Rochester Institute of Technology
RIT Digital Institutional Repository

Theses

4-2024

Transfer learning across domains and sensing modalities

Chowdhury Sadman Jahan
sj4654@rit.edu

Follow this and additional works at: https://repository.rit.edu/theses

Recommended Citation
Jahan, Chowdhury Sadman, "Transfer learning across domains and sensing modalities" (2024). Thesis. Rochester Institute of Technology. Accessed from

This Dissertation is brought to you for free and open access by the RIT Libraries. For more information, please contact repository@rit.edu.

Transfer learning across domains and sensing modalities

by

Chowdhury Sadman Jahan

A dissertation submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy
in Imaging Science

Chester F. Carlson Center for
Imaging Science

Rochester Institute of Technology
Rochester, New York
April, 2024

Transfer learning across domains and sensing modalities

by

Chowdhury Sadman Jahan

Committee Approval:

We, the undersigned committee members, certify that we have advised and/or supervised the candidate on the work described in this dissertation. We further certify that we have reviewed the dissertation manuscript and approve it in partial fulfillment of the requirements of the degree of Doctor of Philosophy in Imaging Science.

Dr. Andreas Savakis — Date
Dissertation Advisor

Dr. Carl Salvaggio — Date
Dissertation Committee Member

Dr. Jan van Aardt — Date
Dissertation Committee Member

Dr. Michael Murdoch — Date
Dissertation Defense Chairperson

Certified by:

Dr. Charles Bachmann — Date
Ph.D. Program Director, Imaging Science

©[2024] [Chowdhury Sadman Jahan]
All rights reserved.

Transfer learning across domains and sensing modalities

by

Chowdhury Sadman Jahan

Submitted to the
Chester F. Carlson Center for Imaging Science
Ph.D. Program in Imaging Science
in partial fulfillment of the requirements for the
Doctor of Philosophy Degree
at the Rochester Institute of Technology

Abstract

Transfer learning facilitates the training of a deep learning (DL) model with limited or no labeled data, by initializing the network parameters using a similar model already trained on a different but related dataset or task. This dissertation examines two special cases of transfer learning for image classification tasks: cross-modal supervised learning, and cross-domain unsupervised adaptation. This dissertation proposes to apply cross-modal transfer learning to guide the training process of a DL model on Synthetic Aperture Radar (SAR) images via knowledge distillation from a DL model trained on corresponding electro-optical (EO) images. Furthermore, this approach explores class-balanced sampling strategies and multi-stage training procedures to account for the high class-imbalance encountered in a real-world SAR image dataset.

When models trained in one domain (source) are deployed in a new environment (target), they may encounter performance degradation due to the data distribution shift between the source and the target. Domain adaptation (DA) aims to address this limitation by aligning the source domain features with those extracted from the target domain. Drawing inspiration from continual learning, we refine source-free continual unsupervised domain adaptation methods ConDA and UCL-GV, which are buffer-fed networks that adapt to the continually incoming small batches of unlabelled target data. Our models outperform state-of-the-art (SOTA) continual DA models on both static, and dynamic (gradually changing) target domains. We further introduce new synthetic aerial datasets under gradually degrading weather conditions, and propose techniques to improve training stability of continual DA methods.

Recent tools for the commercialization of DL models have sparked concerns about protecting proprietary DL technologies during end-user deployment. We explore black-box domain adaptation (BBDA) as a means to mitigate these concerns. We propose a curriculum-guided domain adaptation method called CABB that splits the target data into clean and noisy subsets via pseudolabel distribution modeling, and progressively adapts to the reliable and clean pseudolabels first, and then to the noisy pseudolabels later. Our method

outperforms existing BBDA models by up to 9.3% across several popular DA datasets, and is on par with white-box DA models.

All the object categories in the source and the target domains may not necessarily fully overlap, and the target domain may contain samples from novel classes that are absent in the source domain. We introduce Unknown Sample Discovery (USD) as a source-free open set domain adaptation (SF-OSDA) method that also utilizes pseudolabel distribution modeling to conduct known-unknown target sample separation. USD operates within a teacher-student framework using co-training and temporal consistency between the teacher and the student models, thereby significantly reducing error accumulation resulting from imperfect known-unknown sample separation. Empirical results show that USD is superior to existing SF-OSDA methods by as much as $\sim$20% in terms of prediction accuracy.

Acknowledgments

I would like to express my heartfelt thanks to my PhD advisor Dr. Andreas Savakis for his constant support and belief in me. Without his wisdom and guidance, I could not have completed my dissertation. He gave me full freedom to pursue any and all of my topics of interest, but steered my *ship* any time I got derailed. He believed in me when I doubted myself. His words of reassurance were the motivations at the end of a bad day, and there were many, to start the next day afresh.

I would like to thanks my committee members Dr. Carl Salvaggio and Dr. Jan van Aardt, and external chair Dr. Michael Murdoch for their support and critique of my research. I would also like to thank my mentor Dr. Bo Mu, Director of Algorithm Development at OmniVision Technologies, Inc. for his guidance during my internship at the company.

I would also like to thank the faculty at the Chester F. Carlson Center for Imaging Science. The well designed courses of Imaging Science truly gave me a strong foothold to pursue research in the multidisciplinary fields of imaging and computer vision. I would like to acknowledge the administrative staff at Imaging Science, and the support staff at the Vision and Image Processing lab of the Department of Computer Engineering. During the Covid pandemic, everyone provided me all the accommodations I asked for, and ensured that the transition from in-person to remote work was as smooth as possible.

I would like to thank my fellow labmates over the years: Abu, Navya, Bruno, Raaga, Christian, Udit, Divyansh, Mihir, Georgi, Navin, Rajat and Alec.

The ones who I am most grateful to are my wife, my parents, and my elder brother's family. They have been the source of my inspiration throughout this process. This journey would have been lonely and frustrating, if not for their constant emotional and mental support.

Finally, I would like to extend my acknowledgements to my classmates, my friends at RIT, and the whole RIT community. Research Computing at RIT deserves special recognition for making many of the resources needed for this dissertation available.

This dissertation is dedicated to my wife, my parents, and my brother.

Publications

Chowdhury Sadman Jahan, and Andreas Savakis. "Unknown Sample Discovery for Source Free Open Set Domain Adaptation". Accepted to the 1st Workshop on Test-Time Adaptation: Model, Adapt Thyself! (MAT) at The IEEE / CVF Computer Vision and Pattern Recognition Conference (CVPR) 2024.

Chowdhury Sadman Jahan, and Andreas Savakis. "Continual Domain Adaptation on Aerial Images under Gradually Degrading Weather". Journal of Applied Remote Sensing 18, no. 1 (2024): 016504-016504.

Chowdhury Sadman Jahan, and Andreas Savakis. "Curriculum Guided Domain Adaptation in the Dark." IEEE Transactions on Artificial Intelligence (2023).

Abu Md Niamul Taufique*, **Chowdhury Sadman Jahan***, and Andreas Savakis. "Continual Unsupervised Domain Adaptation in Data-Constrained Environments". IEEE Transactions on Artificial Intelligence, vol. 5, no. 1, pp. 167-178, Jan. 2024. (*equal contribution)

Rajat Sahay, Georgi Thomas, **Chowdhury Sadman Jahan**, Mihir Manjrekar, Dan Popp and Andreas Savakis. "On the Importance of Attention and Augmentations for Hypothesis Transfer in Domain Adaptation and Generalization". Sensors 23, no. 20 (2023): 8409.

Chowdhury Sadman Jahan, and Andreas Savakis. "Balanced sampling meets imbalanced datasets for SAR image classification." In Geospatial Informatics XIII, vol. 12525, pp. 37-45. SPIE, 2023.

Chowdhury Sadman Jahan, Andreas Savakis, and Erik Blasch. "Sar image classification with knowledge distillation and class balancing for long-tailed distributions." In 2022 IEEE 14th Image, Video, and Multidimensional Signal Processing Workshop (IVMSP), pp. 1-5. IEEE, 2022.

Chowdhury Sadman Jahan, Andreas Savakis, and Erik Blasch. "Cross-modal knowledge distillation in deep networks for SAR image classification." In Geospatial Informatics XII, vol. 12099, pp. 20-27. SPIE, 2022.

Abu Md Niamul Taufique, **Chowdhury Sadman Jahan**, and Andreas Savakis. "Unsupervised continual learning for gradually varying domains." In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 3740-3750. 2022.

Navya Nagananda, Abu Md Niamul Taufique, Raaga Madappa, **Chowdhury Sadman Jahan**, Breton Minnehan, Todd Rovito, and Andreas Savakis. "Benchmarking domain adaptation methods on aerial datasets." Sensors 21, no. 23 (2021): 8070.

Contents

1 Introduction **1**

1.1 Objectives 3

1.2 Dissertation outline 4

1.2.1 Chapter 2: Background 4

1.2.2 Chapter 3: EO guided SAR image classification 4

1.2.3 Chapter 4: Continual unsupervised domain adaptation 4

1.2.4 Chapter 5: Black-box domain adaptation 5

1.2.5 Chapter 6: Source-free open-set domain adaptation 5

1.2.6 Chapter 7: Conclusion and future work 5

2 Background **6**

2.1 Perceptron 6

2.2 Multi-layer Perceptron 9

2.3 Convolutional Neural Network (CNN) 10

2.4 Transformers in vision 12

3 EO-guided SAR image classification **14**

3.1 Introduction 15

3.2 Related Work 17

3.2.1 SAR image classification 17

3.2.2 Knowledge distillation 18

3.3 Methodology 19

3.3.1 Stage 1: EO training 19

3.3.2 Stage 2: SAR training with knowledge distillation 20

3.3.3 Alternate Stage 2: Class balanced SAR training 21

3.3.4 Optional Stage 3: SAR training with class balancing 23

3.4 Datasets and Experiments 24

3.4.1 Datasets 24

3.4.2 Implementation details 25
3.5 Results 25
3.6 Conclusion 29

4 Continual unsupervised domain adaptation 31
4.1 Introduction 32
4.2 Related Work 36
4.2.1 Domain adaptation 36
4.2.2 Continual learning 37
4.2.3 Continual domain adaptation 38
4.2.4 Domain adaptation on aerial imagery 39
4.3 ConDA 40
4.3.1 Method 40
4.3.2 Experimental Setup 45
4.3.3 Results 50
4.4 UCL-GV 55
4.4.1 Method 55
4.4.2 Datasets and experiments 61
4.4.3 Results 63
4.5 Continual domain adaptation on aerial images under gradually degrading weather 66
4.5.1 Benchmark Datasets 66
4.5.2 Implementation details 67
4.5.3 Results and discussion 68
4.6 Conclusions 73

5 Curriculum-Guided Domain Adaptation in the Dark 76
5.1 Introduction 77
5.2 Related Work 78
5.2.1 Unsupervised domain adaptation 78
5.2.2 Source-free domain adaptation 79
5.2.3 Black box domain adaptation 80
5.2.4 Curriculum learning 81
5.2.5 CABB comparison with other BBDA methods 81
5.3 Methodology 82
5.3.1 Clean-noisy separation 83
5.3.2 Ensemble based pseudolabeling 84

5.3.3 Curriculum-guided noisy learning 85
5.4 Experimental setup 87
5.4.1 Datasets 87
5.4.2 Implementation details 89
5.5 Results 91
5.5.1 Overall evaluation 91
5.5.2 Ablation study 92
5.6 Conclusion 93

6 Unknown Sample Discovery for Source Free Open Set Domain Adaptation 94
6.1 Introduction 95
6.2 Related Work 97
6.2.1 Unsupervised domain adaptation 97
6.2.2 Source free domain adaptation 97
6.2.3 Open set domain adaptation 98
6.3 Method 99
6.3.1 Known-unknown sample separation 99
6.3.2 Teacher-student co-training and regularization 102
6.4 Experimental Setup 104
6.4.1 Datasets 104
6.4.2 Implementation details 106
6.4.3 Evaluation metrics 106
6.5 Results 107
6.5.1 Overall results 107
6.5.2 Ablation study 107
6.6 Conclusion 111

7 Conclusion and Future Work 112

List of Figures

2.1 Rosenblatt's perceptron proposed in 1958 [174]. 6
2.2 Multi layered perceptron (MLP) network. 9

3.1 Representative examples from the coupled EO-SAR dataset. The top row shows EO images and the bottom row shows SAR images. Each column represents one class. From left to right, the classes are: sedan, SUV, van, motorcycle, flatbed truck, and pickup truck with trailer. 16

3.2 Proposed framework for cross-modal training with knowledge distillation. In stage 1: EO training, only the top branch is trained and the bottom branch along with KD block is removed. In Stage 2: SAR training with knowledge distillation, the top branch is locked and the bottom branch is trained for SAR image classification with knowledge distillation from the EO network. In an optional Stage 3: SAR training with class balancing, only the bottom branch is trained and the top branch along with KD block is removed. 20

3.3 Proposed framework for Balanced Cross-KD during SAR training, where the EO network is locked and the SAR network undergoes training. The Sampling Block selects samples either based on instance sampling or class-balanced sampling strategy. The laters FC, BN and WN represent Fully Connected layer, Batch Normalization layer and Weight Normalization layer, respectively. 22

3.4 Examples of correct predictions after training to illustrate the effects transfer learning and knowledge distillation in Stage 2 for **Cross-KD** and subsequent class balanced training in Stage 3 for **Cross-KD+**. Red labels indicate incorrect prediction, green labels indicate correct prediction, and blue labels indicate ground truth. 28

4.1 Continual DA paradigm where initial training is performed with source domain labeled data and the trained model is deployed in the target domain. During deployment, unlabelled target domain data are received in streaming batches and the model is continuously adapted with each new batch of target data. 33

4.2 Proposed ConDA framework adapting on target domain data that arrive in small batches. A subset of the samples that are already seen by the network are stored in a buffer for replay with the incoming batches. The buffer manager is responsible for selecting the samples that populate the buffer. The incoming target samples are mixed with the current buffer samples and sent to the network for adaptation. 40

4.3 Continual adaptation for multi-target domains. For demonstration purpose, we consider 5 categories from the Office-Home dataset. The network processes each batch $\mathcal{X}_{t_i}$ only once along with the replayed buffer samples. When samples from target domain $\mathcal{D}_{t_1}$ end, batch samples from new domain $\mathcal{D}_{t_2}$ start to be fetched by the network and the same process continues until the last domain $\mathcal{D}_{t_\tau}$ is fetched. 42

4.6 Feature visualization plots for 10 classes before and after continual adaptation from Real-World (Source) to Product (Target) from the Office-Home dataset. (a) t-SNE plot for source-trained model on Real-World before adaptation, and (b) t-SNE plot for the target-adapted model. 54
4.7 Proposed paradigm of Unsupervised Continual Learning for Gradually Varying domain adaptation (UCL-GV). The network is trained on a source domain and continually adapts using small incoming batches of data from a gradually varying target domain that has no labels. 55
4.8 Proposed UCL-GV method for unsupervised continual learning for domain adaptation in gradually varying domains. 56
4.9 Application of contrastive loss using the buffer prototypes (cluster centers) and the batch samples, for better clustering. 58
4.10 Rotating MNIST dataset. 61
4.11 CORe50 [137] dataset in a gradual time varying setting. 62
4.12 Performance of UCL-GV on the rotating MNIST target domain $\mathcal{D}_{tar}$ during continual adaptation on each incremental batch from the combined intermediate and target domain $\mathcal{D}_t$. 64
4.14 AID-CC dataset with cloud cover degradation. (1) is the source domain, (8) is the target domain, and (2-7) are progressively degrading intermediate domains. 66
4.15 UCM-CC dataset with cloud cover degradation. (1) is the source domain, (8) is the target domain, and (2-7) are progressively degrading intermediate domains. 67
4.16 AID-SF dataset with snowfall degradation. (1) is the source domain, (6) is the target domain, and (2-5) are progressively degrading intermediate domains. 68
4.17 UCM-SF dataset with snowfall degradation. (1) is the source domain, (6) is the target domain, and (2-5) are progressively degrading intermediate domains. 69
4.18 Effect of gradient normalization on adaptation stability for the continual models ConDA and UCL-GV. 70
4.19 Continual models ConDA and UCL-GV with Swin backbone, showing increase in adaptation stability and final accuracy due to gradient normalization. 74

5.1 Overview of BBDA, where the source model parameters are not available during adaptation. The source model may only be accessed as a black box to generate pseudolabels for the unlabeled target data. These pseudolabels may be used to adapt the target model on the target domain without true labels. 77

5.2 UDA pipeline in CABB. The target data is fed to the source model f_s and the knowledge generated from f_s is transferred to both target branches f_{t_1} and f_{t_2}. The source predicted pseudolabels are also used to calculate JSD and produce clean-noisy sample sets. In subsequent co-training of f_{t_1} and f_{t_2}, the samples sets created by one branch are used to update the other branch, using curriculum guided losses to progressively adapt to clean samples first, and the noisy samples later. 80
5.3 Ensemble-based pseudolabeling in CABB. Each sample is augmented to produce 6 different views that are fed through both branches f_{t_1} and f_{t_2} to create a total of 12 output predictions, which are then averaged to produce the soft pseudolabel for co-training f_{t_1} and f_{t_2}. 84
5.4 Accuracy on the clean sample set achieved via clean-noisy sample separation using low JSD (CABB) vs low CE (BETA), after distillation from the source teacher at the first epoch. 88

6.1 Different domain adaptation settings depending on the classes present in the source and target domains. For open-set DA, the classes novel to the target domain are grouped into a single unknown class during adaptation. 95
6.2 Pseudolabel generation for the target samples and known-unknown sample separation based on JSD 100
6.3 Adaptation process for USD using co-training. The student model receives pseudolabels for the target samples (see Figure 6.2) and is optimized using a combination of triplet, weak-strong consistency, information maximization (IM) and cross-entropy losses. The teacher model is updated via exponential moving averages (EMA) at the end of each epoch. 100
6.4 Impact of JSD threshold δ_t on HOS for Office dataset. 108
6.5 Impact of co-training on reducing error accumulation during adaptation on Office-Home dataset. 108

List of Tables

3.1 Sample distribution across the ten classes in the dataset. 24
3.2 Class-wise accuracy for the EO model after training on the EO images only. 25
3.3 Class-wise accuracy for different forms of our model. Cross-KD and Balanced Cross-KD are 2 stage processes, while CrossKD+ is a 3 stage process. 26

3.4 Ablation study for Stage 2 SAR model training in CrossKD. "TL" means transfer learning to initiate the student model with teacher model parameters. "KD" refers to knowledge distillation from EO teacher to SAR student model. 27
3.5 Balanced Cross-KD results for various configurations. "Base" refers to the mixture of class balanced sampling and instance sampling, and trained with knowledge distillation and cross-entropy losses only. 27
3.6 Ablation study for the loss functions in the third stage for CrossKD+ SAR training with class-balanced loader. 28

4.1 Mean accuracy of adaptation using the Office-31 dataset. The top part of the table shows results of traditional DA methods using the full target dataset. The bottom part of the table shows results for ConDA and SHOT using the continual setting. The ConDA experiments are performed with a continual batch size of 32 and buffer size of 124 (four samples per class). 48
4.2 Mean accuracy of adaptation using the Office-Home dataset. The top part of the table shows results of traditional DA methods using the full target dataset. The bottom part of the table shows results for ConDA and SHOT using the continual setting. The ConDA experiments are performed with a continual batch size of 128 and buffer size of 520 (eight samples per class). 49
4.3 Mean per class accuracy of adaptation using the VisDA-C dataset. The top part of the table shows results of traditional DA methods using the full target dataset. The bottom part of the table shows results for ConDA and SHOT using the continual setting. The ConDA experiments are performed with a continual batch size of 32 and a buffer size of 96 (eight samples per class). 49
4.4 Multi domain adaptation results for Office-Caltech dataset. The benchmark results are obtained from [170]. The continual experiments are done with an incoming batch size of 32 samples and a buffer size of 40 samples per domain (four samples per class per domain). 51
4.5 Multi domain adaptation results for Office-Home dataset. The benchmark results are obtained from [170]. The continual experiments are done with an incoming batch size of 128 samples and a buffer size of 260 samples per domain (four samples per class per domain). 51
4.6 Ablation study of ConDA on the effects of using a buffer and $\mathcal{L}_{eqdiv}$ using the Office-31 dataset. The ablation study for ConDA had a continual batch size of 32 and a buffer size of 124 (four samples per class). 51
4.7 Percent accuracy of UCL-GV and comparison with other methods. The experiments on rotating MNIST are performed with a continual batch size of 128 and buffer size of 512. CORe50 experiments are performed with a continual batch size of 16 and buffer size of 32. All evaluations are conducted on the target domain $\mathcal{D}_{tar}$. 62

