, and Tim Weyrich. A system for high-volume acquisition and matching of fresco fragments: Reassembling Theran wall paintings. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 27(3), August 2008.

[186] Marie-Morgane Paumard, David Picard, and Hedi Tabia. Image reassembly combining deep learning and shortest path problem. In *ECCV*, 2018.

[187] William Marande and Gertraud Burger. Mitochondrial dna as a genomic jigsaw puzzle. *Science*, 318(5849):415–415, 2007.

[188] Andrew C. Gallagher. Jigsaw puzzles with pieces of unknown orientation. In *CVPR*, 2012.

[189] Taeg Sang Cho, Shai Avidan, and William T. Freeman. A probabilistic image jigsaw puzzle solver. In *CVPR*, 2010.

[190] Yingyi Chen, Xi Shen, Yahui Liu, Qinghua Tao, and Johan AK Suykens. Jigsaw-vit: Learning jigsaw puzzles in vision transformer. *Pattern Recognition Letters*, 2022.

[191] J. Xu, L. Xiao, and A. M. López. Self-supervised domain adaptation for computer vision tasks. *IEEE Access*, 7:156694–156706, 2019.

[192] Longlong Jing, Xiaodong Yang, Jingen Liu, and Yingli Tian. Self-supervised spatiotemporal feature learning via video rotation prediction. *arXiv preprint arXiv:1811.11387*, 2018.

[193] Yuexiang Li, Jiawei Chen, Xinpeng Xie, Kai Ma, and Yefeng Zheng. Self-loop uncertainty: A novel pseudo-label for semi-supervised medical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 614–623. Springer, 2020.

[194] Yuhan Zhang, Mingchao Li, Zexuan Ji, Wen Fan, Songtao Yuan, Qinghuai Liu, and Qiang Chen. Twin self-supervision based semi-supervised learning (ts-ssl): Retinal anomaly classification in sd-oct images. *Neurocomputing*, 462:491–505, 2021.

[195] Zhirong Wu, Yuanjun Xiong, Stella X. Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *CVPR*, 2018.

[196] Yonglong Tian, Chen Sun, Ben Poole, Dilip Krishnan, Cordelia Schmid, and Phillip Isola. What makes for good views for contrastive learning? In *NeurIPS*, 2020.

[197] Senthil Purushwalkam and Abhinav Gupta. Demystifying contrastive self-supervised learning: Invariances, augmentations and dataset biases. In *NeurIPS*, 2020.

[198] Longhui Wei, Lingxi Xie, Jianzhong He, Jianlong Chang, Xiaopeng Zhang, Wengang Zhou, Houqiang Li, and Qi Tian. Can semantic labels assist self-supervised visual representation learning? *arXiv preprint arXiv:2011.08621*, 2020.

[199] Yifan Zhang, Bryan Hooi, Dapeng Hu, Jian Liang, and Jiashi Feng. Unleashing the power of contrastive self-supervised visual models via contrast-regularized fine-tuning. *arXiv preprint arXiv:2102.06605*, 2021.

[200] Orchid Majumder, Avinash Ravichandran, Subhransu Maji, Marzia Polito, Rahul Bhotika, and Stefano Soatto. Revisiting contrastive learning for few-shot classification. *arXiv preprint arXiv:2101.11058*, 2021.

[201] Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C. Lawrence Zitnick, and Ross B. Girshick. CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning. In *CVPR*, 2017.

[202] Adam Santoro, David Raposo, David G. T. Barrett, Mateusz Malinowski, Razvan Pascanu, Peter W. Battaglia, and Tim Lillicrap. A simple neural network module for relational reasoning. In *NeurIPS*, 2017.

[203] David Raposo, Adam Santoro, David G. T. Barrett, Razvan Pascanu, Tim Lillicrap, and Peter W. Battaglia. Discovering objects and their relations from entangled scene representations. In *ICLR Workshop*, 2017.

[204] Adam Santoro, Ryan Faulkner, David Raposo, Jack W. Rae, Mike Chrzanowski, Theophane Weber, Daan Wierstra, Oriol Vinyals, Razvan Pascanu, and Timothy P. Lillicrap. Relational recurrent neural networks. In *NeurIPS*, 2018.

[205] Junting Pan, Siyu Chen, Mike Zheng Shou, Yu Liu, Jing Shao, and Hongsheng Li. Actor-context-actor relation network for spatio-temporal action localization. In *CVPR*, 2021.

[206] Vinicius Zambaldi, David Raposo, Adam Santoro, Victor Bapst, Yujia Li, Igor Babuschkin, Karl Tuyls, David Reichert, Timothy Lillicrap, Edward Lockhart, Murray Shanahan, Victoria Langston, Razvan Pascanu, Matthew Botvinick, Oriol Vinyals, and Peter Battaglia. Deep reinforcement learning with relational inductive biases. In *ICLR*, 2019.

[207] Han Hu, Jiayuan Gu, Zheng Zhang, Jifeng Dai, and Yichen Wei. Relation networks for object detection. In *CVPR*, 2018.

[208] Peter Battaglia, Jessica Blake Chandler Hamrick, Victor Bapst, Alvaro Sanchez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, Caglar Gulcehre, Francis Song,

Andy Ballard, Justin Gilmer, George E. Dahl, Ashish Vaswani, Kelsey Allen, Charles Nash, Victoria Jayne Langston, Chris Dyer, Nicolas Heess, Daan Wierstra, Pushmeet Kohli, Matt Botvinick, Oriol Vinyals, Yujia Li, and Razvan Pascanu. Relational inductive biases, deep learning, and graph networks. *arXiv:1806.01261*, 2018.

[209] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. Learning to compare: Relation network for few-shot learning. In *CVPR*, 2018.

[210] Hongguang Zhang, Piotr Koniusz, Songlei Jian, Hongdong Li, and Philip H. S. Torr. Rethinking class relations: Absolute-relative supervised and unsupervised few-shot learning. In *CVPR*, 2021.

[211] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. In *ICLR*, 2019.

[212] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical Report 7694, California Institute of Technology, 2007.

[213] Antonio Torralba and Alexei A. Efros. Unbiased look at dataset bias. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[214] Jia Deng, Wei Dong, Richard Socher, Li jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[215] Steven Bird, Ewan Klein, and Edward Loper. *Natural language processing with Python: analyzing text with the natural language toolkit*. O'Reilly Media, Inc., 2009.

[216] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *AISTATS*, 2011.

[217] Rich Caruana. Multitask learning. *Mach. Learn.*, 28(1):41–75, 1997.

[218] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2012.

[219] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[220] Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. *arXiv preprint arXiv:2007.01434*, 2020.

[221] Zhengming Ding and Yun Fu. Deep domain generalization with structured low-rank constraint. *IEEE Transactions on Image Processing*, 27:304–313, 2017.

[222] Kaiyang Zhou, Yongxin Yang, Timothy Hospedales, and Tao Xiang. Deep domain-adversarial image generation for domain generalisation. *AAAI*, 2020.

[223] Haohan Wang, Songwei Ge, Zachary Lipton, and Eric P Xing. Learning robust global representations by penalizing local predictive power. In *NeurIPS*, 2019.

[224] Toshihiko Matsuura and Tatsuya Harada. Domain generalization using a mixture of multiple latent domains. In *AAAI*, 2020.

[225] Dino Sejdinovic, Bharath K. Sriperumbudur, Arthur Gretton, and Kenji Fukumizu. Equivalence of distance-based and rkhs-based statistics in hypothesis testing. *The Annals of Statistics*, 41(5):2263–2291, 2013.

[226] Fabio Maria Carlucci, Lorenzo Porzi, Barbara Caputo, Elisa Ricci, and Samuel Rota Bulò. Just DIAL: domain alignment layers for unsupervised domain adaptation. In *Image Analysis and Processing - ICIAP*, 2017.

[227] Spyros Gidaris, Andrei Bursuc, Nikos Komodakis, Patrick Pérez, and Matthieu Cord. Learning representations by predicting bags of visual words. In *CVPR*, 2020.

[228] Yude Wang, Jie Zhang, Meina Kan, Shiguang Shan, and Xilin Chen. Self-supervised equivariant attention mechanism for weakly supervised semantic segmentation. In *CVPR*, 2020.

[229] Antonio D'Innocente, Francesco Cappio Borlino, Silvia Bucci, Barbara Caputo, and Tatiana Tommasi. One-shot unsupervised cross-domain detection. *arXiv:2005.11610*, 2020.

[230] Antonio Alliegro, Davide Boscaini, and Tatiana Tommasi. Joint supervised and self-supervised learning for 3d real-world challenges. *arXiv:2004.07392*, 2020.

[231] Tanvi A. Patel, Vipul K. Dabhi, and Harshadkumar B. Prajapati. Survey on scene classification techniques. In *IEEE ICACCS*, 2020.

[232] Delu Zeng, Minyu Liao, Mohammad Tavakolian, Yulan Guo, Bolei Zhou, Dewen Hu, Matti Pietikäinen, and Li Liu. Deep learning for scene classification: A survey. *arXiv:2101.10531*, 2021.

[233] Bolei Zhou, Agata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. Learning deep features for scene recognition using places database. In *NIPS*, 2014.

[234] D. Banica and C. Sminchisescu. Second-order constrained parametric proposals and sequential search-based structured prediction for semantic segmentation in rgb-d images. In *CVPR*, 2015.

[235] Saurabh Gupta, Pablo Arbelaez, Ross Girshick, and Jitendra Malik. Indoor scene understanding with rgb-d images: Bottom-up segmentation, object detection and semantic segmentation. *IJCV*, 112:133–149, 2014.

[236] A. Wang, J. Cai, J. Lu, and T. Cham. Modality and component aware feature fusion for rgb-d scene classification. In *CVPR*, 2016.

[237] S. Gupta, J. Hoffman, and J. Malik. Cross modal distillation for supervision transfer. In *CVPR*, 2016.

[238] Xinhang Song, Shuqiang Jiang, Luis Herranz, and Chengpeng Chen. Learning effective rgb-d representations for scene recognition. *IEEE TIP*, 28(2):980–993, 2019.

[239] Camille Couprie, Clément Farabet, Laurent Najman, and Yann LeCun. Indoor semantic segmentation using depth information. In *ICLR*, 2013.

[240] Yanhua Cheng, Rui Cai, Zhiwei Li, Xin Zhao, and Kaiqi Huang. Locality-sensitive deconvolution networks with gated fusion for RGB-D indoor semantic segmentation. In *CVPR*, 2017.

[241] Anran Wang, Jiwen Lu, Jianfei Cai, Tat Jen Cham, and Gang Wang. Large-margin multi-modal deep learning for rgb-d object recognition. *TMM*, 17(11):1887–1898, 2015.

[242] Yabei Li, Junge Zhang, Yanhua Cheng, Kaiqi Huang, and Tieniu Tan. Df2net: Discriminative feature learning and fusion network for rgb-d indoor scene classification. In *AAAI*, 2018.

[243] Yuan Yuan, Zhitong Xiong, and Qi Wang. Acm: Adaptive cross-modal graph convolutional neural networks for rgb-d scene recognition. In *AAAI*, 2019.

[244] Ali Ayub and Alan R. Wagner. Centroid based concept learning for rgb-d indoor scene classification. In *BMVC*, 2020.

[245] Y.-H. Tsai, W.-C. Hung, S. Schulter, K. Sohn, M.-H. Yang, and M. Chandraker. Learning to adapt structured output space for semantic segmentation. In *CVPR*, 2018.

[246] Yuang Liu, Wei Zhang, and Jun Wang. Source-free domain adaptation for semantic segmentation. In *CVPR*, 2021.

[247] Suman Saha, Anton Obukhov, Danda Pani Paudel, Menelaos Kanakis, Yuhua Chen, Stamatios Georgoulis, and Luc Van Gool. Learning to relate depth and semantics for unsupervised domain adaptation. In *CVPR*, 2021.

[248] Shuang Li, Chi Harold Liu, Qiuxia Lin, Qi Wen, Limin Su, Gao Huang, and Zhengming Ding. Deep residual correction network for partial domain adaptation. *IEEE TPAMI*, 2020.

[249] Zhangjie Cao, Kaichao You, Mingsheng Long, Jianmin Wang, and Qiang Yang. Learning to transfer examples for partial domain adaptation. In *CVPR*, 2019.

[250] Abhijit Bendale and Terrance E Boult. Towards open set deep networks. In *CVPR*, 2016.

[251] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In *ECCV*, 2016.

[252] Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. In *AAAI*, 2018.

[253] Jesse Dodge, Suchin Gururangan, Dallas Card, Roy Schwartz, and Noah A. Smith. Show your work: Improved reporting of experimental results. In *EMNLP*, 2019.

[254] Mario Lucic, Karol Kurach, Marcin Michalski, Sylvain Gelly, and Olivier Bousquet. Are gans created equal? a large-scale study. In *NeurIPS*, 2018.

[255] The Machine Learning Reproducibility Checklist. https://www.cs.mcgill.ca/~jpineau/ReproducibilityChecklist.pdf. Accessed: 4-3-2020.

[256] Reproducibility Challenge. https://reproducibility-challenge.github.io/neurips2019/. Accessed: 4-3-2020.

[257] Mohammad Reza Loghmani, Markus Vincze, and Tatiana Tommasi. Positive-unlabeled learning for open set domain adaptation. *Pattern Recognition Letters*, 136:198 – 204, 2020.

[258] Kaichao You, Zhangjie Cao, Mingsheng Long, Jianmin Wang, and Michael I. Jordan. Universal domain adaptation. In *CVPR*, 2019.

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

[260] Kuniaki Saito, Shohei Yamamoto, Yoshitaka Ushiku, and Tatsuya Harada. Open set domain adaptation by backpropagation. *arXiv preprint arXiv:1804.10427*, 2018.