4.8 Ablation studies of UCL-GV on the rotating MNIST dataset. Experiments are performed with a continual batch size of 128 and buffer size of 512. 65
4.9 Initial results on the gradually degrading AID and UCM datasets with ResNet-50 backbone on the final target domain. Source-trained refers to the model trained on the source data only, without any adaptation. The top accuracy is in bold and the second best is underlined. 68
4.10 Results on the gradually degrading AID and UCM datasets, using ResNet-50 backbone, gradient normalization, and initial learning rate of $\eta_0 = 0.002$ and $\eta_0 = 0.02$. Source-trained refers to the model trained on the source data only, without any continual target adaptation over the intermediate and target domains. The top accuracy is in bold and the second best is underlined. 71
4.11 Results on AID-CC, and AID-SF with ResNet-50, ViT-B, and Swin-B backbones, at initial learning rate of 0.002. Source-trained method refers to the model trained on the source data only, without any continual target adaptation over the intermediate and target domains. The top accuracy is in bold and the second best is underlined. respectively. 72
4.12 Results on UCM-CC, and UCM-SF with ResNet-50, ViT-B, and Swin-B backbones, at initial learning rate of 0.02. Source-trained method refers to the model trained on the source data only, without any continual target adaptation over the intermediate and target domains. The top accuracy is in bold and the second best is underlined. respectively. 73

5.1 Methodology comparison between CABB and existing BBDA methods. 78
5.2 Mean accuracy on the Office31. 'SF' refers to source-free and 'BB' means black-box. The top performing results among the BBDA methods are in bold letters. 88
5.3 Mean accuracy on the Office-Home dataset. 'SF' refers to source-free and 'BB' means black-box. The top performing results among the BBDA methods are in bold letters. 89
5.4 Mean per-class accuracy on the VisDA-C dataset. 'SF' refers to source-free and 'BB' means black-box. The top performing results among the BBDA methods are in bold letters. 90
5.5 Performance evaluation of curriculum adaptation involving different parts of CABB on the VisDA-C dataset. The 'tick' marks mean the part is present in the model, and the 'cross' mark means that part is absent. When curriculum is absent and $\mathcal{L}_{tn}$ is present, γ_n is set to 0.5. $*$ refers to replacement of our active-passive loss with standard cross-entropy loss for noisy samples. 90
5.6 Performance evaluation of CABB with different kinds of pseudolabeling schemes, and the impact of dual-branched co-training on error accumulation for VisDA-C dataset. 90
5.7 Performance evaluation of CABB with various values for the hyperparameter α for VisDA-C dataset. 91

6.1 Evaluation of USD on Office-31 dataset. * are results computed for the methods using publicly released code. 105
6.2 Evaluation of USD on Office-Home and VisDA-C datasets. * are results computed for the methods using publicly released code. 105
6.3 Evaluation of separation criterion and distribution modeling for known-unknown sample separation in USD on Office dataset. 109
6.4 Ablation study on the objective function, and co-training for USD on Office-Home dataset. 109
6.5 Ablation study on the pseudolabeling scheme for USD on Office-Home dataset. 109

Chapter 1

Introduction

Deep learning has come a long way since the first mathematical model of a neural network was devised by Warren McCulloch and Walter Pitts [148]. Backpropagation [129] is the driving force behind the effective deep learning of today. However, Henry J. Kelley derived the basics of continuous backpropagation not in the context of deep learning, but in the context of control theory in 1960 [96]. Building upon Rosenblatt's single layered artificial neural network [174], Fukusima developed the first convolutional neural network using convolutional and pooling layers [44] in 1982. Yann LeCun married backpropagation with convolutional neural networks in 1989 for handwritten digit recognition [112]. With the advent of graphical processing units (GPUs) around the turn of the century, deep learning based solutions were starting to catch up with support vector machines (SVMs) and other hand-designed feature extractors. In 2009, the ImageNet [34] visual recognition challenge was launched with 14 million labeled images, the ideal *grazing ground* of big data for deep artificial neural networks. Three years later in 2012, Alex Krizhevsky *et al.* introduced a convolutional neural network (CNN) called AlexNet [104], and became champions of the ImageNet visual recognition challenge [34] in 2012, beating the runners-up contestants by more than 10 percentage points, and thus propelling the deep learning revolution. Less than a decade later, the massive research and commercialization of deep learning (DL) has made artificial intelligence ubiquitous.

Successful application of DL, however, relies on the availability of large amounts of data for a model to train. But collection, processing, and annotation of large amounts of data for each task, modality or domain where a DL model is deployed to is not feasible. This has motivated the practice of transfer learning, where a model is trained with labelled source data for one task, and then subsequently trained with target data for another task. In the simplest form of transfer learning, the target data are labelled and belong to the same modality as the source data. In this dissertation, we consider two special forms of transfer learning: (i)

when a model trained on one modality of data is transferred to learn on labelled data from another modality, and (ii) when a model trained with data from one domain is transferred to adapt to unlabelled data from a different domain, and of the same modality.

For cross-modal transfer learning, we investigate aerial Synthetic Aperture Radar (SAR) image classification. SAR signals can pass through adverse weather related occlusions, such as cloud cover, and are thus preferred for aerial remote sensing using drones and other unmanned aerial vehicles (UAVs). SAR is also vital for national defence, as the occlusion penetrating property of SAR is heavily utilized for aerial surveillance. Deep learning solutions to SAR image classification and detection are an active area of research [7, 50, 157, 235, 236]. Several types of neural network architectures have been applied to address this issue, such as CNNs [23, 35], recurrent neural networks (RNNs) [7, 95], and autoencoders [45]. However, SAR data are low resolution, and noisy in nature. This significantly hampers supervised training methods to learn on SAR data. Some newer methods try to overcome the drawbacks of SAR data by fusing them with corresponding EO data [1, 78, 149, 150, 155]. Doing so, however, limits the applicability of such models under poor illumination (such as, inclement weather conditions) as aerial EO data may become practically unusable in such scenarios.

For cross-domain transfer learning, we investigate unsupervised domain adaptation (UDA). A domain gap or distribution shift is manifested when a deep network or model is trained with data from one domain/environment, and the model is deployed is a different domain/environment, resulting in significant performance degradation for the model. UDA attempts to mitigate the effects of this domain shift/gap by aligning the feature spaces of the source (training) domain and the target (deployment) domain. UDA has found its application in many DL tasks, such as classification [207], image segmentation [199], object detection [161], etc. Although many of the existing UDA methods work on the assumption that both the source and target data are available during the adaptation phase [31, 49, 201], concerns about the source data privacy led to another DA paradigm called source-free UDA [125, 221] that poses a more challenging problem by making the source data unavailable during adaptation. However, in a more practical scenario such as autonomous driving and robotics, the target data become available to the model for adaptation only in small batches at a time, and not in their entirety.In this case, the model needs to adapt *on the fly* or during test-time.

As more and more technology companies bring AI products and solutions to the consumer market, protecting their proprietary DL source models becomes a major concern, in addition to protecting the source data. Black-box domain adaptation (BBDA) [126, 219] aims to learn a target model with the target data, and their pseudolabels generated by a black-box source. By keeping the source model behind a veil, and never disclosing its parameters, BBDA helps companies protect their intellectual property from piracy. Due to lack of access to the source model, compared to standard UDA, performing BBDA is a more challenging

problem that has not been sufficiently explored.

Moreover, the target data may have extra/novel classes that are absent in the target domain. This setting is called open set domain adaptation (OSDA) [130, 183]. OSDA necessitates feature alignment in the common classes in the two domains, while increasing the inter-class feature distance between the known and unknown classes in the target domain.

This dissertation explores the two aforementioned forms of transfer learning across modalities and sensing modalities, and proposes several novel methods for cross-modal, and cross-domain transfer learning. Both forms are challenging, and deal with effectively utilizing predictive models learned from the annotated source data. In this dissertation, we propose novel ways to address several research questions related to practical scenarios, and applications of the two forms of transfer learning. The objectives, research outline, and results of our studies are discussed below.

1.1 Objectives

The broad objectives of this dissertation proposal are as follows.

1. Develop methods for synthetic aperture radar (SAR) image classification, using knowledge distillation from a model trained on corresponding electro-optical images, while addressing the class imbalance issue of SAR datasets. (Chapter 3)

2. Formulate novel models for continual domain adaptation for both static, and dynamic target domains, in addition to introducing new synthetic benchmark datasets for gradually degrading weather conditions in aerial images. (Chapter 4)

3. Design new methods for curriculum-guided domain adaptation without accessing the source data, and the source model, thus protecting privacy of the training data and model parameters. (Chapter 5)

4. Develop techniques for identifying novel class samples in source-free open-set domain adaptation, and formulate robust domain adaptation methods with increased inter-class discrimination between known and unknown classes. (Chapter 6)

1.2 Dissertation outline

In this dissertation, we present six chapters in total, with the current introduction chapter being the first chapter, and the sixth chapter being the one with the proposed timeline. The dissertation is outlined as follows.

1.2.1 Chapter 2: Background

In this chapter, we discuss some preliminary background topics of this proposal, including artificial neural networks, convolutional neural networks, and vision transformers.

1.2.2 Chapter 3: EO guided SAR image classification

In this chapter, we propose three DL training schemes for SAR image classification using coupled SAR-EO training images via knowledge distillation from an already learned EO image trained model. This chapter is based on three of our inter-related papers titled "Cross-modal knowledge distillation in deep networks for SAR image classification" [85], "SAR Image Classification with Knowledge Distillation and Class Balancing for Long-Tailed Distributions" [86], and "Balanced sampling meets imbalanced datasets for SAR image classification" [81] that appeared at SPIE DCS 2022, IVMSP 2022, and SPIE DCS 2023, respectively.

1.2.3 Chapter 4: Continual unsupervised domain adaptation

This chapter introduces the more pragmatic continual unsupervised DA paradigm where the target data are available for adaptation in small batches, instead of in their entirety during adaptation. We propose two related models that address continual UDA: one for static target domain, and other for gradually varying target domain. Our methods outperform existing UDA models in both scenarios. We also introduce four new benchmarking datasets for conducting continual UDA under gradually degrading weather conditions. This chapter is based on our IEEE Transactions of Artificial Intelligence paper titled "Continual Unsupervised Domain Adaptation in Data-Constrained Environments" [198], our CVPR 2022 paper titled "Unsupervised Continual Learning for Gradually Varying Domains" [197], and another paper published in the SPIE Journal of Applied Remote Sensing titled "Continual Domain Adaptation on Aerial Images under Gradually Degrading Weather" [84].

1.2.4 Chapter 5: Black-box domain adaptation

This chapter discusses a method to separate reliable and unreliable pseudolabels for domain adaptation by modelling the target data distribution. Specifically, a black-box domain adaptation model called *Curriculum Adaptation for Black-Box (CABB)* is introduced that utilizes curriculum learning strategy to effectively adapt using noisy target pseudolabels generated by a black-box source model by separating them into clean and noisy sets, and outperforms existing state-of-the-art black-box DA methods. This chapter is based on the paper titled "Curriculum Guided Domain Adaptation in the Dark" [82], which has been published in the IEEE Transactions on Artificial Intelligence journal.

1.2.5 Chapter 6: Source-free open-set domain adaptation

This chapter introduces a method named Unknown Sample Discovery (USD) that models the Jensen-Shannon distance between target pseudolabels and network outputs to distinguish between samples in the target domain belonging to classes that are also present in the source domain, and those samples that belong to the target-private classes. USD outperforms current source-free open-set DA methods. This chapter is based on a paper titled "Unknown Sample Discovery for Source Free Open Set Domain Adaptation" [83], which was accepted to the 1st Workshop on Test-Time Adaptation: Model, Adapt Thyself! (MAT) at The IEEE / CVF Computer Vision and Pattern Recognition Conference (CVPR) 2024.

1.2.6 Chapter 7: Conclusion and future work

This chapter presents brief conclusions about the different projects discussed in this dissertation and outlines some future directions for advancing these research projects.

Chapter 2

Background

We begin our background discussion with a brief introduction to the history and mechanism of artificial neural networks, followed by descriptions of convolutional neural network and transformers for vision.

2.1 Perceptron

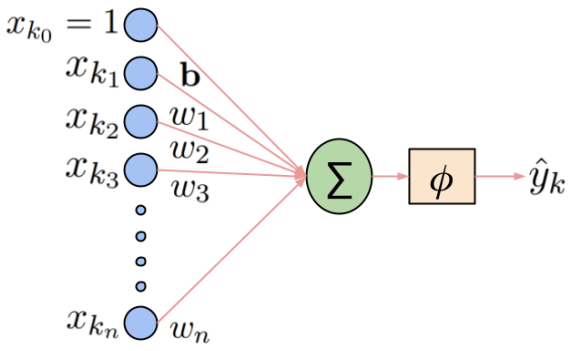


Figure 2.1: Rosenblatt's perceptron proposed in 1958 [174].

Artificial neurons were conceived in the early 19^{th} century when neural events were modeled mathematically and the relations among them were explored [147]. Later in 1958, Rosenblatt [174] laid the foundational groundwork for modern neural networks with a single layer of interconnected artificial neurons, called a

perceptron, to solve a simple binary classification problem. Let us consider a training set $\mathcal{S}$ composed of training samples $\mathcal{S}_k = (x_k, y_k)$, such that $x_k = (x_{k_1}, x_{k_2}, x_{k_3}, ..., x_{k_n})$ is a pattern vector of n dimensions, and y_k is the class label of sample S_k where $y_k \in C$ with $C = \{0, 1\}$ as the label set. The training set $\mathcal{S}$ can also be defined as $\mathcal{S} = \mathcal{S}^1 \bigcup mathcal S^0$, where $\mathcal{S}^1 = \{\mathbf{x}_k; \mathcal{S}_k = (x_k, y_k) \in \mathcal{S}, y_k = 1\}$ is the subset of positive training instances, and $\mathcal{S}^0 = \{\mathbf{x}_k; \mathcal{S}_k = (x_k, y_k) \in \mathcal{S}, y_k = 0\}$ is the subset of negative training instances for a binary classification problem. The objective of the perceptron training algorithm is to find a weight vector $\mathbf{w} = (w_1, w_2, w_3, ..., w_n)$ and the bias $\mathbf{b}$ such that,

$$\begin{aligned} \forall \mathbf{x}_k \in \mathcal{S}^1 : \mathbf{w}^T.\mathbf{x}_k + \mathbf{b} > 0 \\ \forall \mathbf{x}_k \in \mathcal{S}^0 : \mathbf{w}^T.\mathbf{x}_k + \mathbf{b} < 0 \end{aligned} \tag{2.1}$$

If the bias $\mathbf{b}$ is negative, the weighted sum of inputs need to be greater than $|\mathbf{b}|$ in order to get a positive prediction for the sample. The bias $\mathbf{b}$ thus alters the position of the linear decision boundary, while the neuron weights controls the orientation. A heavy-side step activation function is also applied to the network output to model whether or not the neuron is activated. The neuron output thus becomes,

$$\hat{y}_k = \phi(\sum_{i=1}^{n} w_i x_{k_i} + \mathbf{b}) = \begin{cases} 1 & \sum_{i=1}^{n} w_i x_{k_i} + \mathbf{b} \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.2}$$

We note that, the bias term $\mathbf{b}$ can be modeled as a trainable parameter w_{k_0} of the weight vector when the input vector also has a fixed parameter $x_{k_0} = 1$ appended at the beginning. With the estimated output $\hat{y}_k$ for sample $\mathcal{S}_k = (x_k, y_k)$, the network parameters (weights) can now be updated as,

$$\mathbf{w}_{new} = \mathbf{w}_{old} + \eta(y_k - \hat{y}_k)\mathbf{x}_k \tag{2.3}$$

where $\eta > 0$ can be termed as the learning rate. This process is repeated until convergence.

The Rosenblatt perceptron model has several limitations. First, the training will never terminate if the input set is not linearly separable. Second, the heavy-side step activation function splits the input space into two halves, with an infinite gradient at the threshold. This prevents application of optimization techniques, such as gradient descent. Minsky and Papert [151] proposed to replace the step activation function with a sigmoid activation function defined as,

$$\phi(x) = \frac{1}{1 + e^{-x}} \tag{2.4}$$

The advantages of this sigmoid activation function over the step function are manifold. Unlike step function, the output of the sigmoid function is continuous, smooth, and most importantly, differentiable at all positions. A perceptron with sigmoid activation function can therefore be optimized with gradient descent (GD). Since then, many other activation functions have been proposed with similar properties, such as, Rectified Linear Unit (ReLU) [160], Gaussian Error Linear Unit (GELU) [64], Softplus [51], Exponentially Linear Unit (ELU) [29], Scaled Exponential Linear Unit (SELU) [102], Leaky Rectified Linear Unit (Leaky ReLU) [145], Parametric Rectified Linear Unit (PReLU) [62], among others.

The sigmoid function limits the output to be continuous in the range of 0 to 1 and can be considered as output probability of the input sample for belonging to the positive class. This enables utilization of model optimization frameworks such as Maximum Likelihood Estimation (MLE) to heuristically find the probability distribution and model parameters that best explain the observed data. The maximum likelihood $\mathcal{H}$ for our binary classification problem can be written as,

$$\begin{aligned} P(y_k = 1|\mathbf{x}_k) &= \frac{1}{1 + e^{-\mathbf{w}^T.\mathbf{x}_k}} \\ P(y_k = 0|\mathbf{x}_k) &= 1 - P(y_k = 1|\mathbf{x}_k) \\ \mathcal{H} &= \prod_{k=1}^{K} P(y_k = 1|\mathbf{x}_k)^{y_k} P(y_k = 0|\mathbf{x}_k)^{1-y_k} \end{aligned} \tag{2.5}$$

The logistic loss or the binary cross-entropy loss can therefore be written as,

$$\mathcal{L}_{logistic} = -\log \mathcal{H} = -\sum_{k=1}^{K} y_k \log P(y_k = 1|x_k) - \sum_{k=1}^{K} (1 - y_k) \log(1 - P(y_k = 1|x_k)) \tag{2.6}$$

Given the loss $\mathcal{L}$, gradient descent is popularly applied to update the weight vector $\mathbf{w}$ as follows.

$$\mathbf{w} \longleftarrow \mathbf{w} - \eta \frac{\partial}{\partial \mathbf{w}} \mathcal{L} \tag{2.7}$$

Updating weights with gradient descent (GD) after calculating loss on the entire dataset of K samples results in slow convergence, and requires a large memory to store the gradients for each sample. Stochastic gradient descent (SGD) updates the model weights each time a training sample is fed, resulting in faster convergence. However, SGD may be unable to minimize the loss function as well as GD. Stochastic gradient descent with minibatch updates the model parameters with the loss calculated on a randomly selected subset of the training data, and thus achieves fast and optimal convergence.

The learning capacity of a single-layer perceptron is quite limited. It cannot classify patterns that are not separable with a hyperplane, for example a single-layer perceptron can model the linear OR logic, but fails to model the non-linear XOR logic. As a solution, a multi-layered perceptron (MLP) can model the more complex patterns. We briefly discuss MLP in the following section.

2.2 Multi-layer Perceptron

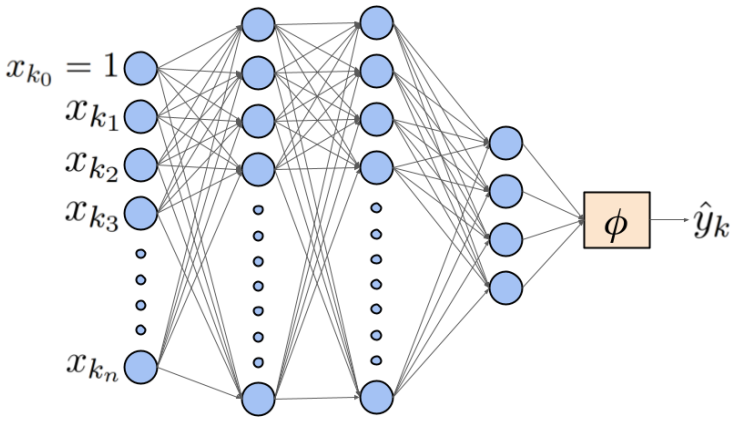


Figure 2.2: Multi layered perceptron (MLP) network.