[261] Pau Panareda Busto and Juergen Gall. Open set domain adaptation. In *ICCV*, 2017.

[262] Bo Fu, Zhangjie Cao, Mingsheng Long, and Jianmin Wang. Learning to detect open classes for universal domain adaptation. In *ECCV*, 2020.

[263] Kuniaki Saito, Donghyun Kim, Stan Sclaroff, and Kate Saenko. Universal domain adaptation through self-supervision. In *NeurIPS*, 2020.

[264] Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *ICML*, 2020.

[265] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *ICCV*, 2017.

[266] Yunsheng Li, Lu Yuan, Yinpeng Chen, Pei Wang, and Nuno Vasconcelos. Dynamic transfer for multi-source domain adaptation. In *CVPR*, 2021.

[267] Shujun Wang, Lequan Yu, Caizi Li, Chi-Wing Fu, and Pheng-Ann Heng. Learning from extrinsic and intrinsic supervisions for domain generalization. In *ECCV*, 2020.

[268] Paola Cascante-Bonilla, Fuwen Tan, Yanjun Qi, and Vicente Ordonez. Curriculum labeling: Revisiting pseudo-labeling for semi-supervised learning. In *AAAI*, 2021.

[269] Lucas Deecke, Lukas Ruff, Robert A. Vandermeulen, and Hakan Bilen. Transfer-based semantic anomaly detection. In *ICML*, 2021.

[270] Ying Cheng, Ruize Wang, Zhihao Pan, Rui Feng, and Yuejie Zhang. Look, listen, and attend: Co-attention network for self-supervised audio-visual representation learning. In *ACM Multimedia*, 2020.

[271] Toby Segaran. *Programming Collective Intelligence: Building Smart Web 2.0 Applications*. O'Reilly, 2007.

[272] Rajat Koner, Poulami Sinhamahapatra, Karsten Roscher, Stephan Günnemann, and Volker Tresp. Oodformer: Out-of-distribution detection transformer. In *BMVC*, 2021.

[273] Lukas Ruff, Jacob R. Kauffmann, Robert A. Vandermeulen, Grégoire Montavon, Wojciech Samek, Marius Kloft, Thomas G. Dietterich, and Klaus-Robert Müller. A unifying review of deep and shallow anomaly detection. *Proceedings of the IEEE*, 109(5):756–795, 2021.

[274] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herve Jegou. Training data-efficient image transformers & distillation through attention. In *ICML*, 2021.

[275] Hyeonseob Nam, HyunJae Lee, Jongchan Park, Wonjun Yoon, and Donggeun Yoo. Reducing domain gap by reducing style bias. In *CVPR*, 2021.

[276] Rui Huang and Yixuan Li. Mos: Towards scaling out-of-distribution detection for large semantic space. In *CVPR*, 2021.

[277] Dario Fontanel, Fabio Cermelli, Massimiliano Mancini, Samuel Rota Bulo, Elisa Ricci, and Barbara Caputo. Boosting deep open world recognition by clustering. *IEEE RAL*, 5(4):5985–5992, 2020.

[278] Thomas Mensink, Jakob Verbeek, Florent Perronnin, and Gabriela Csurka. Metric learning for large scale image classification: Generalizing to new classes at near-zero cost. In *ECCV*, 2012.

[279] Dong Chen, Xudong Cao, Liwei Wang, Fang Wen, and Jian Sun. Bayesian face revisited: A joint formulation. In *ECCV*, 2012.

[280] OpenAI. Gpt-4 technical report, 2023.

[281] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

[282] Lisa Dunlap, Clara Mohri, Devin Guillory, Han Zhang, Trevor Darrell, Joseph E Gonzalez, Aditi Raghunathan, and Anna Rohrbach. Using language to extend to unseen domains. In *The Eleventh International Conference on Learning Representations*, 2022.

[283] Hongjing Niu, Hanting Li, Feng Zhao, and Bin Li. Domain-unified prompt representations for source-free domain generalization. *arXiv preprint arXiv:2209.14926*, 2022.

[284] Yangjun Ruan, Yann Dubois, and Chris J Maddison. Optimal representations for covariate shift. *ICLR*, 2022.

[285] Seonwoo Min, Nokyung Park, Siwon Kim, Seunghyun Park, and Jinkyu Kim. Grounding visual representations with texts for domain generalization. In *European Conference on Computer Vision*, pages 37–53. Springer, 2022.

[286] Yifei Ming, Ziyang Cai, Jiuxiang Gu, Yiyou Sun, Wei Li, and Yixuan Li. Delving into out-of-distribution detection with vision-language representations. *Advances in Neural Information Processing Systems*, 35:35087–35102, 2022.

[287] Giacomo Zara, Alessandro Conti, Subhankar Roy, Stéphane Lathuilière, Paolo Rota, and Elisa Ricci. The unreasonable effectiveness of large language-vision models for source-free video domain adaptation. *arXiv preprint arXiv:2308.09139*, 2023.

[288] Giacomo Zara, Subhankar Roy, Paolo Rota, and Elisa Ricci. Autolabel: Clip-based framework for open-set video domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11504–11513, 2023.

[289] Antonio Alliegro, Francesco Cappio Borlino, and Tatiana Tommasi. 3dos: Towards 3d open set learning-benchmarking and understanding semantic novelty detection on point clouds. *Advances in Neural Information Processing Systems*, 35:21228–21240, 2022.

[290] Siddharth Karamcheti, Suraj Nair, Annie S Chen, Thomas Kollar, Chelsea Finn, Dorsa Sadigh, and Percy Liang. Language-driven representation learning for robotics. *arXiv preprint arXiv:2302.12766*, 2023.

[291] Austin Stone, Ted Xiao, Yao Lu, Keerthana Gopalakrishnan, Kuang-Huei Lee, Quan Vuong, Paul Wohlhart, Brianna Zitkovich, Fei Xia, Chelsea Finn, et al. Open-world object manipulation using pre-trained vision-language models. *arXiv preprint arXiv:2303.00905*, 2023.

[292] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016.

[293] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Anchors: High-precision model-agnostic explanations. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[294] Anna Rogers, Olga Kovaleva, and Anna Rumshisky. A primer in bertology: What we know about how bert works. *Transactions of the Association for Computational Linguistics*, 8:842–866, 2021.

[295] Leonardo Iurada, Silvia Bucci, Timothy M Hospedales, and Tatiana Tommasi. Fairness meets cross-domain learning: a new perspective on models and metrics. *arXiv preprint arXiv:2303.14411*, 2023.

[296] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456, 2015.

[297] Dan Hendrycks, Mantas Mazeika, Saurav Kadavath, and Dawn Song. Using self-supervised learning can improve model robustness and uncertainty. *NeurIPS*, 2019.

[298] Yang You, Igor Gitman, and Boris Ginsburg. Large batch training of convolutional networks. *arXiv preprint arXiv:1708.03888*, 2017.