Multi layered perceptron (MLP) consists of one or multiple hidden layers of fully connected artificial neurons between the input layer and the output layer. Given that the number of hidden layers or the number of nodes in a hidden layer are sufficient, and the activation functions are non-linear, MLPs can theoretically model any continuous function [32]. Let us add a little more complexity to the binary classification problem from the previous section, and assume that the training dataset has four classes, i.e $y_k = \{0, 1, 2, 3\}$ with two additional subsets $\mathcal{S}^2 = \{\mathbf{x}_k; \mathcal{S}_k = (x_k, y_k) \in \mathcal{S}, y_k = 2\}$ and $\mathcal{S}^3 = \{\mathbf{x}_k; \mathcal{S}_k = (x_k, y_k) \in \mathcal{S}, y_k = 3\}$ that constitute the training set $\mathcal{S} = \mathcal{S}^0 \bigcup \mathcal{S}^1 \bigcup \mathcal{S}^2 \bigcup \mathcal{S}^3$. An MLP that can model such a dataset is presented in Figure 2.2. To get the probability distribution across the classes $C = c_1, c_2, c_3, c_4$, the sigmoid activation function also has to be replaced with a different activation function to accommodate more than two output classes, preferably a softmax activation function. If $\mathbf{w_1}$ and $\mathbf{w_2}$ are the weights of the two hidden layers, the model output probability can be written as,

$$P(y_k = c|\mathbf{x}_k) = \frac{\exp\left(\mathbf{w}_c^T \mathbf{w_2}^T (\mathbf{w_1}^T (x_k))\right)}{\sum_{j=1}^{C} \exp\left(\mathbf{w}_j^T \mathbf{w_2}^T (\mathbf{w_1}^T (x_k))\right)} \tag{2.8}$$

If t_k is the one-hot encoded target of C dimensions, the MLE and categorical cross-entropy loss then can be expressed as,

$$\begin{aligned} \mathcal{H} &= \prod_{k=1}^{K} \prod_{c=0}^{C=3} P(y_k = c|\mathbf{x}_k)^{t_{kc}} \\ \mathcal{L}_{softmax} &= -\sum_{k=1}^{K} \sum_{c=0}^{C=3} t_{k_c} \log P(y_k = c|\mathbf{x}_k) \end{aligned} \tag{2.9}$$

MLPs are however ill-suited for computer vision applications. An input RGB image with three channels of size $224 \times 224 \times 3$ has $n = 150528$ dimensions. If the number of nodes in the first hidden layer is only five, the total number of parameters for the first layer alone becomes $5 \times (n + 1) = 752645$. For a deep or wide network, the number of parameters can quickly blow up, making MLPs impractical even for medium resolution images. Moreover, spatial relationship between image pixels are overlooked in MLPs, leading to very low inductive bias. As opposed to MLPs, Convolutional neural network (CNN) preserves the local spatial relationship among image pixels and reduces the number of model parameters drastically using a moving window, and is therefore better suited for dealing with images. CNNs are briefly described in the next section.

2.3 Convolutional Neural Network (CNN)

Inspired by visual nervous system in vertebrates, Fukushima *et al.* [44] proposed in 1982 an artificial neural network that used convolutions on images to extract features, and then pooled these features for global representations. The model consisted of alternate layers of (i) simpler S-cells that extract local features via convolution, and (ii) complex C-cells that pool the features and hierarchically produce global features. This model was used for Japanese character recognition. Convolutional neural networks (CNNs) in their present form was first proposed by Lecun *et al.* [112] in 1989, where the model learns a bank of convolutional filters or kernels that pan around an entire image with shared weights, and introduced gradient descent based backpropagation for deep convolutional model updates. The hyperbolic tangent activation function used in the model, however, faced the problem of exploding or vanishing gradients in a deep network. Over the course of the next decade, the model was refined and Lecun *et al.* [113] came up with LeNet for hand-written digit recognition with more hidden convolutional layers, pooling layers, and a scaled version of the earlier hyperbolic tangent activation function. The model was also trained with stochastic minibatch gradient descent for faster optimization. The LeNet model was, however, limited in terms of capacity to learn features on diverse large-scale datasets.

Starting in 2010, the ImageNet large scale visual recognition challenge (ILSVRC) [34] provided a benchmark dataset for researchers to evaluate their machine learning models on a common large-scale computer vision task. Then in 2012, Krizhevsky, Sutskever, and Hinton [104] proposed a deep CNN based architecture named AlexNet, and significantly reduced the error rate on the ImageNet classification task from 25.8% a year earlier to 16.4%. This development revolutionized neural networks, and propelled us into the new age of artificial intelligence fueled real-world solutions. To achieve quicker convergence, AlexNet normalized the response across all the channels locally at a particular location. It also introduced the rectified linear unit (ReLU) activation function, which was relatively simpler in design and easily differentiable, thereby significantly reducing vanishing and exploding gradients during training. The activation layer for the final classifier was a softmax function with 1000 classes. AlexNet was also drastically efficient; it had about 60 million parameters (same as LeNet), and 95% of the computation was done in the convolution layers which accounted for only 5% of the parameters. AlexNet proved the viability of deep neural networks for addressing real-world problems.

Machine learning based classification models usually consist of two parts: (i) a backbone or feature extractor that embeds the input image into the feature space, and (ii) a classifier that takes these features as input and produces the probability distribution function. Before deep neural networks were practically usable, the features were calculated via hand-crafted methods, such as Harris corner detector [57], SIFT [141], ORB [178] etc. Deep learning with gradient descent enabled the models to learn the proper feature representations without human intervention. Following AlexNet, several newer and deeper convolutional models were proposed, such as VGG [188] and InceptionNet [193]. VGG systematically reduced of 2D dimensions of input image through each layer, while increasing the number of channels in each layer in an organized way, enabling the deeper model with more parameters to learn representations for diverse datasets. Naively increasing the depth of a model however exacerbates the problem of exploding and vanishing gradients, as well as overfitting on the training data due to overparameterization [232]. Residual networks or ResNets [63] were proposed to curb vanishing gradients by carrying over activations from a shallow layer to a deep layer using skip connections. ResNets connect the activations of every other layer with skip connections or "shortcuts", and directly connects the adjacent layers. Two layers with a skip connection between the input and the output together form a residual block. A ResNet model is made up of stacked residual blocks. Since ResNets systematically decrease the input dimensions using pooling layers, it becomes an issue for dense predictions, such as segmentation. HRNet [206] preserves the image at multiple high resolutions (hence the name HRNet: high-resolution net), and fuses the multi-resolution features at the deep layers, enabling improved dense predictions. ResNets and HRNets are however very parameter intensive, with millions of parameters and several GigaFLOPs of operations. To be able to deploy deep neural models in edge devices, such as mobile phones, an efficient CNN called MobileNet [72] was proposed which contained depth wise separable convolutions, thereby considerably reducing the number of parameters and keeping a

sweet balance between output accuracy and network latency.

2.4 Transformers in vision

Transformers can be considered as the latest family of deep neural network backbones. Natural language processing had mostly been done with recurrent neural networks (RNN) that could classify a word in a sentence by drawing context from earlier words in the sentence. However, due to sequential nature of word processing in RNNs, the context at one state or word was dominated by the immediately earlier state or word. In 2017, Vashwani *et al.* [203] presented the modern transformer model for processing natural languages that could capture context from across the entire text, due to its self-attention mechanism. But due to the large number of pixel dimensions in an image, applying transformer models in visual tasks was not feasible at that time. Dosovitskiy *et al.* [37] in 2020, proposed to break an image into patches of 16×16 pixels, embed the patches into a embedding space and feed to a transformer model. They termed their model Vision Transformer (ViT), and added learnable positional embeddings to the image patch embeddings, in order to preserve spatial relationship among the patches. The embeddings pass through a number of stacked transformer encoder blocks, before the output is taken from the zero-th positional MLP head. Each transformer block consists of a normalization layer, followed by a multi-headed self-attention module, then another normalization layer and finally an MLP module. Skip connections connect the inputs at each normalization layer. The self attention module takes the patch embeddings and feeds them through query, key, and value parameters. Let us express the outputs of the three trainable parameters over all the image patches be $Q \in \mathbb{R}^{n \times d_k}, K \in \mathbb{R}^{n \times d_k}$, and $V \in \mathbb{R}^{n \times d_v}$, respectively, where d_k and d_v are the dimensions of the query/key and value parameters, respectively. Self attention is calculated as a dot product as follows,

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{2.10}$$

The dot product between Q and K_T produces a matrix of size $n \times n$, which consists of attention scores for each element i with all other elements in the input. It is then passed through a softmax layer before being scaled with $\sqrt{d_k}$, and then finally multiplied with the value vector V to obtain weighted means of all attention probabilities. The self-attention module enables a transformer model to capture context between two distant patches in an image. ViT is thus able to achieve global contextual understandings of an image, even at the shallower blocks, whereas CNN-based architectures are hierarchical and achieve global feature representations in the deeper layers. Having shown impressive results, ViT paved the way for numerous new

transformer based vision models, such as Swin [132, 134], DeepViT [237], Multiscale ViT (MViT) [41], among others. Swin breaks the image into a number of windows, and each window is then divided into a number of smaller 4×4 pixel patches. Swin applies self-attention more locally, across only the patches within each window. Swin overcomes local feature representations by shifting the windows, and achieves hierarchical feature scale by patch merging. MViT adopts multiscale feature hierarchies in CNN models and develops a multiscale feature pyramid for transformers with small number of channels at the higher resolutions, and larger number of channels at the lower resolution. MViT is thus able to capture context at multiple resolutions.

Chapter 3

EO-guided SAR image classification

This chapter is based on three of our inter-related papers titled "Cross-modal knowledge distillation in deep networks for SAR image classification" [85], "SAR Image Classification with Knowledge Distillation and Class Balancing for Long-Tailed Distributions" [86], and "Balanced sampling meets imbalanced datasets for SAR image classification" [81] that appeared at SPIE DCS 2022, IVMSP 2022, and SPIE DCS 2023 conferences, respectively. Deep learning based classification of SAR images is a challenging task due to the nature of SAR imagery and apparent noise. On the other hand, Electro-Optical (EO) image classification has been extensively studied with great success using deep learning methodologies. In this chapter, we propose a novel framework for knowledge distillation from EO to SAR, that is response-based and takes into consideration the differences in network size and feature representations in the two modalities. Our training approach includes of two/three stages consisting of 1) EO network training, 2) SAR network training with transfer learning and knowledge distillation from the EO network, and an optional 3) class-balanced training of the SAR network to account for long-tailed distributions in the data. Our approach is guided by the differences in physical characteristics between the EO and SAR modalities, as our knowledge distillation is performed at the soft output level and allows different types of features in the EO and SAR networks. Our model is agnostic in the selection of network backbone and does not place any constraints on the network architecture, thus making knowledge transfer applicable even from a smaller network to a larger network. We test our approach on a recent EO-SAR coupled dataset with promising results on SAR image classification. Our method achieves performance gains in each stage and for each component of the model, as evidenced in our ablation studies.

3.1 Introduction

Recent advances in earth observation data collection technology [53] have contributed to a dramatic increase in the amount and variety of remotely sensed images. This has facilitated research into data-driven deep learning methods that can leverage the ever increasing volume of data. Several deep learning based methods have explored ways to perform pixel, object and scene-level image classification of remotely collected satellite imagery [73,121,142]. Most of these methods deal with Electro-Optical (EO) imagery sensed in the visual spectrum [25,26,118]. Some models have been developed for image classification of Synthetic Aperture Radar (SAR) data, either at the pixel level [50, 157, 235] for coarse semantic segmentation of the overall scene, or at the object level [7,236], under certain limitations and prior assumptions. However, none of these methods take advantage of joint learning with coupled EO-SAR sample pairs for knowledge transfer and/or distillation from the EO domain to the SAR domain.

Here, we propose a method to leverage learning of electro-optical image data to guide SAR object classification. This approach is promising because the two modalities are drastically different in their physics of light capture. Due to the difference in image appearance between EO and SAR, different types of features need to be learned in each domain and knowledge transfer is performed during classification. Since EO samples are more easily accessible and less noisy, we propose cross-modal knowledge distillation from an EO trained network to a network trained for SAR image classification. Such an EO-SAR knowledge transfer has not been studied before.

Although several methods [1, 78, 149, 150, 155] have worked with EO-SAR data fusion, they require both the EO and SAR images for inference. In contrast, we utilize corresponding EO-SAR image pairs to guide the training of a SAR classification network. During inference, our model only requires the SAR images, thereby allowing its deployment with SAR data only, under all weather conditions and taking full advantage of radar images. To address the issue of class imbalance in the SAR-EO coupled dataset [136], we have explored two sampling strategies during training: instance sampling and class balanced sampling. Examples of EO and SAR images are shown in Figure 3.1.

The major contributions of this work are:

1. We propose a novel physics-guided deep learning framework for knowledge distillation across modalities.

2. We present an EO-guided SAR image classification scheme, where a network model trained to classify EO data guides the training of a SAR classification network.

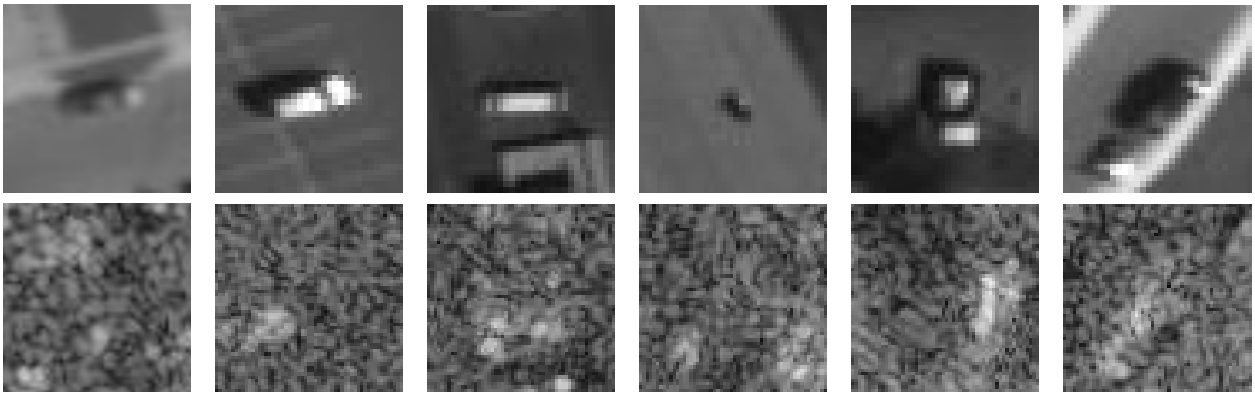


Figure 3.1: Representative examples from the coupled EO-SAR dataset. The top row shows EO images and the bottom row shows SAR images. Each column represents one class. From left to right, the classes are: sedan, SUV, van, motorcycle, flatbed truck, and pickup truck with trailer.

3. Our approach for cross-modal knowledge distillation from the EO to the SAR domain is independent of the network backbone and can transfer knowledge from a smaller network to a larger network or vice-versa.

4. Our multi-stage training procedure addresses the dataset class imbalance with class balanced dataloading strategy, either mixed with instance data sampling in the second training stage, or in a separate third and final stage of training, to reduce classifier bias towards the more populated classes.

5. Our SAR classification model does not require any EO data during inference, unlike existing SAR-EO data fusion methods that require samples from both domains.

6. We validate our model on a new SAR-EO dataset, and conduct ablation studies for different parts of our model and stages of training, showing the efficacy of each component.

The rest of the chapter is organized as follows: In Section 3.2, we discuss related works; in section 6.3, we present the details of our method; in section 3.4, we discuss the SAR-EO coupled dataset we use to evaluate our model; in section 3.5, we show quantitative results and ablation study for the different components of our model; and in section 6.6, we present final remarks.

3.2 Related Work

3.2.1 SAR image classification

Manual SAR image annotation for training classification models remains a challenging task that requires extensive experience and is labor intensive and expensive. Hand-crafted feature extractors that work on scattering properties and texture information were proposed by [61, 153]. Recently, with the advent of unmanned aerial vehicles (UAVs) and cost-effective satellite technologies, significant gains have been made in aerial image collection. The large inflow of such data has lead to feature generation and selection based on data driven learning using deep networks, and has facilitated the development of automated data analysis algorithms dedicated to aerial images. Chen *et al.* [24] proposed a method based on convolutional neural network (CNN) deep feature extraction on hyperspectral images (HSI) in one of the earliest deep learning models on remotely sensed data. Li *et al.* [122] proposed a light, easy-to-train 3D-CNN framework to extract deep spectral–spatial-combined features from HSI.

Object detection using SAR images is an area of active research interest. Most of these methods mostly utilized the Moving and Stationary Target Acquisition Recognition (MSTAR) dataset [2] and the ship detection datasets [76, 172, 186] to validate their models. Chen *et al.* [22] was probably the first to use deep learning on SAR images by developing a sparse autoencoder (SAE) with a mono-layer CNN on random patches of a SAR image and training a softmax classifier on top of it to classify military vehicles. Later, with a simple five-layer CNN, Chen *et al.* [23] achieved 99% accuracy on MSTAR. This led to subsequent investigations into the efficacy of CNNs for SAR image classification. Building on promising results based on deep learning methods, Morgan [154] used a shallow three-layer CNN on MSTAR, while Wilmanski *et al.* [210] probed the effects of various methods of weight initialization and optimizer selection. Ding *et al.* [35] examined the significance of SAR mode specific data augmentation techniques for a CNN-based SAR object detection model. Du *et al.* [38] portrayed the importance of data augmentation of SAR training samples and proposed a CNN invariant to both displacement and rotation. To identify and localize more than one object in a SAR image, Furukawa [45] developed an encoder-decoder segmentation network. Bai *et al.* [7] proposed SAR object classification via a bidirectional convolution-recurrent network under the assumption that target images of an object are generated continuously and sequentially at a fixed azimuth angle intervals.

More recently, researchers have looked at methods to complement legacy DL based SAR classification algorithms. Dechesne *et al.* [33] used a multitask network for detection, classification, and prediction of the length of ships, simultaneously. Mullissa *et al.* [157] used a CNN and Kazemi *et al.* [95] used an RNN architecture on complex valued SAR data and on directly received SAR signals, respectively. Both Rostami

et al. [176] and Huang *et al.* [77] worked on transfer learning from optical modality to the SAR modality for image classification.

Several research works have also considered SAR-EO data fusion. In order to do this, it is imperative that image pairs in both modalities are co-registered. To this end, these methods have relied on deep learning to extract features from the two modalities, match correspondences, concatenate the features and finally train a classifier [78, 155]. Merkel *et al.* [150] used a Siamese network to extract features and a dot product layer as a similarity measure. Abdulkhanov *et al.* [1] developed a feature point descriptor using neural nets and used random sample consensus [43] to match the detected descriptors. Merkle *et al.* [149] used a conditional GAN to generate synthetic SAR images from optical images and then matched them to the real SAR images. The success of this intermediate step in improving precision initiated more research into such approaches [79].

Other than object classification, such SAR-EO data fusion has also been applied to semantic segmentation [6, 224]. It is to be noted that all these methods used data from both SAR and EO modalities for training and prediction, and did not consider predictions using SAR images alone without additional support from EO data. This reliance on EO data poses the risk of biasing the system to perform classification based on the EO features and to a large extent ignore the SAR data. In this work, we take the approach of performing classification using SAR data alone, and using EO data to boost learning through knowledge distillation.

3.2.2 Knowledge distillation

Knowledge distillation (KD) was originally proposed by Hinton *et al.* [66], as a process for model compression where knowledge is distilled or transferred from a larger model to a smaller model. Large and deep networks have achieved impressive results on several computer vision tasks [37, 63, 133]. However, large models are not always feasible for deployment, particularly in mobile devices and embedded systems, due to the model size and computational requirements. Through knowledge distillation, a smaller student model is guided to mimic a larger teacher model.

In vanilla KD, the logits or soft predictions of the teacher network are considered as the knowledge to be distilled to the student model. The objective of KD in student training is to minimize the Kullback-Leibler divergence loss between the teacher and student network logits. This is also called response-based KD. In contrast, feature-based KD [65, 173, 228] uses the intermediate feature representations at different layers of a deep teacher network to train a student network. Passalis and Tefas [165] proposed KD by matching the probability distributions of the teacher and student feature spaces. Jin *et al.* [91] proposed to train the student network through outputs of certain hint layers of the teacher network. Challenges with feature-based KD

include selecting the hint layers in the teacher network and the corresponding guided layers in the student network, as well as finding the appropriate method of distilling knowledge due to the difference in sizes between respective layers in the teacher and the student.

In this work, we present a method to leverage EO domain knowledge to train a SAR object prediction model using response-based knowledge distillation. Our model differs from existing models in that we do not need access to EO data for prediction. This method does away with any patch matching module or any algorithmic overhead needed for data fusion between SAR and optical images. In the deployment stage, we do not require an EO sensor and therefore our model can leverage the advantages of SAR imagery in all weather conditions, including clouds or similar visual obstructions. Our multi-stage training also ensures that our model is not skewed towards any particular class and we take steps to ensure that our model works well for both common and rare class samples it may encounter.

3.3 Methodology

Let us denote labeled samples in the EO domain as $\{x_t^i, y_t^i\}_{i=1}^n$ where n is the total number of samples $x_t^i \in \mathcal{X}_t$ with corresponding labels $y_t^i \in \mathcal{Y}_t$. Similarly, the SAR domain is denoted as $\{x_s^i, y_s^i\}_{i=1}^n$ where the samples and corresponding labels are given by $x_s^i \in \mathcal{X}_s$ and $y_s^i \in \mathcal{Y}_s$, respectively. Our method of leveraging EO data to better train the SAR model for classification involves three stages of training, as illustrated in Figure 3.2. The operations in each stage are described below.