# Appendix A

# Open-Set Cross-Domain Learning

## A.1 ROS

### A.1.1 Further implementation Details

This section contains extensive implementation details of our method ROS, including the parameters used for running all experiments. Our experiments were conducted on Office-31 [7] and Office-Home [4] datasets using ResNet-50 [219] and VGGNet [259] as backbones. For a visual representation of our architecture, please refer to Figure 4.3 in Section 4.2.

**Encoder $E$, ResNet-50:** it consists of all the layers of a standard ResNet-50 up to the average pooling layer. We update only the last convolutional block with a learning rate of 0.0003 starting with the pre-trained model on ImageNet [214].

**Classifiers $C_1$, $C_2$, ResNet-50:** both models consist of two Fully Connected (FC) layers. The first one has output 256 followed by a Batch Normalization layer [296] and a Leaky-ReLU activation function with a negative slope angle of 0.2. The second one has a different output size in function of the classifier: for $C_1$, it has $|\mathscr{C}_s|$ outputs, while for $C_2$, it has $|\mathscr{C}_s|+1$ outputs (including the unknown category). All layers are trained from scratch using a learning rate of 0.003.

**Rotation classifiers $R_1$, $R_2$, ResNet-50:** their structure is the same as the classifiers described above. In particular, $R_1$ has the output size of $4 \times |\mathscr{C}_s|$ and $R_2$ 4. All the layers are trained from scratch with learning rate 0.003.

**Stage I and Stage II, ResNet-50:** in Stage II, we used the network trained in Stage I as a starting point. We know that, for the semantic classifier, the category set increases by one. To account for this, we set the learning rate of the new unknown class to twice that of the known classes (which were already learned in Stage I).

**Encoder $E$, VGGNet** : it is composed by all the layers of a standard VGG-19 up to the second fully connected layer. We start from the encoder model pre-trained on ImageNet [214] and we update only the last two FC layers, finetuning it with learning rate 0.0003.

**Classifiers $C_1$, $C_2$, $R_1$, $R_2$, VGGNet** : they have exactly the same structure used for the ResNet-50 case described above.

**Stage I and Stage II, VGGNet** : The network trained in Stage I is not used as starting point for Stage II. Still we consider the learning rate of the extra unknown class in Stage II higher with respect to the other classes (1.5), but lower than the value used in case of ResNet-50 (2), where Stage II was inheriting the model of Stage I. We also tried to inherit the Stage I model for Stage II as done in the ResNet case, but for VGG that setting produced lower results.

**Office-31, ResNet-50** : we trained our model using a batch size of 32 and a learning rate defined as specified above, which decreases during training with inverse decay scheduling. We used Stochastic Gradient Descent (SGD) with a weight decay of 0.0005 and momentum of 0.9. The loss weights were set as follows: $\lambda_{1,1} = \lambda_{2,2} = 3$ and $\lambda_{1,2} = \lambda_{2,1} = 0.1$. We trained for 80 epochs the Stage I and for 80 epochs the Stage II. Each experiment was repeated three times, and the result on the target was taken at the last epoch.

**Office-31, VGGNet** : we trained our model using a batch size of 32 and a learning rate defined as specified above, which decreases during training with inverse decay

scheduling. We used Stochastic Gradient Descent (SGD) with a weight decay of 0.0005 and momentum of 0.9. The loss weights were set as follows: $\lambda_{1,1} = \lambda_{2,2} = 3$ and $\lambda_{1,2} = \lambda_{2,1} = 0.1$. We trained for 100 epochs the Stage I and for 200 epochs the Stage II. Each experiment was repeated three times, and the result on the target was taken at the last epoch.

**Office-Home, ResNet-50** : we trained our model using a batch size of 32 and a learning rate defined as specified above, which decreases during training with inverse decay scheduling. We used Stochastic Gradient Descent (SGD) with a weight decay of 0.0005 and momentum of 0.9. The loss weights were set as follows: $\lambda_{1,1} = \lambda_{2,2} = 3$ and $\lambda_{2,1} = 0.1$. Compared to the previous datasets, adding the center loss to the rotation classifier $R_1$ seemed less relevant for this dataset. However, we kept it in the optimization process with a low weight of $\lambda_{1,2} = 0.001$. We trained for 150 epochs the Stage I and for 45 epochs the Stage II. Each experiment was repeated three times, and the result on the target was taken at the last epoch.

It is worth mentioning that we used essentially the same set of parameters for all experimental settings. This demonstrates that our method can generalize well across different datasets and network architectures without requiring specific finetuning of hyperparameters.

### A.1.2 Reproducibility Study

In this section, we expand the reproducibility study presented in Section 4.2 by including additional results obtained on the Office-Home dataset. In Table A.1, we provide a comparison of results reported in the official papers of STA [24], OSBP [260], and UAN [258] on the Office-Home dataset. We focus on the OS accuracy metric since it is the only one that is reported by all three works. We replicated the specific settings described in the original publication for UAN. Specifically, for the Office-Home dataset, the first ten classes in alphabetic order are shared between the source and target domains, the next five are private source classes, and all the remaining ones are private target classes. In the case of Office-31, the first 10 classes are shared between the source and target, the next 10 are private source classes, and the remaining ones are private target classes. It is worth noting that despite using the

Table A.1 The OS accuracy (%) reported in the papers compared to the one obtained in our reproducibility study. The results show the average over three runs and across all sub-domains of Office-31 and Office-Home with the indicated backbones.

| **Reproducibility Study** | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Office-31 (ResNet-50) | | | | | | Office-31 (VGGNet) | | | Office-Home (ResNet-50) | | | | | | |
| $STA_{sum}$ | | | UAN | | | OSBP | | | $STA_{sum}$ | | | UAN | | | |
| $OS_{reported}$ | $OS_{ours}$ | gap | $OS_{reported}$ | $OS_{ours}$ | gap | $OS_{reported}$ | $OS_{ours}$ | gap | $OS_{reported}$ | $OS_{ours}$ | gap | $OS_{reported}$ | $OS_{ours}$ | gap |
| 92.9 | 90.6±1.8 | **2.3** | 89.2 | 87.9±0.03 | **1.3** | 89.1 | 84.2 ±0.4 | **4.9** | 69.5 | 63.3±2.1 | **6.2** | 77.0 | 75.1 ±0.2 | **1.9** |

code provided by the authors and following the instructions of the official papers, our results are lower than those reported, with gaps ranging between 1.9% and 6.2%.

To ensure complete transparency, we provide here a summary of all the implementation details, code, and hyper-parameters used for running the competitor methods.

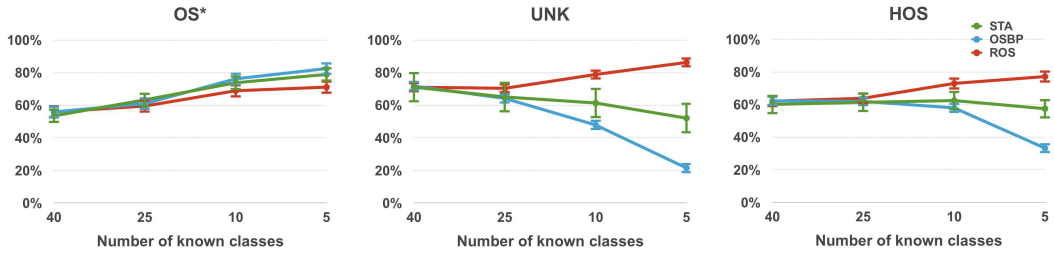
**STA** [24] https://github.com/thuml/Separate_to_Adapt

The code provides instructions on how to run STA for the A→D domain shift of Office-31 using ResNet-50 as the backbone. In Stage I, we trained for 900 iterations (400 for the multi-binary classifier and 500 for the known/unknown classifier), followed by 1900 iterations in Stage II. We used a batch size of 32, SGD with momentum of 0.9, and weight decay set to 0.0005. We used the inverse scheduling for the learning rate that is set as 0.001 in Stage I and 0.0005 in Stage II (10 times smaller for finetuned layers). It's worth noting that since the paper doesn't distinguish between Office-31 and Office-Home regarding hyper-parameters, we used the exact same values for the experiments on Office-Home. It should be noted that there is some ambiguity regarding the specific value of the learning rate for STA. The paper suggests that the learning rate can be fine-tuned within the range of $0.001, 1$ using cross-validation. However, the code does not include any validation procedure, making it unclear how this parameter should be optimized. Furthermore, the learning rate value used for Stage II in the code falls outside the range specified in the paper. In our experiments, we consistently used the learning rate value provided in the code, without further adjustments. We also noticed other inconsistencies between the paper and the code. For example, the paper indicates that in Stage I, the feature extractor is fine-tuned on the source samples, while in the code, the feature extractor is frozen with the pre-trained weights from ImageNet. Furthermore, as we have mentioned in Section 4.2, the paper describes a similarity score based on the *max* operator, while the code uses the *sum* operator. Lastly, although the paper reports results using

VGGNet, the code for this variant is not available, and the paper does not provide specific information about it, making difficult to reproduce the experiments.

**OSBP** [260] https://github.com/ksaito-ut/OPDA_BP
This repository contains the implementation of OSBP using both VGGNet and ResNet-50 backbones. Specifically, the instructions report how to run OSBP on the VisDA-2017 dataset [13] with VGGNet. Regarding the experiments on Office-31 with VGGNet, we used the provided implementation and followed the OSDA paper's instructions using a batch size of 32, SGD with momentum 0.9, learning rate 0.001, and weight decay 0.0005. We trained only the new layers for 500 epochs, while the pre-existing ones were frozen with ImageNet weights. Regarding the experiments done using ResNet-50, we used batch size 32, learning rate 0.001, 300 epochs for Office-Home, and 500 epochs for Office-31. Since the authors highlighted that the library version could significantly affect the results, we used exactly their declared version (Pytorch 0.3), for all the experiments.

**UAN** [258] https://github.com/thuml/Universal-Domain-Adaptation
This repository includes the UAN implementation with ResNet-50 as backbone. The repository contains specific files that provide instructions to run experiments on both Office-31 and Office-Home. Specifically, for Office-31, the model was trained for 20000 iterations with a batch size of 36, SGD with momentum 0.9, learning rate 0.001 for new layers and 0.0001 for finetuned layers using inverse scheduling and weight decay 0.0005. For the Office-Home experiments, the model was trained for 40000 iterations with a batch size of 36, SGD with momentum 0.9, learning rate 0.01 for new layers and 0.001 for finetuned layers using inverse scheduling and weight decay 0.0005. It is important to note that the original evaluation process implemented in the official code saved the UAN performance on the test data at each epoch and presented the best accuracy (OS) at the end of the training. This is not a standard procedure, and to avoid its potentially unfair positive effect, we provide the results obtained after the last epoch, as done for all other benchmark methods in our experiments.




| Office-Home (Avg.) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 40 known classes | | | | 25 known classes | | | |
| | OS | OS* | UNK | **HOS** | OS | OS* | UNK | **HOS** |
| $STA_{sum}$[24] | 53.9±1.7 | 53.5±1.7 | 71.1±3.0 | 60.1±1.7 | 63.5±2.6 | 63.3±3.3 | 65.1±2.2 | 61.4±1.5 |
| OSBP[260] | 56.3±1.6 | 55.9±1.6 | 71.8±3.6 | **62.3±1.8** | 61.4±2.5 | 61.0±2.8 | 64.2±2.8 | 62.0±2.8 |
| **ROS** | 56.2±1.2 | 55.9±1.2 | 71.1±2.8 | 62.1±1.8 | 59.9±1.9 | 59.5±1.9 | 70.4±4.0 | **63.9±2.7** |
| | 10 known classes | | | | 5 known classes | | | |
| | OS | OS* | UNK | **HOS** | OS | OS* | UNK | **HOS** |
| $STA_{sum}$[24] | 72.5±5.8 | 73.8±6.6 | 61.4±14.5 | 62.5±8.6 | 74.4±2.8 | 78.9±6.3 | 52.1±17.7 | 57.5±12.3 |
| OSBP[260] | 73.6±5.7 | 76.2±5.7 | 47.9±4.8 | 58.0±5.1 | 72.3±4.9 | 82.5±5.6 | 21.5±1.6 | 33.3±2.6 |
| **ROS** | 69.9±5.2 | 68.9±5.5 | 78.9±2.9 | **73.0±4.4** | 73.4±6.7 | 71.1±7.6 | 86.3±2.7 | **77.2±5.9** |

Fig. A.1 Accuracy (%) averaged over the three configurations designed for each degree of openness considered: with 40, 25, 10 and 5 known classes. The table reports in details the values used to prepare the plots

### A.1.3 Extended Openness Analysis

Following the openness analysis in Figure 4.7 (Section 4.2), we further evaluate a scenario with lower openness, involving **40** known classes (corresponding to an openness score of $\mathbb{O} = 1 - \frac{40}{65} = 0.38$), using ID:0-39, 15-54, 25-64. Figure A.1 shows the results, which confirm the trend observed in Section 4.2. Given the low UNK and HOS results of UAN, we did not include this method in the ablation analysis and focused only on the two best competitors of ROS, OSBP and STA.