3.3.1 Stage 1: EO training

In the first phase, we train the EO teacher model $f_t : \mathcal{X}_t \rightarrow \mathcal{Y}_t$ using the principles of supervised learning and by minimizing the cross entropy loss as follows.

$$\mathcal{L}_{t,ce}(f_t; \mathcal{X}_t, \mathcal{Y}_t) = -\mathbb{E}_{(x_t,y_t)\in\mathcal{X}_t\times\mathcal{Y}_t} \sum_{m=1}^{C} q_m log(\sigma_m(f_t(x_t), T)) \tag{3.1}$$

where q is the one-hot-encoding of the ground truth labels y_t, such that q_m is 1 for the correct class and 0 for the incorrect class. For the m-th element of the output logits vector z of C-dimensions, the softmax probability function is represented by $\sigma_m(z^m, T)$ as follows. 3.2.

$$\sigma_m(z^m, T) = \frac{exp(z^m/T)}{\sum_i exp(z^i/T)} \tag{3.2}$$

where T is a temperature parameter. In order to facilitate smoother decision boundaries among the categories, we employ label smoothing [156] and modify the objective function as follows.

$$\mathcal{L}_{t,ce}(f_t; \mathcal{X}_t, \mathcal{Y}_t) = -\mathbb{E}_{(x_t,y_t)\in\mathcal{X}_t\times\mathcal{Y}_t} \sum_{m=1}^{C} q_m^{ls} log(\sigma_m(f_t(x_t), T)) \tag{3.3}$$

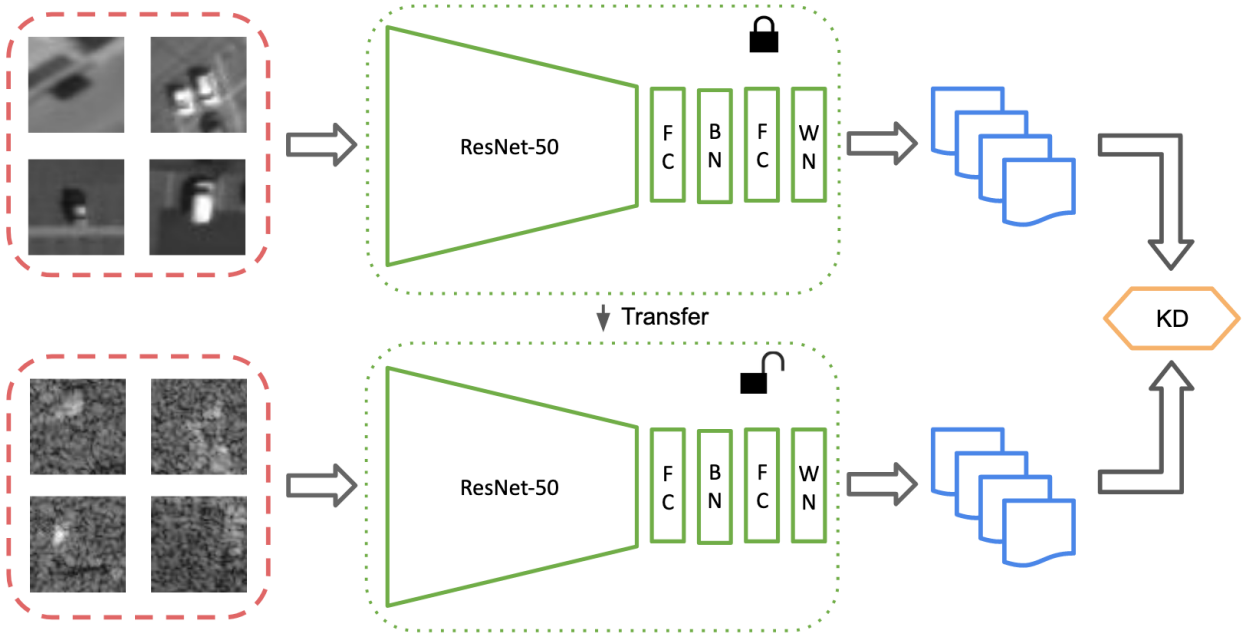


Figure 3.2: Proposed framework for cross-modal training with knowledge distillation. In stage 1: EO training, only the top branch is trained and the bottom branch along with KD block is removed. In Stage 2: SAR training with knowledge distillation, the top branch is locked and the bottom branch is trained for SAR image classification with knowledge distillation from the EO network. In an optional Stage 3: SAR training with class balancing, only the bottom branch is trained and the top branch along with KD block is removed.

3.3.2 Stage 2: SAR training with knowledge distillation

In this stage, we introduce cross modal knowledge distillation from the EO modality (teacher) to the SAR modality (student). We train the SAR student model $f_s : \mathcal{X}_s \rightarrow \mathcal{Y}_s$ by minimizing the following cross entropy loss of the SAR samples x_s and their corresponding ground truth labels y_s.

$$\mathcal{L}_{s,ce}(f_s; \mathcal{X}_s, \mathcal{Y}_s) = -\mathbb{E}_{(x_s,y_s)\in\mathcal{X}_s\times\mathcal{Y}_s} \sum_{m=1}^{C} q_m log(\sigma_m(f_s(x_s), T)) \tag{3.4}$$

Additionally, we conduct KD from the soft targets of the teacher model to the soft targets of the source model. Our response-based KD method takes vectors of logits $z_t = f_s(x_s), z_s = f_s(x_s)$ from the teacher and student models, respectively. The logits are then converted to soft targets $\sigma(z_t, T)$ of the teacher model and soft predictions $\sigma(z_s, T)$ of the student model following equation 3.2. The soft targets of the teacher model are defined as those containing the informative knowledge that is transferred to the student [66]. In our method, we employ the Kullback-Leibler (KL) divergence loss between the $\sigma(z_t, T)$ and $\sigma(z_s, T)$ as our knowledge distillation loss, which can be written as

$$\mathcal{L}_{kd}(\sigma(z_s, T), \sigma(z_t, T)) = KL(\sigma(z_s, T), \sigma(z_t, T)) \tag{3.5}$$

where,

$$KL(a, b) = \sum_{j \in J} a(j) log \frac{a(j)}{b(j)} \tag{3.6}$$

Romero *et. al.* [173] subsequently developed methods that use a feature-level distillation loss or an amalgam of output and feature level distillation losses to transfer knowledge from teacher to student, as proposed in [17,91,99,238]. However, such a process is not applicable in our case of transferring knowledge from EO to SAR, because the two modalities are drastically different in their physics of light capture, image generation and processing. Their respective samples are captured with different sensors at different wavelengths. Since light interacts with the objects in the scene differently at different wavebands, the resulting images in the two EO and SAR modalities are significantly different. Hence, by choice, we have focused our knowledge distillation approach only on the final logit layer. We also understand that, although a SAR image may seem to be very noisy to the naked eye, it contains object specific signatures, as the object may interact differently with radar wavelengths, where the SAR operates, than the visible spectrum of EO images. These differences in image appearance due to the sensing modality may prove to be significant for classification. Therefore, we make the decision to avoid performing any despeckling in the SAR images, as it may remove important and representative object signatures from the image. The objective function of this stage is, therefore,

$$\mathcal{L}_{s,tot}(f_s; \mathcal{X}_s, \mathcal{Y}_s, \mathcal{X}_t) = \mathcal{L}_{s,ce}(f_s; \mathcal{X}_s, \mathcal{Y}_s) + \alpha \mathcal{L}_{kd}(\sigma(z_t, T), \sigma(z_s, T)) \tag{3.7}$$

where α and T are hyper-parameters. We name our model upto this point **Cross-KD** [85].

3.3.3 Alternate Stage 2: Class balanced SAR training

Cross-KD does not consider the dataset imbalance that is prevalent in SAR image datasets, and therefore performs underwhelming for the tail classes if the class imbalance is high. In order to account for this class

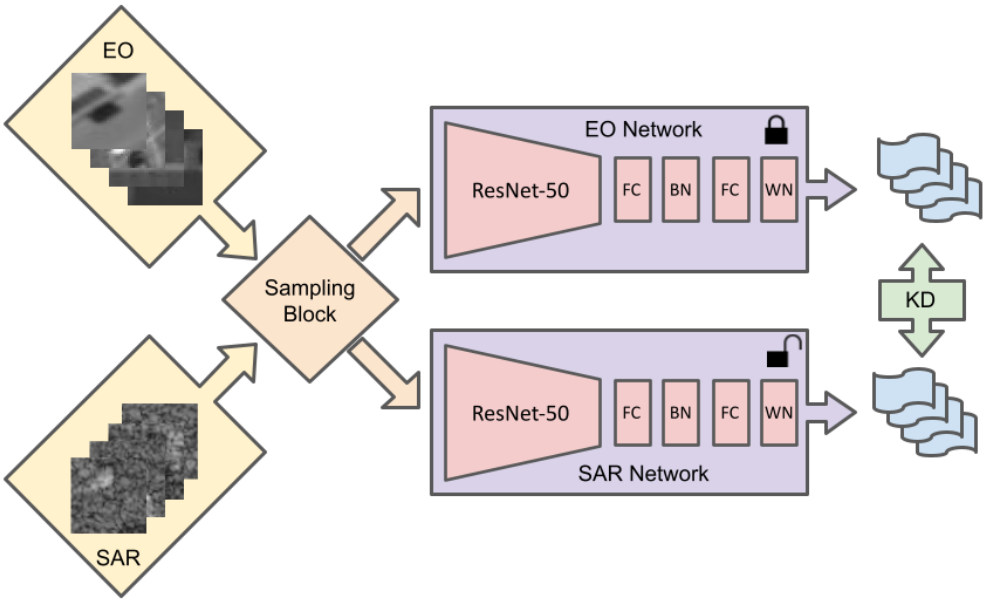


Figure 3.3: Proposed framework for Balanced Cross-KD during SAR training, where the EO network is locked and the SAR network undergoes training. The Sampling Block selects samples either based on instance sampling or class-balanced sampling strategy. The laters FC, BN and WN represent Fully Connected layer, Batch Normalization layer and Weight Normalization layer, respectively.

imbalance in the dataset and the resultant model bias towards the dominant classes, during the SAR model training, we deploy the Sampling Block which implements a carefully curated alternating sampling strategy between instance sampling and class balanced (CB) sampling. During training and optimization via minibatch stochastic gradient descent, one iteration of class balanced sampling is done for every two iterations of instance sampling. This mixture of balanced and imbalanced/instance sampling strategy attempts to de-bias the SAR model to perform well across all the classes, irrespective of the number of samples in the class, while optimally learning feature representations. In addition to cross-entropy loss from equation 3.4 and knowledge distillation loss from equation 3.5, we utilize equal diversity loss L_{eqdiv} [198] for class balanced sampling, shown in Equation (3.8), and to further help the network de-skew, we introduce distributed entropy loss L_{disent} for instance sampling, shown in equation (3.9) below.

$$\mathcal{L}_{eqdiv}(f_s; \mathcal{X}_s) = \sum_{m=1}^{C} p_m log\left(\frac{p_m}{\overline{p}_m}\right) \tag{3.8}$$

$$\mathcal{L}_{disent}(f_s; \mathcal{X}_s) = -\mathbb{E}_{x_s \in \mathcal{X}_s} \sum_{m=1}^{C} \hat{\sigma}_m(f_s(x_s), T) log(\sigma_m(f_s(x_s), T)) \tag{3.9}$$

where, $\hat{\sigma}$ is the distributed probability function, defined as,

$$\hat{\sigma}_m = \begin{cases} \sigma_m & \text{if } \sigma_m = max(\sigma) \\ \frac{1-max(\sigma)}{C-1} & \text{if } \sigma_m \neq max(\sigma) \end{cases} \tag{3.10}$$

and, p is a C dimensional vector of uniform mean response, such that $p_m = \frac{1}{C}$. p is therefore the ideal mean network output under a class balanced sampling strategy. $\overline{p}_m = \mathbb{E}_{x_s \in \mathcal{X}_s}[\sigma(f_s(x_s), T)]$ is the real mean of the output probabilities during the class-balanced sampling. Together with cross-entropy loss and KD loss, the final objective function becomes the following.

$$\mathcal{L}_{s,tot}(f_s; \mathcal{X}_s, \mathcal{Y}_s) = \begin{cases} \mathcal{L}_{s,ce}(f_s; \mathcal{X}_s, \mathcal{Y}_s) + \alpha \mathcal{L}_{kd}(\sigma(z_s, T), \sigma(z_t, T)) + \beta_1 \mathcal{L}_{disent}(f_s; \mathcal{X}_s) \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{for instance sampling} \\ \mathcal{L}_{s,ce}(f_s; \mathcal{X}_s, \mathcal{Y}_s) + \alpha \mathcal{L}_{kd}(\sigma(z_s, T), \sigma(z_t, T)) + \beta_2 \mathcal{L}_{eqdiv}(f_s; \mathcal{X}_s) \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{for CB sampling} \end{cases} \tag{3.11}$$

where, α, β_1 and β_2 are hyper-parameters. We term this model with mixed sampling strategy as **Balanced-CrossKD** [81].

3.3.4 Optional Stage 3: SAR training with class balancing

We further explore separating the SAR feature learning stage and the network debaising stage. Instance sampling helps the network learn better feature representations, while class balanced sampling trains the classifier to perform better on samples from the imbalanced classes [93]. Building upon the training of Stage 2 in Cross-KD, we take the SAR model trained with instance sampling, and train it further with class-balanced minibatches in a separate Stage 3, as opposed to the mixture of instance and class balanced sampling in the single stage 2 for Balanced-CrossKD.

We add to our objective function the equal diversity loss $\mathcal{L}_{eqdiv}$ [198] as in Balanced Cross-KD, and the entropy loss $\mathcal{L}_{ent}$ that helps generate precise predictions. Together, these two form the information maximization (IM) loss, used in [52, 74, 125, 187, 198]. Mathematically, the entropy loss $\mathcal{L}_{ent}$ can be written as,

$$\mathcal{L}_{ent}(f_s; \mathcal{X}_s) = -\mathbb{E}_{x_s \in \mathcal{X}_s} \sum_{m=1}^{C} \sigma_m((f_s(x_s), T) log(\sigma_m((f_s(x_s)), T)) \tag{3.12}$$

The total objective function in this stage is therefore,

$$\mathcal{L}_{s,tot}(f_s; \mathcal{X}_s, \mathcal{Y}_s) = \mathcal{L}_{s,ce}(f_s; \mathcal{X}_s, \mathcal{Y}_s) + \gamma_1 \mathcal{L}_{ent}(f_s; \mathcal{X}_s) + \gamma_2 \mathcal{L}_{eqdiv}(f_s; \mathcal{X}_s) \tag{3.13}$$

where γ_1 and γ_2 are hyper-parameters. This model with a distinct stage 3 of training is termed as **Cross-KD+** [86].

3.4 Datasets and Experiments

3.4.1 Datasets

The dataset used to evaluate our method was released for the "NTIRE 2021 Multi-modal Aerial View Object Classification Challenge - Track 1 (SAR)" competition [136], held as part of a 2021 Conference on Computer Vision and Pattern Recognition Workshop. The dataset has coupled images of the same targets captured by SAR and EO cameras. It has image chips of 10 classes of vehicles: sedan, SUV, pickup truck, van, box truck, motorcycle, flatbed truck, bus, pickup truck with trailer and flatbed truck with trailer. The EO images are of size 31 $\times$ 31 pixels, while the size of the SAR images ranges from 50 $\times$ 50 to 60 $\times$ 60 pixels. The objects of interest are all centered in the images. A few examples of the chips are shown in Figure 3.1. The distribution of the dataset is given in Table 3.1 and shows that the dataset is imbalanced and highly skewed towards the class "sedan". We randomly split the dataset in a ratio of 9:1 for training and testing, respectively. The resultant splits are representative of the class distribution of the whole dataset.

Table 3.1: Sample distribution across the ten classes in the dataset.

Class	Samples in each mode	% of total samples
Sedan	234,209	79.72
SUV	28,089	9.56
Pickup truck	15,301	5.21
Van	10,655	3.63
Box truck	1,741	0.59
Motorcycle	852	0.29
Flatbed truck	828	0.28
Bus	624	0.21
Pickup truck with trailer	840	0.29
Flatbed truck with trailer	633	0.22
Total	293,772	100.00

3.4.2 Implementation details

Table 3.2: Class-wise accuracy for the EO model after training on the EO images only.

sedan	SUV	pickup truck	van	box truck	motor cycle	flatbed truck	bus	pickup truck with trailer	flatbed truck with trailer	Mean per class
99.99	99.41	99.14	98.86	100.0	95.12	100.0	95.31	99.01	100.0	98.68

We use ResNet-50 [63] as the feature extractor backbone for both the teacher and the student models. The feature representations are then passed through a fully connected (FC) layer followed by a batch normalization layer [80] and then another FC layer followed by a weight normalization layer [184]. The soft outputs or class probabilities of the student and teacher models are used for calculating KL-divergence and subsequently for conducting knowledge distillation.

We update our model weights using an SGD optimizer with a momentum of 0.9. The learning rate of the ResNet backbone is set to 1/10th the learning rate of the layers after the backbone. The learning rate for the backbone is set to $\eta_0 = 1e^{-3}$ while that of the later layers is set to $\eta_0 = 1e^{-2}$. A learning rate scheduler $\eta = \eta_0 \cdot (1 + 10 \cdot p)^{-0.75}$ is also used, where p is the ratio of current iteration to maximum iterations and increases from 0 to 1 as the training continues. In the second stage for all three methods, α is set to 0.9. For Balanced Cross-KD, in the second stage, $\beta_1 = \beta_2 = 1$. In the third stage for Cross-KD+, both γ_1 and γ_2 are set to 1. Temperature $T = 4$ for calculating KD loss, in other cases, $T = 1$. The training is done using an NVIDIA GeForce RTX-2080Ti GPU.

3.5 Results

In this section, we present results for our method, as well as the impact of different model components and training strategies. Table 3.2 presents the class-wise accuracy for the EO model after training on the EO images only, after the stage 1 training. We can see that the model performs very well for EO image classification across all classes. Table 3.3 shows results for the three variants of our method. The results for Cross-KD and Balanced Cross-KD are after their two-stage processes, and that for Cross-KD+ are after its three-stage training regimen. It is evident that class balanced sampling greatly improves accuracies across the tail classes, and thus the mean per class accuracies for Balanced Cross-KD and Cross-KD+ are higher than that in Cross-KD. Cross-KD+ beats Cross-KD by $\sim 1 - 6\%$, and Balanced Cross-KD beats Cross-KD by $\sim 1 - 5\%$ across all classes except for the head-most class "sedan". However, Cross-KD with

only instance sampling beats the other two versions in the ”sedan” class accuracy, particularly due to the high degree of dataset imbalance. Moreover, Cross-KD+ beats Balanced Cross-KD, and thus shows the effectiveness of separating the feature representation learning stage with instance sampling, and classifier debiasing with class balanced sampling. Cross-KD+ however underperforms the other two variants for the top ”sedan” class.

Table 3.3: Class-wise accuracy for different forms of our model. Cross-KD and Balanced Cross-KD are 2 stage processes, while CrossKD+ is a 3 stage process.

Class	Model		
	Cross-KD	Balanced Cross-KD	Cross-KD+
Sedan	**99.26**	97.75	97.27
SUV	97.01	97.22	**98.02**
Pickup truck	94.25	95.24	**97.42**
Van	92.37	93.61	**94.76**
Box truck	96.99	98.19	**100.00**
Motorcycle	82.93	**90.24**	**90.24**
Flatbed truck	97.30	97.3	**100.00**
Bus	85.94	89.06	**93.75**
Pickup truck with trailer	**100.00**	**100.0**	**100.00**
Flatbed truck with trailer	93.75	**98.44**	95.31
Mean Per Class	93.98	95.71	**96.68**

Ablation study on transfer learning by initializing the SAR network with the weights from the EO-trained network and knowledge distillation from teacher EO network to student SAR network for **Cross-KD** is given in table 3.4. The results show that the highest performance is obtained when we combine transfer learning with knowledge distillation. This demonstrates the benefit of EO-guided training for SAR object classification, which results in a boost in performance due to better training of the SAR prediction network.

We further conduct an ablation study on the loss functions for training the SAR network with a mix of instance and class balanced sampling in **Balanced Cross-KD** is shown in table 3.5. ”Base” refers to the model where the 2nd stage training is done with transfer learning and knowledge distillation from the EO-trained teacher model, but without $\mathcal{L}_{eqdiv}$ and $\mathcal{L}_{disent}$. We can see that the ”base” Balanced Cross-KD model outperforms Cross-KD in terms of mean per class accuracy by $\sim 0.7\%$, and achieves better accuracy on the five tail classes. This shows the efficacy of class balanced datasampling in debiasing the classifier from being skewed by the dominant classes for the imbalanced EO-SAR coupled dataset. The gradually increasing performance gains with the addition of $\mathcal{L}_{eqdiv}$ and $\mathcal{L}_{disent}$ to the objective function of Balanced Cross-KD are evident in the results. All components of the objective function work in tandem to achieve the

Table 3.4: Ablation study for Stage 2 SAR model training in CrossKD. "TL" means transfer learning to initiate the student model with teacher model parameters. "KD" refers to knowledge distillation from EO teacher to SAR student model.