### A.1.4 Sensitivity analysis of the hyper-parameters

To evaluate how changes in hyper-parameter values affect the performance of ROS, we did a sensitivity analysis on Office-31 using ResNet-50 as backbone. The results are shown in Figure A.2. Our analysis demonstrates that ROS is not highly sensitive to variations in hyper-parameter values, with only $\lambda_{2,1}$ resulting in HOS variation of more than 1.0. It is worth noting that the entropy weight can be safely set to 0.1 without the need for hyper-parameter tuning, as previously done in [33, 24, 53]. Regardless of the specific hyper-parameters used, ROS consistently outperforms its closest competitor, OSBP (with $HOS = 83.7$), thus confirming that the superior performance is mainly due to algorithmic innovation rather than hyper-parameter

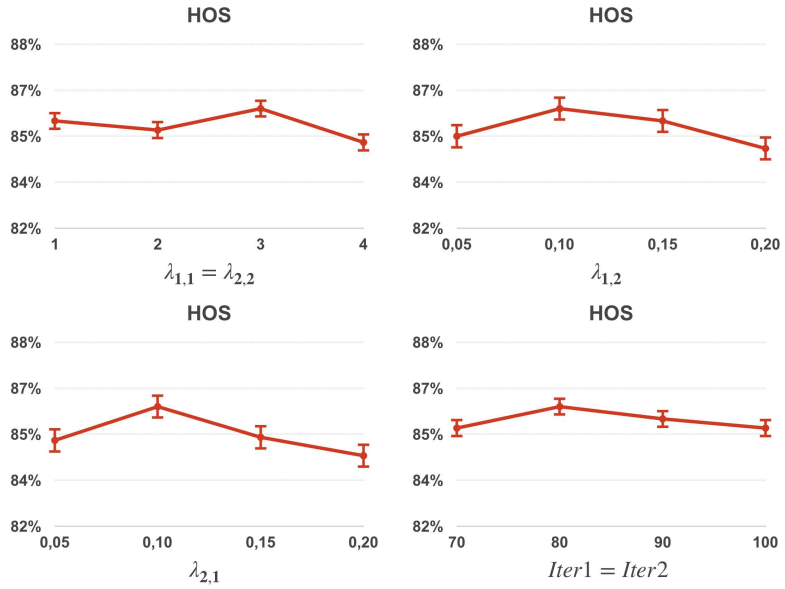



Fig. A.2 Hyper-parameter analysis

Table A.2 Analysis on the use of self-supervised tasks for the two stages of the method and further ablation.

| **Other Self-Supervised Tasks & Ablation Study** | | | | | | | |
|---|---|---|---|---|---|---|---|
| **STAGE I** (AUC-ROC) | A → W | A → D | D → W | W → D | D → A | W → A | **Avg.** |
| **ROS** | 90.1 | 88.1 | 99.4 | 99.9 | 87.5 | 83.8 | **91.5** |
| ROS - Translation | 80.8 | 74.9 | 82.2 | 98.8 | 72.0 | 79.1 | 81.3 |
| ROS - Rotation+Translation | 82.4 | 79.3 | 99.0 | 99.4 | 82.6 | 82.8 | 87.6 |
| ROS - 4-Class Rotation | 58.7 | 57.2 | 70.0 | 78.4 | 55.8 | 56.9 | 62.9 |
| **STAGE II** (HOS) | A → W | A → D | D → W | W → D | D → A | W → A | **Avg.** |
| **ROS** | 82.1 | 82.4 | 96.0 | 99.7 | 77.9 | 77.2 | **85.9** |
| ROS - Jigsaw | 83.1 | 79.3 | 93.5 | 100.0 | 75.5 | 76.1 | 84.6 |
| ROS - Rotation+Jigsaw | 85.7 | 80.5 | 95.0 | 100.0 | 76.0 | 76.7 | 85.7 |
| ROS Stage I - $\lambda_{2,1} = 0$ Stage II | 79.4 | 82.0 | 95.3 | 99.6 | 75.1 | 72.5 | 84.0 |
| ROS Stage I - ROS Stage II+Center Loss | 79.6 | 82.8 | 95.1 | 99.5 | 77.8 | 76.3 | 85.2 |

tuning. Additionally, we underline that we used the same hyper-parameter values for all 18 domain pairs, demonstrating that the choice of hyper-parameter values is robust across datasets. Lastly, we note that ROS has a similar number of parameters compared to other competing approaches. While $\lambda_{1,1}$ and $\lambda_{2,2}$ are defined separately, they are actually constrained to the same value. Therefore, ROS has a total of three parameters, plus two for the training iterations, which is the same as the most recent AoD method (as shown in Equation (3) of [132]).

**Algorithm 2** Compute normality score and Generate $\mathscr{D}_t^{knw}$ & $\mathscr{D}_t^{unk}$

**Input:**

Trained networks $E$ and $R_1$

Target dataset $\mathscr{D}_t = \{x_j^t\}_{j=1}^{N_t}$

**Output:**

Known target dataset $\mathscr{D}_t^{knw} = \{x_j^{t,knw}\}_{j=1}^{N_{t,knw}}$

Unknown target dataset $\mathscr{D}_t^{unk} = \{x_j^{t,unk}\}_{j=1}^{N_{t,unk}}$

**procedure** GETROTATIONSCORE($z$,$i$)
  $\mathbf{o} = zeros(|\mathscr{C}_s|)$ # vector of $|\mathscr{C}_s|$ zeros
  **for each** $k$ **in** $\{1,...,|\mathscr{C}_s|\}$ **do**
    $[\mathbf{o}]_k = [z]_{k\times 4+i}$ # $[\mathbf{a}]_b$ indicated the $b$-th element of vector $\mathbf{a}$
  **return o**
**procedure** GETENTROPYSCORE($z$)
  **return** $z \cdot \log(z)/\log(|\mathscr{C}_s|)$
**procedure** GETNORMALITYSCORE($E$,$R_1$,$\mathscr{D}_t$)
  **for each** $x_j^t$ **in** $\mathscr{D}_t$ **do**
    Initialize: $h = \{\}$, $\mathbf{o} = zeros(|\mathscr{C}_s|)$
    **for each** $i$ **in** $\{1,...,4\}$ **do**
      $\tilde{x}_j = rot90(x_j, i)$
      $z_j = \text{softmax}\big(R_1(E(x_j)||E(\tilde{x}_j))\big)$
      $h \leftarrow getEntropyScore(z_j)$
      $o \mathrel{+}= getRotationScore(z_j, i)$ # element-wise sum of vectors
    $h = mean(h)$
    $o = max(\mathbf{o})$
    $\mathscr{N} \leftarrow \eta_j = max(o, 1-h)$
  **return** $\mathscr{N}$

**procedure** MAIN( )
  Initialize: $\mathscr{D}_t^{knw} = \{\}$, $\mathscr{D}_t^{unk} = \{\}$
  $\mathscr{A} = getNormalityScore(E, R_1, \mathscr{D}_t)$
  **for each** $(x_j, \eta_j)$ **in** $(\mathscr{D}_t, \mathscr{N})$ **do**
    **if** $\eta_j \geq mean(\mathscr{N})$ **then**
      $\mathscr{D}_t^{knw} \leftarrow x_j$
    **else**
      $\mathscr{D}_t^{unk} \leftarrow x_j$

### A.1.5 Other Self-Supervised Tasks and Further Ablation

Our goal is to demonstrate that it is possible to address both sub-tasks of OSDA (known/unknown separation and domain alignment), using a single self-supervised task. Based on previous research on CSDA [191, 33] and anomaly detection [137, 168, 297], rotation classification clearly emerges as the most reliable candidate for our purpose.

To support our claim, we did additional experiments on Office-31 (ResNet-50) using different self-supervised tasks. We used translation classification for anomaly detection (Stage I) based on [137] and jigsaw puzzle for domain alignment (Stage II) based on [33]. Table A.2 shows the results of these experiments: rotation recognition outperforms both the alternative tasks and the combination of them.

We also confirm the crucial contribution of the multi-rotation task instead of the standard 4-Class task in Stage I. The results are shown in Table A.2, which shows that the AUC-ROC decreases by a remarkable 28.6% when using the standard rotation task. For consistency, we kept the anchor (relative rotation) in the 4-Class experiment as well.

Since using the entropy loss in the object classification process across domains is standard practice, we did not include an ablation for Stage II of ROS on this term in Section 4.2. For completeness we present it here. We set $\lambda_{2,1} = 0$ including the results in Table A.2: as expected, without the entropy loss the performance drop on average of 1.9 percentage points, confirming that the entropy helps to adapt with a more evident effect in case of large domain gaps (e.g. A$\rightarrow$W, W$\rightarrow$A). Moreover, the center loss is not as crucial for Stage II as it is for Stage I and would introduce an additional hyper-parameter. The results in Table A.2 suggest that adding the center loss to Stage II may even result in a slight decrease in performance.

### A.1.6 Normality Score Pseudo-code

We summarized in Algorithm 2 the procedure used to calculate the normality score at the end of Stage I of ROS.

# Appendix B

# Improved Supervised models for Cross-Domain Learning

## B.1 HyMOS

### B.1.1 Qualitative Analysis

The t-sne plots in Figure B.1 show the distribution of source and target data in the feature space (i.e., the output of the contrastive head). We specifically examine the Ar, Pr, Rw $\rightarrow$ Cl case of the Office-Home dataset: the red dots indicate the source domain, the blue dots the known samples of the target domain, and the green dots the unknown samples. We take three snapshots of the data on the hyperspherical embedding: initially when the backbone network is inherited from SupClr [41] pre-trained on ImageNet, right before the first *break-point* (i.e. before self-training), and at the end of the training process. By observing the intermediate plot, it's clear that source balancing and style transfer contribute to a good alignment of most of the known target clusters (blue) with their respective source known clusters (red). The last plot shows that the self-training process further enhances the alignment, while the unknown samples (represented by green dots) remain located in the regions between the clusters.

We randomly zoomed on a known sample (the bike) and an unknown sample (the speaker) to observe how their positions changed during training. The bike sample moved from an isolated region with high-class confusion among its top five

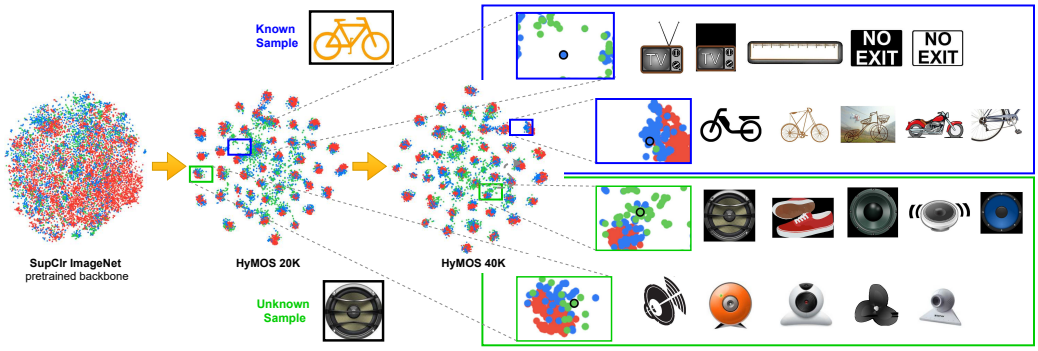



Fig. B.1 Qualitative analysis of the Ar, Pr, Rw → Cl case in the Office-Home dataset. The red dots represent the source domain, the blue dots represent the known samples of the target domain, and green dots represent the unknown samples of the target domain. HyMOS 20k: source balancing and style transfer already favor a good alignment of most known target classes with their respective source known clusters. HyMOS 40k: self-training further move the target known samples towards the respective source clusters, while the unknown samples remain in the regions among the clusters. The zooms demonstrate how the neighborhood of a known target sample (bike) and an unknown target sample (speaker) changes during training.

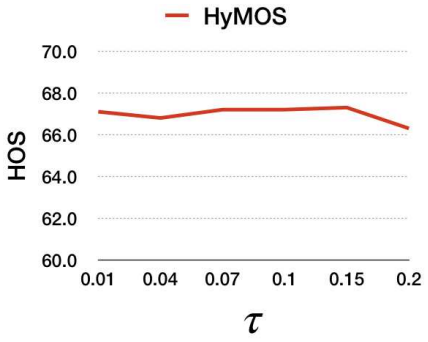



Fig. B.2 Sensitivity analysis for the temperature value $\tau$ on Office-Home.

neighbors towards the correct bike class. The speaker starts from a neighborhood populated by several samples of classes webcam and fan and finally appears in a different region shared mostly by other instances of the class speaker.

## B.1.2 Further experiments

**Robustness to temperature variation** The contrastive loss (Section 5.1 Eq. 5.1) uses a fixed temperature $\tau$ value of 0.07, as suggested in [169]. Figure B.2 show that even when varying $\tau$, the results remain stable and consistently higher than ROS (65.3).

**Algorithm 3** HyMOS evaluation procedure

**Input:** $\mathscr{T}$; trained $Enc$ and $Proj$
**Output:** Predictions on $\mathscr{T}$

```
procedure FINALEVAL()
    α ← (Section 4.2 Eq. (4))
    for each x_t in 𝒯 do
        z^t = Proj(Enc(x^t))
        h_{y^s} ← nearest prototype to z^t
        if d_{h_{y^s}}(z^t) < α then
            ŷ^t = y^s
        else
            ŷ^t = unknown
procedure MAIN()
    finalEval()
```

### B.1.3 Implementation Details

For our implementation of HyMOS, we used a ResNet-50 [219] backbone for the *encoder*, along with two fully connected layers of dimension 2048 and 128 for the *contrastive head*. The network was trained using the contrastive loss (refer to Eq. 5.1 in Section 5.1), with a fixed $\tau$ value of 0.07 as suggested in [169]. Our distance-based classifier lives in the hyperspherical space generated by the model, whose dimension is not constrained by the number of classes. Therefore, the architecture of the model is the same across all of our experiments.

To initialize the backbone network, we used the SupClr model pre-trained on ImageNet [41]. HyMOS was trained for 40k iterations using a balanced data mini-batch containing one sample from each class of every source domain.