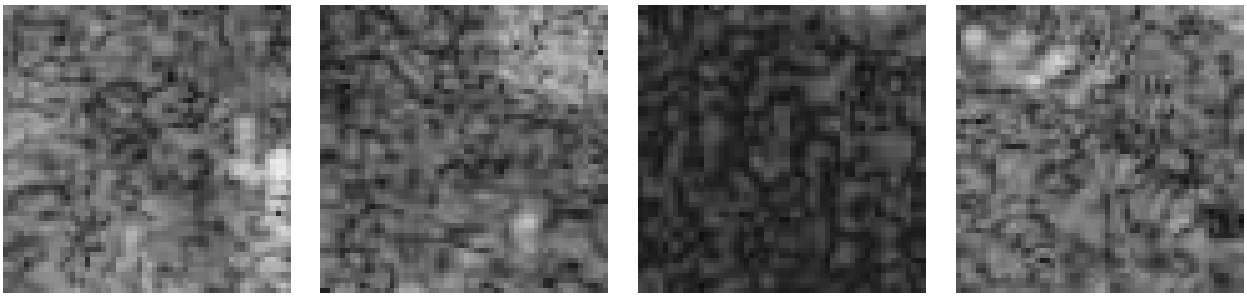
Model Configuration for Cross-KD			
Class	w/o TL, w/o KD	w TL, w/o KD	w TL, w KD
Sedan	99.20	99.35	99.26
SUV	92.42	94.26	97.01
Pickup truck	92.93	93.33	94.25
Van	88.75	89.42	92.37
Box truck	96.99	96.99	96.99
Motorcycle	84.15	81.71	82.93
Flatbed truck	94.59	94.59	97.30
Bus	85.94	84.38	85.94
Pickup truck with trailer	99.01	99.01	100.00
Flatbed truck with trailer	90.62	93.75	93.75
Mean Per Class	92.46	92.68	**93.98**

Table 3.5: Balanced Cross-KD results for various configurations. "Base" refers to the mixture of class balanced sampling and instance sampling, and trained with knowledge distillation and cross-entropy losses only.

Model Configuration for Balanced Cross-KD						
Base	✓	✓	✓	✓	✓	✓
$\mathcal{L}_{eqdiv}$		✓	✓		✓	✓
$\mathcal{L}_{disent}$			✓			✓
Class	Percent Accuracy			F1 Score		
Sedan	97.78	97.58	97.75	0.98	0.98	0.98
SUV	94.85	93.53	97.22	0.94	0.95	0.96
Pick-up Truck	93.66	92.93	95.24	0.93	0.94	0.95
Van	91.23	92.76	93.61	0.91	0.89	0.90
Box truck	96.89	98.19	98.19	0.96	0.96	0.96
Motorcycle	86.09	89.02	90.24	0.90	0.91	0.91
Flat-bed truck	97.95	98.65	97.3	0.97	0.97	0.98
Bus	88.62	90.62	89.06	0.83	0.83	0.84
Pick-up truck with trailer	100.0	100.0	100.0	0.98	1.00	1.00
Flat-bed truck with trailer	100.0	96.88	98.44	0.95	0.95	0.96
Mean Per Class	94.71	95.02	**95.71**	0.935	0.938	**0.944**

Table 3.6: Ablation study for the loss functions in the third stage for CrossKD+ SAR training with class-balanced loader.

Model Configuration for Cross-KD+		
Class	Stage 3	
	w/o $\mathcal{L}_{ent} + \mathcal{L}_{eqdiv}$	with $\mathcal{L}_{ent} + \mathcal{L}_{eqdiv}$
Sedan	96.75	97.27
SUV	97.08	98.27
Pickup truck	96.90	97.42
Van	93.42	94.76
Box truck	99.4	100.00
Motorcycle	89.02	90.24
Flatbed truck	100.00	100.00
Bus	93.75	93.75
Pickup truck with trailer	100.00	100.00
Flatbed truck with trailer	98.44	95.31
Mean Per Class	96.47	**96.68**



(a) w/o TL, w/o KD: sedan
Cross-KD: flatbed truck
Cross-KD+: flatbed truck
GT: flatbed truck

(b) w/o TL, w/o KD: sedan
Cross-KD: sedan
Cross-KD+: flatbed truck
GT: flatbed truck

(c) w/o TL, w/o KD: sedan
Cross-KD: sedan
Cross-KD+: bus
GT: bus

(d) w/o TL, w/o KD: sedan
Cross-KD: bus
Cross-KD+: bus
GT: bus

Figure 3.4: Examples of correct predictions after training to illustrate the effects transfer learning and knowledge distillation in Stage 2 for **Cross-KD** and subsequent class balanced training in Stage 3 for **Cross-KD+**. Red labels indicate incorrect prediction, green labels indicate correct prediction, and blue labels indicate ground truth.

best results, and leaving out any of them hurts optimal model performance.

We also conduct an ablation study for training on SAR images in a separate 3rd stage with class-balanced loader for **Cross-KD+**, and present the results in Table 3.6. We observe in the first column of Table 3.6 that the mean per-class accuracy due to class balanced sampling in the third stage, increases by $\sim 2.49\%$ compared to instance sampling, as seen in results from Stage 2 in Cross-KD in Table 3.4. Results on

individual classes improve for all cases, except "sedan", validating our assumption that the imbalanced nature of the dataset requires additional training to reduce bias towards any particular class. Our method of class-balanced sampling in the third stage trains the network to avoid the pitfalls of the long-tailed SAR dataset, such as bias towards the most populated class. However, the network achieves diversity in prediction at the expense of performance in the head class "sedan". In a practical deployment situation, such as when an aircraft or UAV equipped with a SAR sensor hovers above a parking lot or a highway, our model is more likely to come across sedans and less likely to encounter a more rare class, for instance a flatbed truck with trailer. Towards this end, we added the entropy loss $\mathcal{L}_{ent}$, and equal diversity loss $\mathcal{L}_{eqdiv}$, as components in the information maximization loss. The results of training in Stage 3 with the information maximization loss, in addition to class balanced sampling are shown in the last column of Table 3.6. Training with the $\mathcal{L}_{ent}$ loss increases the certainty of prediction by learning a more robust decision boundary between classes, while the $\mathcal{L}_{eqdiv}$ loss maintains a global diversity of the outputs. This push-pull mechanism, increases the performance for all classes of vehicles except for "flatbed truck with trailer", which is the least represented class in the training dataset. Categories "sedan", "suv", "pickup truck", "van", "box truck", and "motorcycle" gain classification accuracy increases between 1.33% and 0.52%. Other underrepresented classes maintain their performance between the two variations of losses during the Stage 3 training. This illustrates that our model is effective for both frequent and rare cases. A few examples of how our model learns during the three stages are shown in Figure 3.4.

3.6 Conclusion

The classification of object classes in SAR images is a challenging task, owing to the nature of the image and noise associated with it. On the other hand, EO image classification, including aerial images, is more tractable and has been extensively studied with deep learning methodologies. In this work, our aim is to take the knowledge gained from an EO image classification network, and employ it to train a SAR image classification network. We employ knowledge distillation from EO to SAR for the first time, in a manner that takes into consideration the differences between feature representations in the two modalities. Our models can work across a variety of networks and can even transfer knowledge from a smaller to a larger network, thus offering the flexibility needed for various training and deployment platforms.

Our training takes place in either two or three stages that progressively incorporates transfer learning, knowledge distillation, and class balancing strategies. In order to account for high class imbalance in SAR datasets, we perform SAR model training via a mix of balanced and instance sampling, either together in a single stage or separately in two stages, and use entropy loss and equal diversity loss to mitigate model bias towards any particular class. We explicitly define our loss functions to implement class balancing and ensure

that our models are not biased towards the most populated classes. Our ablation studies show gradual improvement in performance for each component of our models and illustrate their benefits. We hope that this work on EO-guided SAR image classification will invite more research in the area of cross-modal and physics-guided learning, and advance deep network models that are suitable for deployment on platforms with SAR sensors.

Chapter 4

Continual unsupervised domain adaptation

This chapter is based on our IEEE Transactions of Artificial Intelligence paper titled "Continual Unsupervised Domain Adaptation in Data-Constrained Environments" [198], our CVPR 2022 paper titled "Unsupervised Continual Learning for Gradually Varying Domains" [197], and another paper published in the SPIE Journal of Applied Remote Sensing titled "Continual Domain Adaptation on Aerial Images under Gradually Degrading Weather" [84]. Domain Adaptation (DA) techniques aim to overcome the domain shift between the source domain used for training and the target domain where testing takes place. However, current DA methods assume that the entire target domain is available during adaptation, which may not hold in practice. We introduce a new, data-constrained DA paradigm where unlabeled target samples are received in batches and adaptation is performed continually. We propose a novel source-free method for continual unsupervised domain adaptation that utilizes a buffer for selective replay of previously seen samples. In our continual DA framework, we selectively mix samples from incoming batches with data stored in a buffer using buffer management strategies and use the combination to incrementally update our model. We evaluate and compare the classification performance of the continual DA approach with state-of-the-art (SOTA) DA methods based on the entire target domain. Results on three popular DA datasets demonstrate the benefits of our method when operating in data constrained environments. We also conduct experiments for continual domain adaptation to multiple sequential target domains, and our method performs favorably against the SOTA methods. We further extend our work to address a gradually evolving target domain fragmented into multiple sequential batches where the model continually adapts to the gradually varying stream of data in an unsupervised manner. To tackle this challenge, we incorporate a contrastive loss is for better alignment of the buffer samples and the continual stream of batches. Our experiments on the rotating MNIST and CORe50 datasets confirm the benefits of our unsupervised continual learning method for gradually varying domains as well. We also synthesize two gradually worsening weather conditions on real images from

two existing aerial imagery datasets, generating a total of four benchmark datasets for evaluating continual domain adaptation under gradually varying weather conditions. The combination of the constraints of continual adaptation, and gradually deteriorating weather conditions provide the practical DA scenario for aerial deployment in unmanned aerial vehicles and drones. We evaluate our continual models with both convolutional and transformer architectures for comparison. We discover potential stability issues during adaptation for our buffer-fed continual DA methods, and offer gradient normalization as a simple solution to curb training instability.

4.1 Introduction

Domain adaptation (DA) methods based on deep learning have received significant attention in recent years for mitigating the domain shift from the source domain used for training to the target domain where inference takes place [31, 48, 90, 111, 125, 201]. In closed-set, unsupervised domain adaptation (UDA), the target domain is not labeled, and the same classes are present in the source and target domains. The distribution shift between the source domain data and target domain data causes a drop in classification accuracy. Many of the popular deep learning based DA methods [21, 31, 108, 139] employ adversarial training using both the source and target data to learn domain agnostic features [48], or to align the feature spaces of the source and target domains [201]. Inspired by Hypothesis Transfer Learning (HTL) [109], some recent methods transfer only the source trained model for target adaptation [106, 111, 125], significantly reducing the data storage footprint.

Current DA methods operate under the assumption that the entire target dataset is available during adaptation, which may not be feasible in practice. For example, when a robot or an autonomous vehicle is deployed in a new environment, it is unreasonable to expect all data from the new drastically different environment to be available at the same time. This inspires a new DA paradigm where the deployed model is updated continually as new data arrive in small batches, as depicted in Fig 4.1. In this work, our model is initially trained using source domain data and is then deployed in a new domain where target data are collected incrementally in small batches and the model adapts continually.

In a related approach, Hoffman et al. [68] proposed a manifold-based method that deals with streaming target data from an evolving target domain that is changing slowly. Bitarafan et al. [10] used a semi-supervised method for target adaptation under the assumption that there is no drastic domain shift between the source and the first sequence of the target domain or between consecutive sequences of the evolving target domain. Wulfmeier et al. [212] proposed a generative adversarial network based continual domain adaptation method for a gradually changing target domain. A meta learning approach was presented in [131]

to learn the representation of continuously evolving domains to avoid catastrophic forgetting. Moon et al. [152] proposed a two-step adaptation process where the first step aligns the incoming target sequence with earlier target sequences via a mean-target transformation matrix to reduce the distribution discrepancy between target sequences. However, these methods were not applied to standard DA datasets, and assumed that there was no sudden domain shift between the source and target domains or between two consecutive time instances within the target domain. In our continual DA framework, the shift between the source and target can be sudden due to differences between the two domains, and the target distribution may be significantly different than the source distribution.

In another approach, Volpi et al. [205] proposed domain adaptation to continual time varying domains with a significant domain shift between the source and target domains. A meta-learning approach with auxiliary meta domains was used to avoid forgetting during adaptation. However, this work assumed that each target domain was available at once, which does not accurately represent real-world scenarios. It also lacked comparison with standard domain adaptation benchmarks.

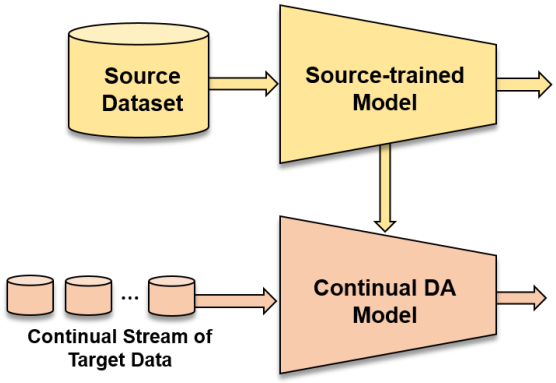


Figure 4.1: Continual DA paradigm where initial training is performed with source domain labeled data and the trained model is deployed in the target domain. During deployment, unlabelled target domain data are received in streaming batches and the model is continuously adapted with each new batch of target data.

In contrast, we present a scenario where the target distribution is not directly related to the source distribution and the target data are received in a series of smaller batches over time, as shown in Fig. 4.1. Our approach is broader in scope, introduces a deep learning framework, and our results are reported on standard DA datasets, making comparison with existing DA methods possible.

Continual learning (CL) can be broadly categorized into two major paradigms [164]. The first one deals with *incremental batch learning*, where a labeled dataset is fragmented into multiple distinct batches. Tasks are incrementally added while training and the model is allowed to train over multiple epochs for each new batch of data [15, 60, 97]. Once a batch is processed, it is discarded and the next batch is provided to the

learner. The other paradigm is *streaming learning* which is a special case of incremental batch learning, where a single training sample is fed to the learner and the model is only allowed to train for one epoch on the entire training dataset [46, 47].

In this work, we consider a *batch streaming* approach to split the target domain data into small, distinct, unlabeled batches that are input sequentially to the network. Our model is allowed to adapt over only *one epoch* of each incoming batch. Once a batch is processed, the next batch is fetched and provided to the network. We note that this paradigm is more challenging than standard domain adaptation, due to the streaming nature of processing and limited availability of target domain data during adaptation. This setting also differs from the existing CL paradigms, due to lack of labels for the target samples and the challenge of dealing with drastic domain shift from the source dataset to the target dataset.

Various techniques have been developed to overcome the challenges in CL settings. Three major techniques are: partial replay, Elastic Weight Consolidation (EWC), and distillation [59]. In partial replay, training samples are selectively stored and replayed during learning along with the incoming batch of samples. This procedure shows strong results for supervised CL for image classification [15, 71, 171, 211]. In EWC, the weights of the network are regularized by a quadratic term to enforce minimum change between already learned weight and weights updated from a new batch [101]. In the distillation procedure, soft labels for the distillation loss are computed for a new batch with the already learned weights from the previous batches, and the distillation loss [66] is optimized along with the classification loss.

To tackle the continual DA problem, we take inspiration from CL methods [58, 171, 211] that utilize episodic memory replay and propose the Continual Domain Adaptation (**ConDA**) framework that includes a buffer to hold processed target samples and their predicted labels, and buffer management strategies to selectively store and replay previously seen target samples. Furthermore, our method incorporates features with better generalization capabilities that improve upon the performance of the state-of-the-art (SOTA) source-free DA methods.

The proposed ConDA approach continually adapts the source model to the target domain as data arrive in batches, which greatly reduces the data storage requirements. Our method does not require any source data during adaptation, and additionally does not need to store the whole target domain at any time. During adaptation, ConDA only requires the incoming batch of target data along with the data stored in the buffer. This data constrained setting makes our work useful for edge AI systems, where neither the entire target domain data are fully available at the same time, nor storing all previously encountered data is feasible.

We evaluate several buffer configurations, along with specific loss functions for continual adaptation, and propose a buffer management strategy and associated adaptation procedure that is well-suited for continual

DA. ConDA outperforms many standard (non-continual) DA methods that utilize the full target domain, yet it operates at a fraction of their data storage footprint.

We also investigate continual domain adaptation when the target domain is dynamic (gradually evolving), and each batch of data is presented only once to the network. Fast and dynamic adaptation is a key challenge for such a case. For this task, we propose **UCL-GV**, a novel method based on selectively storing samples in a buffer and replaying them when a new batch of samples is fetched, similar to ConDA. To mitigate the small domain shift between the existing buffer samples and the incoming batch samples, due to the gradually varying nature of the target data, we propose to perform alignment using a contrastive loss.

We further extend our work to specifically deal with continual DA on aerial imagery, where the target data distribution is gradually shifting away from that of the source data due to inclement weather. We propose four benchmark datasets for assessing domain adaptation on aerial images, given the domain shifts are gradual. We consider degradation types of cloud cover layer, and snowfall layer on two widely used aerial image datasets AID [213] and UCM [223]. The descriptions of the datasets, the degradation types, and how they were created are described in Section 4.5.1. We then evaluate one standard source-free DA model [125] and our two continual DA models on our newly constructed aerial datasets. We discover that our continual DA models may suffer from stability issues, that not only harm optimal adaptation, but may potentially collapse the model, if left unaddressed. We propose the simple solution of normalizing gradients before model optimization to increase adaptation stability, and show empirical results to back our claim. We also replace the original ResNet-50 [63] with attention-based transformer networks Vision Transformer (ViT) [37] and Swin [133,135], and evaluate the models with the state-of-the-art feature extractors to explore the effect of stronger backbone architectures on continual DA.

The main contributions of our work are outlined below.

1. We introduce a new paradigm of continual unsupervised DA that operates under data-constrained conditions where batches of unlabeled target samples are received sequentially.
2. We propose a source-free DA framework named ConDA, which adapts continually to incoming batches of unlabeled target data by utilizing a buffer for selective replay of previously encountered samples.
3. We introduce equal diversity loss for effective adaptation across all classes.
4. Results indicate that although ConDA only has access to a small fraction of the target data at a time, it is superior to several domain adaptation methods that require access to the entire source and target datasets during adaptation.

5. We extend ConDA to continually adapt to multiple target domains one after the other, and demonstrate that our method performs well in such settings without significant forgetting.
6. We further incorporate dynamic or gradually varying target domains, and propose UCL-GV that utilizes a contrastive loss in addition to the replay strategy in ConDA.
7. We synthesize 4 new benchmark datasets for replicating continual domain adaptation under gradually degrading weather, and evaluate our continual models and a standard DA model.
8. We evaluate the effect of transformer network backbones on continual DA for our newly synthesized changing weather aerial datasets, and also propose the simple solution of gradient normalization to stabilize the adaptation process.

The rest of the paper is organized as follows: In Section 4.2, we discuss existing research in the field of domain adaptation, continual learning, continual domain adaptation, and domain adaptation on aerial imagery. In Section 4.3, we introduce the methodology of the continual framework for ConDA, discuss the standard DA datasets we use to evaluate ConDA, and the obtained results. In Section 4.4, we describe the adaptation process for UCL-GV, and the gradually varying domain adaptation datasets used to evaluate the model, and corresponding results. In Section 4.5, we present our novel gradually varying weather aerial datasets, and discuss our observations on the continual DA methods using the datasets. In Section 4.6, we present final remarks on continual DA based on our evaluations.

4.2 Related Work

4.2.1 Domain adaptation

A domain gap manifests due to the dataset bias when the data distributions in the source and target domains are significantly different [200]. Many unsupervised DA (UDA) techniques have been proposed to mitigate this domain gap for computer vision tasks, such as object detection and semantic segmentation [20, 98, 125]. Long et al. [138] and Tzeng et al. [202] proposed minimizing the maximum mean discrepancy (MMD) for UDA. Zellinger et al. [229] proposed minimizing central moment discrepancy (CMD) by matching higher order central moments of probability distributions in the source and target data. Ganin et al. [49] aligned distributions of source and target domains via an adversarial domain discriminator. Many other methods since then have implemented aligning latent spaces adversarially [140, 166]. Tzeng et al. [201] adversarially aligned features of source and target domain data while transferring the source domain classifier to the target domain. Pan et. al. [163] trains a separate source classifier with labels and a separate target classifier with

pseudo-labels, and aligns the score distributions of the individual classifiers to enforce prediction consistency across domains. Likewise, generative models have also been employed to create source-like images at the pixel level for domain adaptation [239]. However, during such adversarial alignment, the intrinsic target feature discrimination may get lost, leading to suboptimal performance. Tang et. al. [194] addresses this clustering the target features by regularizing using the source feature distribution. Ruijia et al. [218] proposed to adapt the feature norms of source and target domains to a large range of scalers, thus facilitating a reliable knowledge transfer from the source domain.

Adversarial methods require access to source data at the time of adaptation, but this is likely to create issues related to storage requirements or privacy when sharing of sensitive and private data. Domain adaptation research has been exploring such practical scenarios where adaptation is done without using source data. Source-free UDA methods consist of an initialization stage with access to source data for training and an adaptation stage with access only to the target data without any of the source data [107]. Chidlovskii et al. [27] proposed a semi-supervised source-free DA framework where no source domain data are available during adaptation, but some representation of the source domain is available, such as class means or a few annotated target samples. Liang et al. [124] identified a subspace where target and source centroids are only modestly shifted and used class-wise distribution estimator of the source data to conduct distant supervision for target adaptation. An end-to-end, source-free DA method based on information maximization was proposed in [125].

4.2.2 Continual learning

Mammals, as opposed to artificial neural networks trained within the standard deep leaning framework, learn continuously so that their intelligence increases gradually over time. When neural networks are subjected to such a process, they run the risk of catastrophic forgetting, where they forget the knowledge gained in earlier training stages [146]. Continual or lifelong learning methods have proposed a few mechanisms to mitigate catastrophic forgetting in deep neural networks. Among them, the most prominent are (i) replay of previously seen data [58,171,211], (ii) constraining network parameter updates according to a regularization scheme [101, 123, 230], and (iii) network expansion with increasing data [70, 179, 227]. Memory replay mimics the mechanism of the human brain, where during both the sleeping [89] and awake [94] phases, past experiences are regenerated from encoded representations and the neocortex is trained on them [162, 191]. Rebuffi et al. first applied memory replay in iCaRL [171], for class-incremental learning in the context of neural networks, where 20 raw samples from each class were stored for later replay. More recent replay methods extended iCaRL to make it end-to-end trainable [15], introduced a loss function to correct for class bias [211], and stored mid-level features instead of raw images to reduce storage footprint [58].

Regularization based models learn new tasks incrementally, while preserving knowledge from previous tasks by varying the plasticity of the network's convolutional filter weights, which are significant for retaining earlier knowledge. Kirkpatric et al. [101] proposed to selectively lower the learning rate from one task to the next. Z. Li and D. Hoiem [123] proposed to regularize the network updates using the network outputs from the original model for the new task data.

In this work, we mainly draw from the concept of memory replay. We present a way to continually adapt a source trained model to a new target domain when the target data are received in small batches. This is an area of domain adaptation that, to the best of our knowledge, has not yet been explored.

4.2.3 Continual domain adaptation

Existing DA research formulates the problem of continual domain adaptation in primarily two major ways: gradually evolving domain shift [10, 11, 68, 105], and sudden domain shift [170, 175, 195] between the source and target domains. Rostami et al. [175] proposed a continual DA technique where multiple target domains are sequentially fetched by the network. To mitigate catastrophic forgetting due to domain shift, the method proposed to selectively store raw samples and replay them with the samples from the next domains. A Gaussian Mixture Model was utilized to consolidate the distribution of already learned domains. Rakshit et. al. [170] similarly proposed a continual domain adaptation approach named FRIDA across multiple sequential target domains, where an entire domain is made available at each time step. A Generative Adversarial Network (GAN) was proposed in conjunction with an existing domain adaptation approach to learn the domain distribution of each domain and produce samples for replay in future time steps. This method showed effectiveness in mitigating catastrophic forgetting. A similar formulation is used by [195], where EWC is used to mitigate catastrophic forgetting.

The work in [105] proposed an UDA method for an evolving target domain. The sequential gradually varying data were split into three different domains: a source domain, an intermediate domain, and a target domain. The intermediate domain was introduced to represent the gradually evolving nature of the data, rather than having a drastic domain shift between the source and the target domains. A meta learning approach was proposed for continual adaptation. Following [105], the work in [18] proposed to perform domain adaptation without having the sequential indexes of the intermediate domains.

Our work differs from [170, 175] in terms of the continual settings considered. The methods in [170, 175] consider transitions across multiple target domains, but for each new target domain, the setting is similar to that of standard DA where the entire target domain is made available for adaptation. In our setting, we consider continual adaptation within each domain, in addition to transitions across multiple target domains.

At a given time, ConDA only has access to a small batch of unlabeled samples from the target domain, instead of the entire target domain. This batch streaming setting makes adaptation more challenging, but results in a more efficient system that requires a much smaller data storage footprint.

Our proposed setting of for continual domain adaptation to gradually evolving domains also has two major differences from [18, 105]. First, each batch of data from the intermediate and target domains are only fed once rather than multiple times as proposed in [18, 105]. Second, both the source data and the intermediate/target data are required during meta training, while ours is a more realistic source-free adaptation setting to address the constraints in data access or privacy concerns.

4.2.4 Domain adaptation on aerial imagery

Although DA has been extensively studied for ground-level imagery, few studies have explored DA on aerial images. Nagananda et al. [159] and Xu et al. [215] evaluated the state-of-the-art standard (non-continual) DA methods on aerial datasets. Nagananda *et al.* [159] created three pairs of aerial datasets for DA based on common class labels. However, both works [159, 215] dealt with standard DA settings, with sudden and drastic domain shift between the source and the target domains, and did not consider gradually varying domains. To the best of our knowledge, continual domain adaptation has not yet been studied within the scope of remote sensing datasets and there are no aerial datasets that could be utilized to assess continual DA on gradually changing environments. In this work, we prepare four continually varying weather condition aerial datasets, and evaluate ConDA and UCL-GV on these benchmarks.

4.3 ConDA

4.3.1 Method

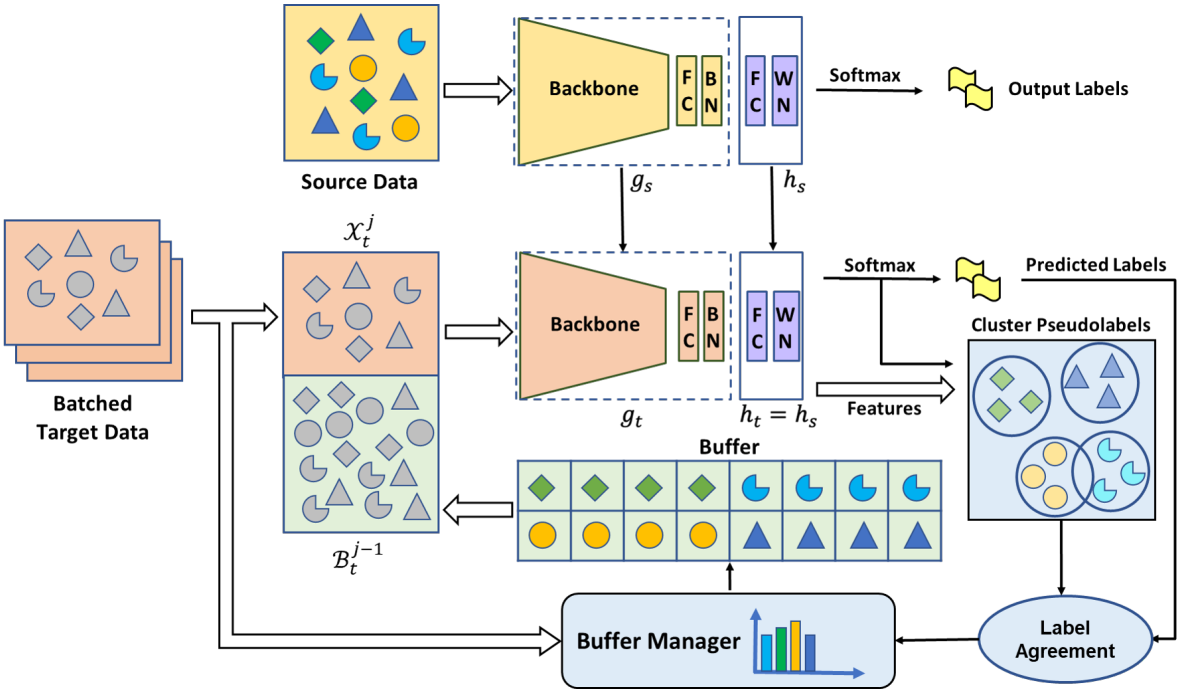


Figure 4.2: Proposed ConDA framework adapting on target domain data that arrive in small batches. A subset of the samples that are already seen by the network are stored in a buffer for replay with the incoming batches. The buffer manager is responsible for selecting the samples that populate the buffer. The incoming target samples are mixed with the current buffer samples and sent to the network for adaptation.

Let us denote the source domain as $\mathcal{D}_s$ with labelled samples $\{x_s^i, y_s^i\}_{i=1}^{n_s}$, where n_s is the total number of samples $x_s^i \in \mathcal{X}_s$, and corresponding labels $y_s^i \in \mathcal{Y}_s$. The target domain is $\mathcal{D}_t$ with n_t unlabeled samples $\{x_t^i\}_{i=1}^{n_t}$ and $x_t \in \mathcal{X}_t$. In closed-set UDA, the classes $\mathcal{C}_s$ present in the source domain are the same as the classes $\mathcal{C}_t$ present in the target domain, and the task is to predict the target labels $\{y_t^i\}_{i=1}^{n_t}$ where $y_t \in \mathcal{Y}_t$.

In the continual DA setting, the target domain $\mathcal{D}_t$ is randomly divided into m i.i.d. batches, i.e., $\mathcal{X}_t = \{\mathcal{X}_t^1, \mathcal{X}_t^2, \mathcal{X}_t^3, ..., \mathcal{X}_t^m\}$ with samples $\{x_t^{j,i}\}_{j=1,i=1}^{m,n_t^j}$ where n_t^j is the number of i.i.d. samples in the j^{th} batch and $j \in \{1, 2, 3, ..., m\}$. Operating in a data-constrained environment, the source trained model $f_s : \mathcal{X}_s \rightarrow \mathcal{Y}_s$ has access to only a batch of unlabeled target samples $\mathcal{X}_t^j$ at a time and our objective is to learn a target model $f_t : \mathcal{X}_t^j \rightarrow \mathcal{Y}_t^j$ where $\mathcal{Y}_t^j$ represents the predicted labels of $\mathcal{X}_t^j$.

The continual DA scenario runs the risk of the model overfitting to the current batch of target samples and

failing to adapt to the marginal distribution of the entire target domain due to the continual nature of the incoming samples. Therefore, our task is to reduce the performance gap between the model that is adapted based on continuous batches of target data, i.e., $f_t : \mathcal{X}_t^m \rightarrow \mathcal{Y}_t^m$ and the model that is adapted given the entire target domain simultaneously (standard DA framework), i.e., $f_t : \mathcal{X}_t \rightarrow \mathcal{Y}_t$, both evaluated on the full target domain $\mathcal{X}_t$.

In our continual adaptation setting, the network is continually fed with small incoming batches of target domain data $\mathcal{X}_t = \{\mathcal{X}_t^1, \mathcal{X}_t^2, \mathcal{X}_t^3, ..., \mathcal{X}_t^m\}$, as illustrated in Figure 4.2. After processing each $\mathcal{X}_t^i$ batch, few samples are selectively stored in a memory buffer according to our buffer management strategy, and the samples that are not stored in the buffer are discarded. The buffer configuration is described in Section 4.3.1, and the details of our buffer management scheme are given in Section 4.3.1. When the next batch of target domain data is received, the existing buffers samples are combined with the new incoming batch samples and adaptation is performed on the combined set of samples over only one pass. This process of storing samples in a buffer and replaying them with incoming batches continues until all the target batches are continually fed into the network. Since only one pass of the combined set of incoming batch samples and existing buffer samples takes place, the total number of passes during the whole adaptation process is equal to m, the total number of incoming target batches.

Our ConDA framework for continual adaptation is shown in Fig. 4.2. The source model $f_s(x) = h_s(g_s(x))$ consists of two parts: a feature generator model g_s, consisting of a backbone and a fully-connected (FC) layer followed by a batch normalization (BN) layer, and a hypothesis model h_s that includes a fully connected layer and a weight normalization (WN) layer [125]. Inspired by [125], we train the source model f_s in a supervised manner with label smoothing [156]. During target adaptation, the target hypothesis model is set to the source model, $h_t = h_s$, and the parameters remain unchanged during adaptation. The target feature extractor g_t is initialized with the source feature extractor model g_s and adapts continually with incoming batches of target samples.

Our continual adaptation setting can further be extended to multiple target domains $\{\mathcal{D}_{t_1}, \mathcal{D}_{t_2}, \cdots, \mathcal{D}_{t_\tau}\}$ where $t_1, t_2, \cdots, t_\tau$ are sequential time steps when samples from various distinct domains are fetched as shown in Figure 4.3. Each target domain is fragmented into multiple i.i.d. batches as mentioned above for a single domain.

Buffer

We introduce a buffer $\mathcal{B}_t$ with states $\{\mathcal{B}_t^1, \mathcal{B}_t^2, ..., \mathcal{B}_t^m\}$ each corresponding to m batches of target data to conduct continual domain adaptation. We maintain a class-balanced $\mathcal{B}_t$, i.e., an equal number of buffer

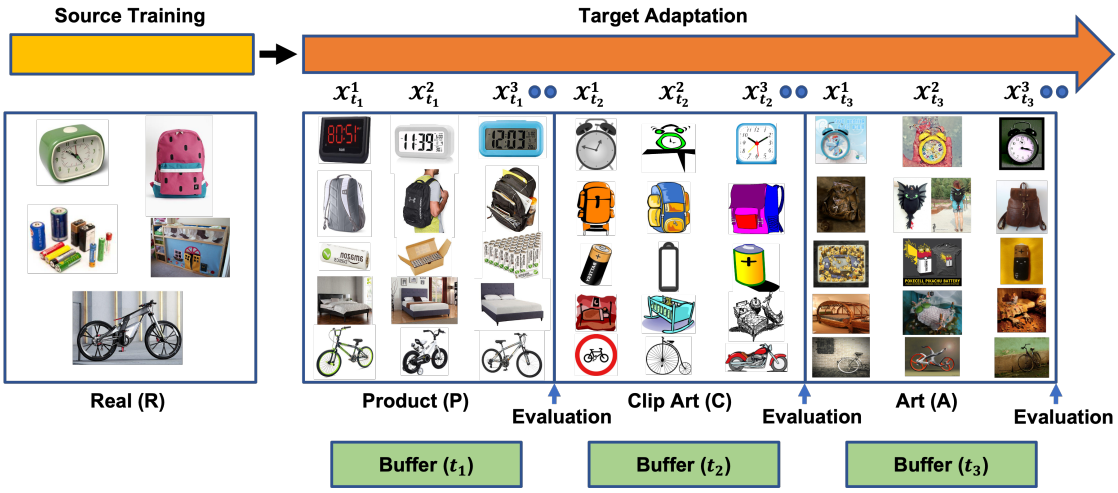


Figure 4.3: Continual adaptation for multi-target domains. For demonstration purpose, we consider 5 categories from the Office-Home dataset. The network processes each batch $\mathcal{X}_{t_i}$ only once along with the replayed buffer samples. When samples from target domain $\mathcal{D}_{t_1}$ end, batch samples from new domain $\mathcal{D}_{t_2}$ start to be fetched by the network and the same process continues until the last domain $\mathcal{D}_{t_\tau}$ is fetched.

slots are allocated for each class calculated from buffer length and the number of classes present in the target domain assuming that $\mathcal{C}_t = \mathcal{C}_s$. The buffer is populated after the network is trained on a batch of target samples. The buffer stores the samples and their corresponding class labels predicted by the network. Our model only requires access to the samples stored in the buffer for subsequent adaptation along with new target batches that arrive. The sample selection process to populate the buffer is handled by a buffer manager discussed in the following section.

Buffer Manager

The network is adapted on a batch $\mathcal{X}_t^j$ and outputs $\{\mathcal{Y}_t^j, \mathcal{U}_t^j\}$ where $\mathcal{U}_t$ is the softmax classification score. We compute the soft labels $\mathcal{V}_t^{j-1}$ for the buffer samples with the current state of the model $f_t : \mathcal{B}_t^{j-1} \rightarrow \mathcal{V}_t^{j-1}$. The buffer manager takes in $\{\mathcal{X}_t^j, \mathcal{Y}_t^j, \mathcal{U}_t^j, \mathcal{B}_t^{j-1}$, and $\mathcal{V}_t^{j-1}\}$ and outputs $\mathcal{X}_t' \subseteq \mathcal{X}_t^j \bigcup \mathcal{B}_t^{j-1}$ and corresponding labels to populate the buffer state $\mathcal{B}_t^j$.

At first, both the batch and buffer samples are filtered based on the softmax prediction and clustering pseudo labels. Only the samples for which there is a match between the softmax label and pseudo label are retained. Then the incoming batch samples are grouped based on the output label $\mathcal{Y}_t^j$, and samples of each class are sorted based on the confidence $\mathcal{U}_t^j$. Then, the buffer manager only picks the high confidence samples if the number of samples for any class exceeds the allotted number of slots for that class in the buffer. Finally, if

available, the remaining space for that class is filled with randomly drawn samples from $\mathcal{B}_t^{j-1}$ of that class.

Several buffer selection mechanisms have been proposed for supervised CL. Popular techniques include uniform random [15, 16], minimum logit distance [16], minimum confidence [59], minimum margin [59], maximum loss [59], maximum time since last replay [59] and minimum replays [59]. However, during unsupervised DA, the true labels are not available and these methods are not suitable except for the uniform random selection. The reason is intuitive, because for the supervised case when the true labels are available, it is beneficial to replay the low confidence samples to train the network to identify class boundaries. But since we compute pseudo labels via clustering in unsupervised DA, the low confidence samples may be assigned incorrect labels and training through replaying such samples will create more confusion and result in performance reduction.

We conducted multiple experiments with various buffer selection techniques, such as choosing the incoming samples randomly, or selecting the buffer samples based on the cosine distance to the nearest self-supervised cluster centers. We did not find any significant performance variation with various buffer sample selection techniques. We found a slight increase in performance with the sample selection mechanism based on the higher confidence scores.

When multiple domains are fetched by the network, we store $\mathcal{B}_{t_1}, \mathcal{B}_{t_2}, \cdots, \mathcal{B}_{t_\tau}$ for τ target domains and randomly replay $\mathcal{R}$ samples with uniform probability from the existing buffer. The replay samples $\mathcal{R}$ are selected from the entire buffer samples available at any instance, e.g., at step 1, $\mathcal{R} \in \mathcal{B}_{t_1}$, at step 2, $\mathcal{R} \in \mathcal{B}_{t_1} \bigcup \mathcal{B}_{t_2}$, and at final step τ, $\mathcal{R} \in \mathcal{B}_{t_1} \bigcup \mathcal{B}_{t_2} \bigcup \cdots \bigcup B_{t_\tau}$. This process ensures that samples from all of the domains previously seen by the network are provided in conjunction with the current batch of samples. The multi-target continual adaptation process is demonstrated in Fig. 4.3.

In the $(j+1)^{th}$ batch, the current buffer samples $\mathcal{B}_t^j$ and the incoming batch samples $\mathcal{X}_t^{j+1}$ are appended and provided to the network. We do not use any label information of the buffer samples when they are concatenated with the incoming batch samples. During adaptation with the incoming batch and buffer samples, we performed clustering to compute pseudo labels. The clustering technique is described next.

Clustering

Several clustering-based pseudo-labelling approaches [14, 125, 194, 208] have been explored in literature for unlabelled target data. We adopted a self-supervised clustering method introduced in [125] as an extension of the Deep Cluster [14] method. The combination of the batch and the buffer samples is denoted as $\mathcal{X}_t^* = \mathcal{X}_t^j \bigcup \mathcal{B}_t^{j-1}$. The initial estimate of the cluster centers is obtained by utilizing the softmax output of the input

target samples as follows.

$$c_k^{(0)} = \frac{\sum_{x_t \in \mathcal{X}_t^*} \hat{f}_t(x_t)\hat{g}_t(x_t)}{\sum_{x_t \in \mathcal{X}_t^*} \hat{f}_t(x_t)} \tag{4.1}$$

After computing the initial estimate of the centroids, the initial estimate of the pseudo labels $\hat{y}_t^{(0)}$ is found using the cosine distance function.

$$\hat{y}_t^{(0)} = \underset{k}{\operatorname{argmin}}\, d(\hat{g}_t(x_t), c_k^{(0)}) \tag{4.2}$$

where $d(\cdot, \cdot)$ is the cosine distance function. After computing the initial estimates of the pseudo labels, the cluster centers are recomputed as follows.

$$c_k^{(1)} = \frac{\sum_{x_t \in \mathcal{X}_t^*} \mathbb{1}(\hat{y}_t = k)\hat{g}_t(x_t)}{\sum_{x_t \in \mathcal{X}_t^*} \mathbb{1}(\hat{y}_t = k)} \tag{4.3}$$

where $\mathbb{1}(\cdot)$ is the indicator function. The final pseudo labels are computed using the updated cluster centers.

$$\hat{y}_t^{(1)} = \underset{k}{\operatorname{argmin}}\, d(\hat{g}_t(x_t), c_k^{(1)}) \tag{4.4}$$

where $\hat{y}_t^{(1)} \in \hat{\mathcal{Y}}_t^*$.