We used a linear warm-up schedule that gradually increased the learning rate from 0 to 0.05, reaching the maximum value at iteration 2500. Subsequently, we applied a cosine annealing schedule that gradually decreased the learning rate back to 0 by the end of the training (iteration 40k). We used the LARS optimizer [298] with a momentum of 0.9 and weight decay of $10^{-6}$. For the first 20k iterations, we trained only on the source data, using the target data exclusively for style transfer-based data augmentation in the supervised contrastive learning objective. After this, we then perform an evaluation step, referred to as the self-training *break-point*, to begin including confident known target samples in the learning objective. We repeated the *break-point* eval step every 5K iterations until the end of the training.

To perform style transfer data augmentation, we used the standard VGG19-based AdaIN model with default hyperparameters [265]. This model was trained using content data from the available source domains, with target samples as style data.

Regarding instance transformations, we utilized the same data augmentations originally proposed for SimClr [40] and extended them with style transfer. Specifically, we used random resized crop with scale in the range $0.08, 1$ and random horizontal flip. For the source images, style transfer was applied with probability $p = 0.5$, while non-stylized images were transformed using color jittering with probability $p = 0.8$ and grayscale with probability $p = 0.2$.

In Algorithm 3 is summarized the final evaluation procedure of HyMOS.

## B.2 ReSeND

### B.2.1 Implementation details

We used ResNet-18 [219] pretrained on ImageNet1k [214] as feature extractor $f_\theta$, removing the original final classification layer. The relational module $r_\gamma$ has the same structure of the transformer in ViT [146]: we use 4 multi-head self-attention encoder blocks, which balances performance and time complexity given that the number of blocks affects the total number of learnable parameters in the network.

Before entering the transformer, the features extracted by the backbone go through an FC projection layer. The input sequence for the transformer is obtained by concatenating the representations of a pair of samples with the learnable label token $[z_l, z_i, z_j]$. The resulting output token $v_l$ is then passed through the final FC layer, which represents the regression head $c_\delta$.

The transformer procedure is summarized in the following equations:

$$z^0 = [z_l; z_i; z_j] \tag{B.1}$$

$$\tilde{z}^b = \text{MSA}(\text{LN}(z^{b-1})) + z^{b-1}, \qquad b = 1 \dots B \tag{B.2}$$

$$z^b = \text{MLP}(\text{LN}(\tilde{z}^b)) + \tilde{z}^b, \qquad b = 1 \dots B \tag{B.3}$$

$$v_l = \text{LN}(z_l^B)\ . \tag{B.4}$$

Our network is trained end-to-end on ImageNet1k using the MSE loss (as defined in Eq. 5.6 in Section 5.2) applied to the output of the regression head. The training procedure consists of 13k iterations with a batch size of 4096, where each element of the batch is an image pair. The learning rate starts with a linear warm-up for 500 iterations and is then fixed at 0.008. We use LARS optimizer [298] with the momentum 0.9 and weight decay $5 \cdot 10^{-5}$. We build image pairs for training by selecting an image from the dataset as anchor and pairing it with another sample that has the same label (*positive* pair), or a different label (*negative* pair). Our experimental results are averaged over three runs. The training and evaluation procedures for ReSeND are summarized in Algorithm 4 and 5.

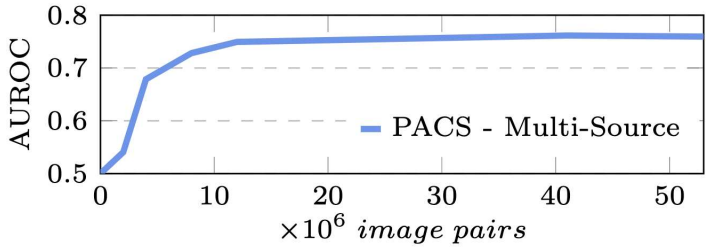



Fig. B.3 Performance trend increasing the number of image pairs

### B.2.2 Further Analysis

**Number of image pairs**. Our training objective relies on the use of image pairs that are randomly built by combining samples from the training dataset. Although the number of image pairs that can be created from the ImageNet1k dataset is quite large (approximately $820 \times 10^9$), we demonstrate in Fig. B.3 that ReSeND reaches convergence after processing only a relatively small fraction of them.

**Algorithm 4** ReSeND train procedure

**Input:** $\mathscr{S}$, $\mathscr{T}$, $f_\theta : \mathbb{R}^{H\times W\times C} \to \mathbb{R}^d$, $r_\gamma$, $c_\delta$

```
procedure CREATE_PAIRS(𝒮)
    pairs = []
    for each (x^s, y^s) in {𝒮} do
        pairs.append((rand_same_class(y^s), x^s, 1))
        pairs.append((rand_diff_class(y^s), x^s, 0))
    return pairs
procedure MAIN()
    for epoch in range(n_epochs) do
        pairs = create_pairs(𝒮)
        shuffle(pairs)
        for iter in range(iters_epoch) do
            pairs_batch = next_batch(pairs)
            x1, x2, labels = pairs_batch
            z1 = f_θ(x1)
            z2 = f_θ(x2)
            feats_pairs = (z1, z2)
            predictions = c_δ(r_γ(feats_pairs))
            MSE_loss = ℒ(predictions, labels)        ▷ Eq. 5.6
            Update θ, γ, δ ← ∇MSE_loss
```

**Algorithm 5** ReSeND eval procedure

**Input:** $\mathscr{S}$, $\mathscr{T}$, $f_\theta : \mathbb{R}^{H\times W\times C} \to \mathbb{R}^d$, $r_\gamma$, $c_\delta$

**procedure** COMPUTE_PROTOTYPES($\mathscr{S}$)

$prototypes = zeros((|\mathscr{Y}_s|,))$

$counters = zeros(|\mathscr{Y}_s|)$

**for each** $(x^s, y^s)$ **in** $\{\mathscr{S}\}$ **do**

$z^s = f_\theta(x^s)$

$prototypes[y^s] += z^s$

$counters[y^s] += 1$

**for** $i$ **in** $range(|\mathscr{Y}_s|)$ **do**

$prototypes[i] /= counters[i]$

**return** $prototypes$

**procedure** MAIN()

$normality\_scores = []$

$prototypes = compute\_prototypes(\mathscr{S})$

**for each** $x^t$ **in** $\{\mathscr{T}\}$ **do**

$z^t = f_\theta(x^t)$

$pairs = (prototypes, z^t.repeat())$

$predictions = c_\delta(r_\gamma(pairs))$

$score = max(softmax(predictions))$

$normality\_scores.append(score)$

# Appendix C

# Project Contributions and Computational Resources Support

This thesis contributes to RoboExNovo (ERC grant 637076), ACROSSING (EU H2020 grant 676157) and Elise (EU H2020) projects. It has been also partially supported by the Blanceflor Foundation (travel grant).

Computational resources were provided by the Italian Institute of Technology (IIT) (HPC infrastructure) and Politecnico di Torino (HPC@PoliTo). We also acknowledge the CINECA award IsC94 TrOSDG under the ISCRA initiative and NVIDIA for the GPU received thanks to the Academic Hardware Grant.