Adaptation Objective Function

For our objective function, we consider the information maximization (IM) loss from [52, 74, 125, 187] to produce individually precise predictions, while maintaining a global diversity of the network outputs. The IM loss is a combination of the entropy loss $\mathcal{L}_{ent}$ and equal diversity loss $\mathcal{L}_{eqdiv}$ functions shown below.

$$\begin{aligned} \mathcal{L}_{ent}(f_t; \mathcal{X}_t) &= -\mathbb{E}_{x_t \in \mathcal{X}_t^*} \sum_{k=1}^{C_s} \sigma_k(f_t(x_t)) \log(\sigma_k(f_t(x_t))) \\ \mathcal{L}_{eqdiv}(f_t; \mathcal{X}_t) &= \sum_{k=1}^{C_s} q_k \log\left(\frac{q_k}{\hat{q}_k}\right) \end{aligned} \tag{4.5}$$

where $\sigma_k(a) = \frac{exp(a_k)}{\sum_i exp(a_i)}$ is the softmax function. Since we maintain a class-balanced buffer, we take q_k as the ideally uniform mean response, such that q_k is a C_s dimensional vector with all values of $1/C_s$ and $\hat{q}_k = \mathbb{E}_{x_t \in \mathcal{X}_t^*}[\sigma(f_t(x_t))]$ is the mean of the softmax output for the incoming target batch and buffer samples.

The equal diversity loss L_{eqdiv} attempts to make network predictions equally diverse for all classes and is calculated as the KL divergence between the ideal uniform distribution and the softmax distribution from the network outputs. Additionally, $f_t(x_t) = h_t(g_t(x_t))$ is a C_s-dim output for each target sample.

We further minimize $\mathcal{L}_{ce}$, the cross-entropy loss for the target samples, as shown below.

$$\mathcal{L}_{ce}(f_t; \mathcal{X}_t) = \mathbb{E}_{x_t \in \mathcal{X}_t^*, \hat{y}_t \in \hat{\mathcal{Y}}_t^*} \sum_{k=1}^{C_s} \mathbf{1}_{[k=\hat{y}_t]} \log(\sigma_k(f_t(x_t))) \tag{4.6}$$

Our final objective function therefore becomes,

$$\mathcal{L}(g_t) = \mathcal{L}_{ent} + \gamma_1 \mathcal{L}_{eqdiv} + \gamma_2 \mathcal{L}_{ce} \tag{4.7}$$

where γ_1 and γ_2 are hyper-parameters.

We present Algorithm 1 to demonstrate the overall procedure of our proposed method for multi-domain experiments.

4.3.2 Experimental Setup

Datasets

We use three commonly used DA benchmarks for our experiments: Office [180], Office-Home [204] and VisDA-C [167]. **Office-31** is a small-scale DA dataset consisting of images of 31 classes of common objects found in an office across 3 domains viz. Amazon (A), Webcam (W), and DSLR (D). **Office-Home** is a medium-sized DA dataset consisting of 4 domains viz. Art (A), Clipart (C), Product (P), and Real-World (R). The dataset contains images of 65 classes of items found in office and home environments. **VisDA-C** is a large-scale dataset consisting of 12 classes of objects across two domains: Synthetic (S) and Real (R). The 152K synthetic images are generated by 3D rendering and taken as the source domain. The 55K real samples are taken from MS COCO dataset [128] and taken as the target domain.

For our multi domain experiments, we utilize the **Office-Caltech** [54] and **Office-Home** datasets. Office-Caltech has 4 domains, Caltech (C) is added as a domain in addition to the three domains of the Office dataset. The Office-Caltech dataset has 10 shared classes.

Algorithm 1: ConDA Algorithm

Input : Trained source model $f_s : \mathcal{X}_s \rightarrow \mathcal{Y}_s$, streaming batches of target data $\{\mathcal{X}_{t_1}^1, \mathcal{X}_{t_1}^2, \cdots \mathcal{X}_{t_1}^m\} \bigcup \cdots \bigcup \{\mathcal{X}_{t_\tau}^1, \mathcal{X}_{t_\tau}^2, \cdots \mathcal{X}_{t_\tau}^m\}$ from domains $\{\mathcal{D}_{t_1} \cdots \mathcal{D}_{t_\tau}\}$.

Output : A set of models $\{f_{t_1}, f_{t_2}, \cdots, f_{t_\tau}\}$ after continually adapting on each domains $\{\mathcal{D}_{t_1} \cdots \mathcal{D}_{t_\tau}\}$.

Init. : The target model f_{t_1} is initialized with the source trained model f_s. The feature extraction network is set to trainable on the target data while keeping the hypothesis (classification) network frozen throughout the entire adaptation process.

for $i \leftarrow 1$ **to** τ ; `/* τ = number of target domains */`
do
  Get the target samples: $\{\mathcal{X}_{t_i}^1, \mathcal{X}_{t_i}^2, \cdots \mathcal{X}_{t_i}^m\}$ from $\mathcal{D}_{t_i} \in \{\mathcal{D}_{t_1} \cdots \mathcal{D}_{t_\tau}\}$;
  for $j \leftarrow 1$ **to** m ; `/* m represents the number of continual batches in` $\mathcal{D}_{t_i}$ `*/`
  do
    if $i = 1$ & $j = 1$ **then**
      $X \leftarrow \mathcal{X}_{t_i}^j$; `/* No buffer for the very first incoming batch */`
    else
      $X \leftarrow \mathcal{X}_{t_i}^j \bigcup \mathcal{B}_{t_i}^{j-1} \bigcup_{c=1, i \neq 1}^{i-1} \mathcal{B}_{t_c}$;
    end
    $\hat{Y} \leftarrow$ Compute pseudo labels for X;
    for $k \leftarrow 1$ **to** n_b ; `/* n_b = number of minibatches */`
    do
      Get i.i.d batch samples from $(X, \hat{Y})$;
      Optimize model f_{t_i} using Eq. 4.7;
    end
    $\mathcal{B}_{t_i}^j \leftarrow$ Fill buffer with samples $\{\mathcal{X}_{t_i}, \mathcal{B}_{t_i}^{j-1}\}$;
  end
  Evaluate f_{t_i} on test samples from domains $\{\mathcal{D}_{t_1} \cdots \mathcal{D}_{t_\tau}\}$;
  Store $\bigcup_{c=1}^{i} \mathcal{B}_{t_c}$; `/* Store the buffer for further adaptation to newer domains for multi-domain adaptation. */`
end

Continual Dataset Setup

In our continual DA experiments on Office-31, the buffer size is set to 124 samples (four samples per class when fully balanced) and the number of incoming samples in each batch is set to 32 samples. For our experiments on Office-Home, the buffer size is set at 520 (eight samples per class when fully balanced) and the incoming batch size contains 64 samples. In the case of VisDA-C dataset, the experiments are conducted with a buffer size of 96 samples (four samples per class when fully balanced) and the number of samples in each incoming batch was set to 32 samples. For all the datasets, each time a new incoming batch of data arrives, our model is trained for only 1 epoch of the memory buffer and incoming batch.

For the multi-domain experiments, we consider the Office-Home dataset with domain sequence $R \rightarrow P \rightarrow C \rightarrow A$, and the Office-Caltech with domain sequence $A \rightarrow D \rightarrow W \rightarrow C$, consistent with [170]. For each domain, 70% of the samples are randomly drawn as adaptation samples and remaining 30% samples are set aside for testing. For all datasets, all the source domain samples are used for source training. The target domains are fragmented into batch sizes of 32 samples for Office-Caltech and 128 samples for Office-Home. For both Office-Caltech and Office-Home, tests are performed with a buffer size such that the buffer holds a maximum of 4 samples per class per domain. Therefore, for Office-Home, the buffer size is set at 260 samples per domain, i.e. 260 for the first domain, and another 250 for the second domain, and so on. For Office-Caltech dataset, the buffer size is set at 40 samples per domain. The number of samples that are randomly chosen from the multi-domain buffer for replay at each iteration is limited to 128 or fewer in all cases.

Implementation Details

We use ResNet50 [63] as the common backbone for all our models except for VisDA-C dataset for which we use ResNet101, along with a bottleneck fully connected (FC) layer with 256 units and a batch normalization layer, as shown in Fig. 4.2, followed by a final task-specific FC classifier and weight normalization layer, respectively [125].

We train our network with stochastic gradient descent (SGD) optimizer with 0.9 momentum. The learning rate for the layers after the ResNet backbone is set to 10 times the learning rate of the backbone. The learning rate for the backbone is set to $\eta_0 = 1e^{-3}$ for all datasets except for VisDA-C which has a learning rate of $\eta_0 = 1e^{-4}$. We also use a learning rate scheduler $\eta = \eta_0 \cdot (1 + 10 \cdot p)^{-0.75}$ where p changes from 0 to 1 as training progresses [125]. We empirically find that $\gamma_1 = 1$ and $\gamma_2 = 0.5$ work best for all of the datasets.

Evaluation Protocol

We calculate mean accuracy for the entire target domain for our single domain experiments to easily compare our results with existing SOTA methods. For multiple domains, we calculate three metrics: (a) Average accuracy, ACC, (b) Forgetting, FG, and (c) Forward Transfer, FW. The ACC and FG metrics are used for direct comparison with [170] while the FW metric is inspired by existing CL approaches [39, 59]. Considering transitions through τ target domains, we compute the mean accuracy on the test dataset of every target domain at each transition step of $t_1, t_2, \cdots, t_\tau$ which will provide an accuracy matrix $\mathcal{A} \in \mathbb{R}^{\tau \times \tau}$.

$$AC(\mathcal{D}_T) = \frac{1}{\tau} \sum_{i=1}^{\tau} \frac{1}{\tau - i + 1} \sum_{j \geq i}^{\tau} \mathcal{A}_{i,j} \tag{4.8}$$

$$FG(\mathcal{D}_T) = \frac{1}{\tau - 1} \sum_{i=1}^{\tau-1} \frac{1}{\tau - i} \sum_{j > i}^{\tau} \mathcal{A}_{i,j} - \mathcal{A}_{i,j-1} \tag{4.9}$$

$$FW(\mathcal{D}_T) = \frac{1}{\tau - 1} \sum_{i=2}^{\tau} \frac{1}{i - 1} \sum_{j < i}^{\tau} \mathcal{A}_{i,j} \tag{4.10}$$

Table 4.1: Mean accuracy of adaptation using the Office-31 dataset. The top part of the table shows results of traditional DA methods using the full target dataset. The bottom part of the table shows results for ConDA and SHOT using the continual setting. The ConDA experiments are performed with a continual batch size of 32 and buffer size of 124 (four samples per class).

Method	Target	A ⟶ D	A ⟶ W	D ⟶ A	D ⟶ W	W ⟶ A	W ⟶ D	Mean
ResNet50 [63]	Full	68.9	68.4	62.5	96.7	60.7	99.3	76.1
DAN [138]	Full	78.6	80.5	63.6	97.1	62.8	99.6	80.4
DANN [48]	Full	79.7	82.0	68.2	96.9	67.4	99.1	82.2
JAN [140]	Full	84.7	85.4	68.6	97.4	70.0	99.8	84.3
MADA [166]	Full	87.8	90.0	70.3	97.4	66.4	99.6	85.2
SAFN+ENT [218]	Full	92.1	90.3	73.4	98.7	71.2	100.0	87.6
ALDA [21]	Full	94.0	95.6	72.2	97.7	72.5	100.0	88.7
MDD+IA [90]	Full	92.1	90.3	75.3	98.7	74.9	99.8	88.8
GVB-GD [31]	Full	95.0	94.8	73.4	98.7	73.7	100.0	89.4
SRDC [194]	Full	95.8	95.7	76.7	99.2	77.1	100.0	90.9
SHOT [125]	Full	94.0	90.1	74.7	98.4	74.3	99.9	88.6
SHOT [125]	Cont.	84.74±0.00	85.32±0.07	69.77±0.18	97.86±0.00	65.50±0.16	99.20±0.00	83.73±0.05
ConDA	Cont.	84.74±0.00	88.68±0.58	72.75±0.93	98.20±0.39	70.04±0.92	99.80±0.00	85.70±0.09

The average model accuracy is denoted as ACC. We compute the average accuracy at each step when a new target domain is fetched $ACC(\mathcal{D}_{t_i})$. We average the ACC scores over all domains to obtain $ACC(\mathcal{D}_T)$. Forgetting, denoted as FG, is computed for every step except the last, and the FG scores over all domains

Table 4.2: Mean accuracy of adaptation using the Office-Home dataset. The top part of the table shows results of traditional DA methods using the full target dataset. The bottom part of the table shows results for ConDA and SHOT using the continual setting. The ConDA experiments are performed with a continual batch size of 128 and buffer size of 520 (eight samples per class).

Method	Target	Ar → Cl	Ar → Pr	Ar → Rw	Cl → Ar	Cl → Pr	Cl → Rw	Pr → Ar	Pr → Cl	Pr → Rw	Rw → Ar	Rw → Cl	Rw → Pr	Mean
DANN [49]	Full	45.6	59.3	70.1	47.0	58.5	60.9	46.1	43.7	68.5	63.2	51.8	76.8	57.6
ALDA [21]	Full	53.7	70.1	76.4	60.2	72.6	71.5	56.8	51.9	77.1	70.2	56.3	82.1	66.6
SAFN [218]	Full	54.4	73.3	77.9	65.2	71.5	73.2	63.6	52.6	78.2	72.3	58.0	82.1	68.5
MDD+IA [90]	Full	56.2	77.9	79.2	64.4	73.1	74.4	64.2	54.2	79.9	71.2	58.1	83.1	69.5
CADA-P [108]	Full	56.9	76.4	80.7	61.3	75.2	75.2	63.2	54.5	80.7	73.9	61.5	84.1	70.2
GVB-GD [31]	Full	57.0	74.7	79.8	64.6	74.1	74.6	65.2	55.1	81.0	74.6	59.7	84.3	70.4
SPL [208]	Full	54.5	77.8	81.9	65.1	78.0	81.1	66.0	53.1	82.8	69.9	55.3	86.0	71.0
SRDC [194]	Full	52.3	76.3	81.0	69.5	76.2	78.0	68.7	53.8	81.7	76.3	57.1	85.0	71.3
HDMI [111]	Full	57.8	76.7	81.9	67.1	78.8	78.8	66.6	55.5	82.4	73.6	59.7	84.0	71.9
SHOT [125]	Full	57.1	78.1	81.5	68.0	78.2	78.1	67.4	54.9	82.2	73.3	58.8	84.3	71.8
LDAuCID [175]	Full	48.3	67.4	74.1	48.7	61.9	63.8	49.6	42.1	71.3	60.3	47.6	76.6	59.4
SHOT [125]	Cont.	49.3±0.19	71.0±0.03	75.0±0.16	59.9±0.07	70.1±0.22	70.2±0.13	58.7±0.02	47.2±0.3	76.7±0.05	69.4±0.07	54.0±0.03	79.6±0.10	65.1±0.06
ConDA	Cont.	54.9±0.35	75.2±0.18	79.4±0.10	65.9±0.39	75.3±0.54	77.0±0.56	64.5±0.15	53.5±0.29	80.0±0.05	73.0±0.29	55.9±0.24	81.8±0.20	69.7±0.08

Table 4.3: Mean per class accuracy of adaptation using the VisDA-C dataset. The top part of the table shows results of traditional DA methods using the full target dataset. The bottom part of the table shows results for ConDA and SHOT using the continual setting. The ConDA experiments are performed with a continual batch size of 32 and a buffer size of 96 (eight samples per class).

Method	Target	plane	bycycl	bus	car	house	knife	mcycle	person	plant	sktbrd	train	truck	Per class
DANN [49]	Full	81.9	77.7	82.8	44.3	81.2	29.5	65.2	28.6	51.9	54.6	82.8	7.8	57.6
SAFN [218]	Full	93.6	61.3	84.1	70.6	94.1	79.0	91.8	79.6	89.9	55.6	89.0	24.4	76.1
ALDA [21]	Full	93.8	74.1	82.4	69.4	90.6	87.2	89.0	67.6	93.4	76.1	87.7	22.2	77.8
SHOT [125]	Full	94.3	88.5	80.1	57.3	93.1	94.9	80.7	80.3	91.5	89.1	86.3	58.2	82.9
SHOT [125]	Cont.	94.8±0.00	74.1±0.00	82.5±0.02	60.0±0.00	92.5±0.00	94.5±0.05	86.3±0.00	80.3±0.00	88.0±0.01	76.5±0.08	84.5±0.01	48.2±0.01	80.0±0.01
ConDA	Cont.	95.2±0.17	81.1±0.36	81.4±0.81	61.0±1.71	92.9±0.43	93.2±2.20	84.7±1.24	81.4±0.22	87.3±0.79	88.3±0.95	84.2±1.11	52.6±0.58	81.9±0.24

are averaged to obtain $FG(\mathcal{D}_T)$. A negative value of FG represents forgetting. The Forward Transfer is denoted as FW and is computed for every step except the first. We take the mean of the FW scores over all domains to obtain $FW(\mathcal{D}_T)$. This score represents how well the model generalizes to the unseen domains.

4.3.3 Results

Single Target Domain

The continual DA results for Office-31 are shown in Table 4.1. The SHOT [125] method struggles in the continual setting and performance drops by 4.87% compared to SHOT using the full target domain. ConDA outperforms continual SHOT by 1.97%. ConDA performs reasonably well compared to other SOTA standard DA methods [21, 90, 108, 218] that use the full target dataset, although ConDA requires only a fraction of the data and memory footprint that other methods need. Another constraint imposed on ConDA that contributes to loss in performance on this small dataset is the limit of one pass over the entire target dataset during adaptation.

For the Office-Home dataset, ConDA, while operating in a continual setting, outperforms some of the recent standard DA methods that access the full target dataset, such as ALDA [21], SAFN [218] and MDD+IA [90], and is on par with other SOTA DA methods. While SHOT [125] is one of the most effective methods on Office-Home, in the continual setting it loses its top performance by 6.7%. ConDA outperforms continual SHOT by 4.6% in terms of mean accuracy.

In the VisDA-C dataset, ConDA outperforms existing state-of-the-arts methods like ALDA [21] and SAFN [218]. SHOT [125] is the most accurate of the models with mean per class accuracy of 82.9%. However, continual SHOT is 2.9% less accurate than SHOT. ConDA outperforms continual SHOT by 1.9% and trails behind standard SHOT by only 1%. We attribute this to the fact that VisDA-C contains a large number (1,730) continual batches of target data which allows ConDA to approximate the performance of the baseline SHOT model.

Mutiple Target Domains

We further extend our continual adaptation experiments to multiple target domains and validate our method on multi-domain transitions using the Office-Caltech and Office-Home datasets. For comparison with SOTA under similar conditions, we consider the multi-domain adaptation results by FRIDA provided in [170] that are shown in Tables 4.5 and 4.5. DANN [49] is a standard unsupervised DA method developed for single

Table 4.4: Multi domain adaptation results for Office-Caltech dataset. The benchmark results are obtained from [170]. The continual experiments are done with an incoming batch size of 32 samples and a buffer size of 40 samples per domain (four samples per class per domain).

Method	$\mathcal{D}_{t_1}(D)$		$\mathcal{D}_{t_2}(W)$			$\mathcal{D}_{t_3}(C)$		$\mathcal{D}_T(Average)$		
	AC	FG	AC	FG	FW	AC	FW	AC	FG	FW
DANN [49]	94.44	0.00	82.03	-4.49	-	70.92	-	82.46	-2.25	-
IADA [212]	95.14	-1.04	85.39	-2.25	-	87.83	-	89.45	-1.65	-
CUA [11]	95.13	-1.04	84.83	+1.12	-	80.71	-	86.89	+0.04	-
EWC [101]	92.36	-3.13	84.83	-1.12	-	76.56	-	84.58	-2.13	-
LwF [123]	95.84	-1.05	85.95	-1.13	-	82.49	-	88.09	-0.55	-
FRIDA [170]	97.67	-1.03	99.07	-1.87	-	88.42	-	95.05	-1.45	-
SHOT-Cont. [125]	92.16	0.0	94.62	+2.15	88.17	91.01	87.57	92.60	+1.08	87.87
ConDA	92.81	+2.94	96.78	+2.15	87.1	92.98	86.99	94.18	+2.54	87.05

Table 4.5: Multi domain adaptation results for Office-Home dataset. The benchmark results are obtained from [170]. The continual experiments are done with an incoming batch size of 128 samples and a buffer size of 260 samples per domain (four samples per class per domain).

Method	$\mathcal{D}_{t_1}(P)$		$\mathcal{D}_{t_2}(C)$			$\mathcal{D}_{t_3}(A)$		$\mathcal{D}_T(Average)$		
	AC	FG	AC	FG	FW	AC	FW	AC	FG	FW
DANN [49]	73.42	-5.70	45.30	+0.23	-	45.40	-	54.71	-2.73	-
IADA [212]	75.60	+1.31	46.18	0.00	-	59.26	-	60.35	+0.65	-
CUA [11]	76.10	-1.35	47.45	+1.50	-	55.42	-	59.66	+0.07	-
EWC [101]	73.22	-6.25	46.10	+1.37	-	47.33	-	55.55	-2.69	-
LwF [123]	72.20	-5.33	44.47	+1.45	-	50.48	-	55.72	-1.94	-
FRIDA [170]	77.40	-0.41	64.31	+2.06	-	67.76	-	69.82	+0.83	-
SHOT-Cont. [125]	78.05	-0.98	52.79	-0.99	49.01	72.43	65.77	67.75	-0.98	57.39
ConDA	80.63	+0.11	55.65	-0.46	49.01	73.25	67.62	69.84	-0.18	58.32

Table 4.6: Ablation study of ConDA on the effects of using a buffer and $\mathcal{L}_{eqdiv}$ using the Office-31 dataset. The ablation study for ConDA had a continual batch size of 32 and a buffer size of 124 (four samples per class).

Method	Buffer	$\mathcal{L}_{eqdiv}$	Target	A→D	A→W	D→A	D→W	W→A	W→D	Mean
SHOT [125]			Cont.	84.74±0.00	85.32±0.07	69.77±0.18	97.86±0.00	65.50±0.16	99.20±0.00	83.73±0.05
ConDA		✓	Cont.	84.54±0.00	85.28±0.00	70.46±0.08	97.99±0.00	65.60±0.03	99.20±0.00	83.85±0.01
ConDA	✓		Cont.	83.73±0.00	87.26±0.06	71.52±0.12	97.86±0.10	70.16±0.47	99.67±0.09	85.03±0.03
ConDA	✓	✓	Cont.	84.74±0.00	88.68±0.58	72.75±0.93	98.20±0.39	70.04±0.92	99.80±0.00	85.70±0.09

target DA. IADA [212] and CUA [11] are continual domain adaptation methods specifically developed for continually changing target domains. Both of these methods proposed to utilize replay strategies to aid domain adaptation and mitigate catastrophic forgetting. They utilize DANN as the base DA method. Elastic Weight Consolidation (EWC) [101] and Learning without Forgetting (LwF) [123] are two popular methods for supervised CL. These techniques are implemented alongside DANN for benchmarking the continual multi-domain experiments. FRIDA [170] is a continual domain adaptation method specially developed to tackle incremental domain adaptation on continually varying target domains. SHOT [125] is our baseline method and we provide the SHOT results on continual settings per our proposed paradigm.

The results for the Office-Caltech dataset are shown in Table 4.4, and those for Office-Home are shown in Table 4.5. The computed Accuracy (ACC), Forgetting (FG), and Forward transfer (FW) metrices are provided in both Tables 4.4 and 4.5. Our method outperforms existing standard SOTA DA methods, beating IADA and CUA by large margins in ACC: 4,73% and 7.29%, respectively in the Office-Caltech dataset, and 9.49% and 10.18%, respectively in the Office-Home dataset. ConDA in the multi-domain also outperforms CL methods, beating EWC and LwF by 9.6% and 6.09%, respectively in the Office-Caltech dataset, and by 14.29% and 14.12%, respectively in the Office-Home dataset. This demonstrates the CL capability of our method. ConDA also comes on top of the baseline SHOT method in the continual setting by 1.58% in the Office-Caltech dataset, and by 2.09% in the Office-Home dataset. ConDA is on par with the multi-target DA method FRIDA [170] in the Office-Home dataset, and performs slightly worse than FRIDA in the Office-Caltech dataset, in terms of AC scores.

In terms of the FG metric, while ConDA has a small negative score on Office-Home, it has a positive score on the Office-Caltech dataset. To explain this, we have to be mindful of the limited room for ConDA to achieve stability on smaller datasets. Unlike FRIDA which adapts over multiple epochs, ConDA adapts over only a single epoch of the target data. The positive FG score for ConDA on the smaller Office-Caltech dataset is evidence that model stabilty may not be achieved fully after adaptation to new domains if the number of samples is low. But the presence of a multi-domain buffer and sample replay from earlier domains during adaptation to subsequent domains not only prevents forgetting, but makes the model more stable across all target domains. The issue of model stability is resolved for the larger Office-Home dataset. The small negative FG score is expected. The larger target domains enable stable adaptation, and consequently selective sample replay prevents drastic forgetting as newer domains are adapted. Since ConDA is designed and expected to operate continually on large datasets, stability should not be an issue in real-life applications.

We only have scores for ConDA and baseline SHOT in the continual setting for the FW metric, since results are not available for other methods. ConDA consistently outperforms SHOT on both datasets, exhibiting ConDA's better generalization capability on unseen domains. To visualize the effect of adaptation with

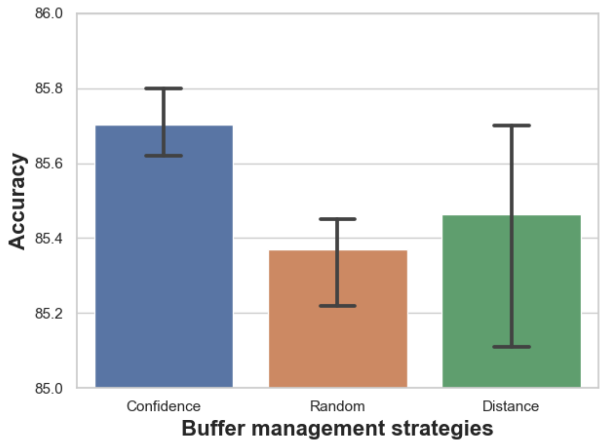


Figure 4.4: Performance with various buffer management strategies on Office-31 dataset. The experiments are performed with a batch size of 32 and buffer size of 124.

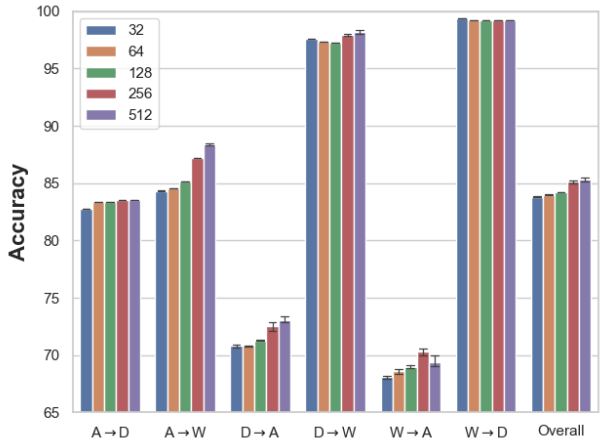
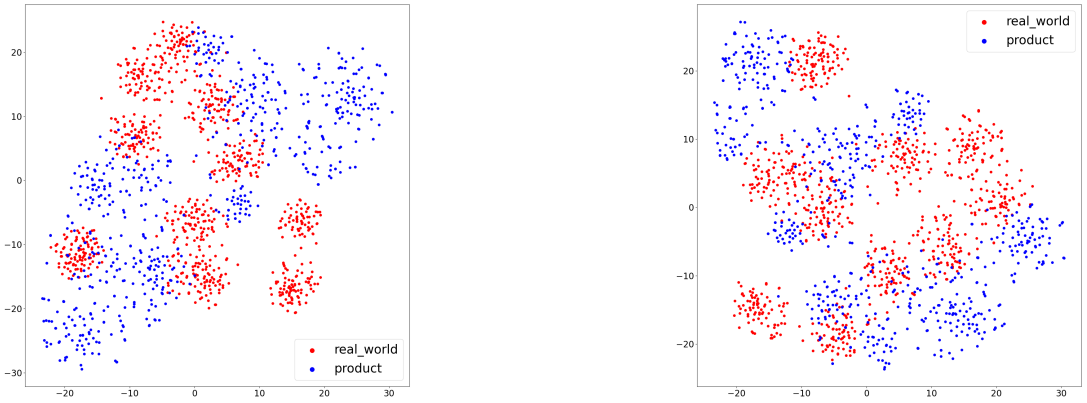


Figure 4.5: Ablation studies on Office-31 dataset with varying buffer sizes for batch size of 128.

ConDA on the latent feature space, we present t-SNE plots [67] in Figure 4.6 for continual adaptation from Real-World to Product in the Office-Home dataset. We only show the first 10 classes for clearer visualization and to avoid crowding in the plot.

The performance variation with various buffer management strategies is shown in Fig. 4.4. As mentioned in Sec. 4.3.1, the sample selection strategy based on softmax confidence provides slightly better performance over the sample selection mechanisms based on the distance to the cluster center (closer) and random selection with uniform probability.

We perform ablation studies shown in Table 4.6 to demonstrate the impact of various components of our



(a) Before adaptation

(b) After adaptation

Figure 4.6: Feature visualization plots for 10 classes before and after continual adaptation from Real-World (Source) to Product (Target) from the Office-Home dataset. (a) t-SNE plot for source-trained model on Real-World before adaptation, and (b) t-SNE plot for the target-adapted model.

model on the Office-31 dataset. These studies are performed on continual adaptation to single target domains. We note that, in the absence of a memory buffer or other continual DA modification, the performance of SHOT drops significantly during continual adaptation (in batch mode). With the addition of the buffer, performance improves by 1.3% over continual SHOT. The addition of equal diversity loss without the buffer marginally improves performance compared to continual SHOT. However, ConDA with the equal diversity loss and buffer, outperforms continual SHOT by 1.97%, which demonstrates the effectiveness of our proposed continual adaptation method.

We perform additional experiments on Office-31 (single domain), as shown in Fig. 4.5, to understand the impact of varying the buffer size during continual adaptation. To study the impact of buffer size, we fix the continual batch size at 128 samples and consider four different buffer sizes; 32, 64, 128, 256, and 520 samples. Our findings indicate that increasing the buffer length improves performance.

4.4 UCL-GV

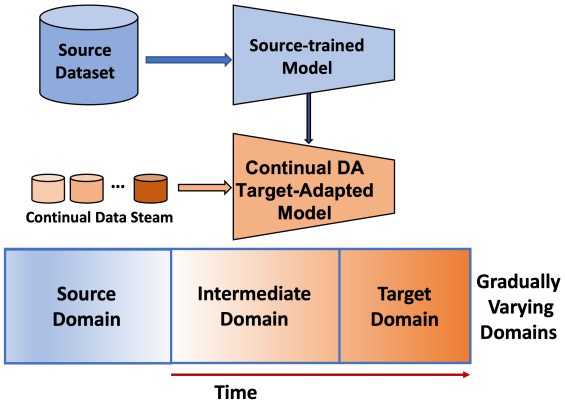


Figure 4.7: Proposed paradigm of Unsupervised Continual Learning for Gradually Varying domain adaptation (UCL-GV). The network is trained on a source domain and continually adapts using small incoming batches of data from a gradually varying target domain that has no labels.

4.4.1 Method

For the UDA problem, we consider three domains as illustrated in Fig. 4.7: a source domain, an intermediate domain, and a target domain. The source domain, $\mathcal{D}_s$, has $\mathcal{C}_s$ classes with source data $\{x_s^i, y_s^i\}_{i=1}^{n_s}$ with n_s labeled samples, where $x_s \in \mathcal{X}_s$ with labels $y_s \in \mathcal{Y}_s$. As in [105], we further consider an unlabeled intermediate domain, D_{int}, that has $\mathcal{C}_{int}$ classes with $\mathcal{X}_{int}$ samples, and an unlabeled target domain, $\mathcal{D}_{tar}$, that has $\mathcal{C}_{tar}$ classes with samples $\mathcal{X}_{tar}$. By generalizing the notations, we combine the intermediate and target domain as $\mathcal{D}_t$ with unlabeled data $\mathcal{X}_t = \mathcal{X}_{int} \bigcup \mathcal{X}_{tar}$ with $\mathcal{C}_t = \mathcal{C}_{int} = \mathcal{C}_{tar} = \mathcal{C}_s$ classes. Here $x_t \in \mathcal{X}_t$ and $\{x_t^i\}_{i=1}^{n_t}$ with n_t is the total number of unlabeled samples and t is gradually varying, $t \in [0, 1]$. We further consider that $\mathcal{D}_t$ is split into m sequential batches $\mathcal{X}_t = \{\mathcal{X}_{t_1}, \mathcal{X}_{t_2}, \mathcal{X}_{t_3}, \cdots, \mathcal{X}_{t_m}\}$ where $t_1 < t_2 < t_3 < \cdots < t_m$ and each batch has n_{t_i} i.i.d. samples where $n_t = \sum_{i=1}^{m} n_{t_i}$. Since we consider a gradual domain adaptation, we assume that the domain change in continual batches is small, i.e., $\lim_{\Delta t \to 0} d(\mathcal{D}_t, \mathcal{D}_{t+\Delta t}) = 0$ for any domain distribution distance measurement method d [131].

The objective of UCL-GV is to train a model $f_s : \mathcal{X}_s \to \mathcal{Y}_s$, parameterized by θ_s, and continually adapt it on $\mathcal{D}_t$ so that the model $f_t : \mathcal{X}_{t_i} \to \mathcal{Y}_{t_i}$, parameterized by θ_t, provides better performance on $\mathcal{X}_{tar}$ when $i = m$, compared to $f_t : \mathcal{X}_t \to \mathcal{Y}_t$ with having only f_s and $\mathcal{X}_t$ during adaptation. The overall objective can also be represented in terms of the loss computation as follows [131].

$$\min_{\theta_t} \mathbb{E}_{t\sim U(0,1)} \mathbb{E}_{(x_t,y_t)\sim \mathcal{D}_{tar}} \mathcal{L}(f_t(x_t), y_t) = \min_{\theta_t} \int_0^1 \mathbb{E}_{(x_t,y_t)\sim \mathcal{D}_{tar}} \mathcal{L}(f_t(x_t), y_t)\, dt \tag{4.11}$$

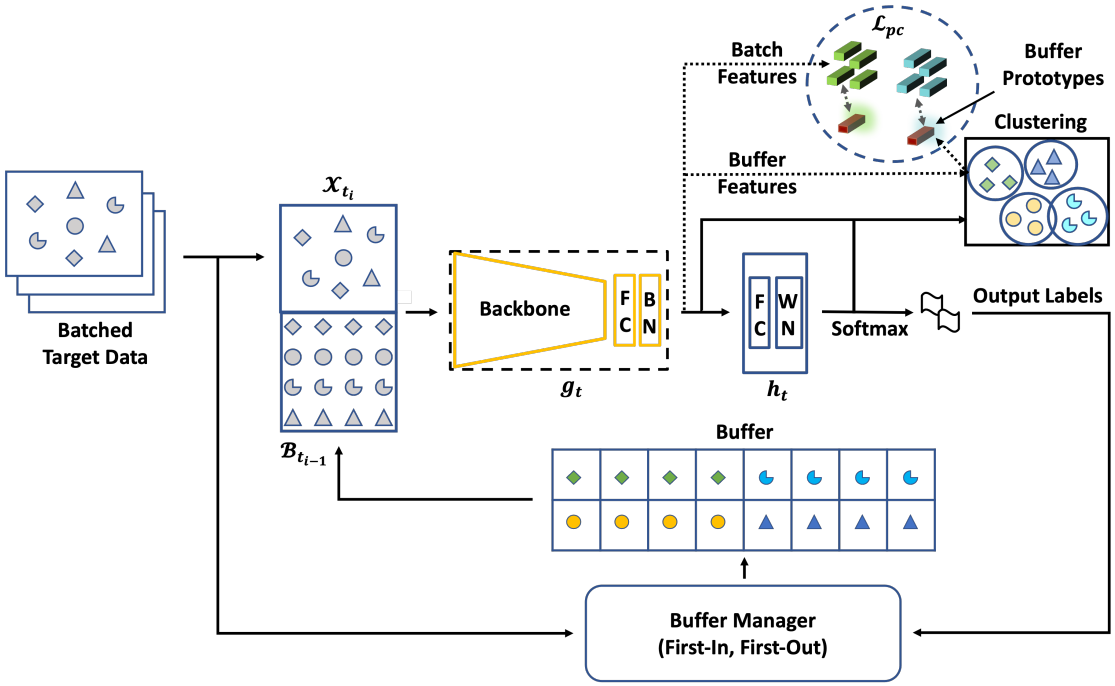


Figure 4.8: Proposed UCL-GV method for unsupervised continual learning for domain adaptation in gradually varying domains.

The architecture of UCL-GV is shown in Fig. 4.8. Inspired by [125], we initially train our source model $f_s(x) = h_s(g_s(x))$ on the source data. The model consists of two parts, a feature extractor with a backbone followed by a fully connected layer and a batch normalization layer denoted as g_s. The generated features are passed through the hypothesis layer that consists of a fully convolutional layer, followed by a weight normalization layer denoted as h_s. The source network is trained with a label smoothing loss. For the target model, $f_t(x) = h_t(g_t(x))$, the feature extractor model g_t is initialized with g_s and set as trainable, while the transferred hypothesis model $h_t = h_s$ is kept frozen throughout the adaptation procedure.

The unlabeled data from $\mathcal{D}_t$ are sequentially presented to the network and certain samples are selectively stored in a buffer after processing each incoming new batch, $\mathcal{X}_{t_i}$. At each step in time when a new batch is received, the existing buffer samples are added to the incoming batch samples for adaptation. This prevents the clusters from deviating too much from one batch to the next. The details of the buffer and buffer man-

agement strategies are provided next, in Sec. 4.4.1 and 4.4.1. Since the incoming samples are without labels, clustering is needed for pseudo-label assignment. However, the clustering techniques utilized in [14, 125] primarily deal with samples from a stationary distribution and are not suitable for gradually varying domains. In this paper, we improve upon this clustering technique to incorporate samples from non-stationary distributions. Since the domain gap between the incoming batch samples and the buffer samples is small, we utilize contrastive alignment between the buffer prototypes (cluster centers) and the batch samples by minimizing the prototypical contrastive loss $\mathcal{L}_{pc}$, as shown in Fig. 4.8. The procedure is detailed in Sec. 4.4.1. It is important to note that the existing buffer samples and new incoming batch samples are fed through the network only once, i.e. only one epoch of the $\mathcal{B}_{t_{i-1}} \bigcup \mathcal{X}_{t_i}$ samples is allowed at each time step during adaptation. The total number of adaptation time steps is equal to the number of sequential incoming batches of data from $\mathcal{D}_t$, the combined intermediate and target domain.

Buffer

In our setting we consider closed-set domain adaptation where $\mathcal{C}_s = \mathcal{C}_t$ with the same classes in the source and target domains. We allocate equal number of samples from each class in the buffer $\mathcal{B}_t = \{\mathcal{B}_{t_1}, \mathcal{B}_{t_2}, \cdots, \mathcal{B}_{t_m}\}$ based on pseudo-label assignment on incoming target samples. This allows the class-wise data distribution to be considerably uniform throughout the adaptation process. The buffer stores raw samples for adaptation, and the buffer samples are managed by a buffer manager as described in the next subsection.

Buffer Manager

The buffer manager is responsible for populating the buffer with new samples while partially or fully dropping the existing samples depending on the number of batch and buffer sizes. We considered multiple buffer sample selection mechanisms that exist for the supervised CL paradigm. One popular scheme of sample selection is uniform random, where all the incoming batch samples are combined with the existing buffer samples and the samples to be stored for the next time step are randomly selected with uniform probability [15]. Minimum logit distance is another method where the samples are selected based on the distance to a decision boundary [16]. Some other mechanisms are also introduced in [59] such as choosing samples with minimum confidence, maximum loss, maximum time since last replay, and so on. However, we argue that most of the supervised buffer management strategies are not readily applicable to unsupervised continual learning, except the random selection technique. We tested several schemes for updating the buffer samples, such as selecting samples randomly with uniform probability, samples with high confidence, samples closer to the cluster center, and samples with first-in, first-out queue. We found that *first-in, first-out queue*

performs slightly better than all of the other methods for gradually varying domain adaptation. Intuitively, since the domain is gradually evolving, the estimated pseudo labels are the most appropriate when the domain shift within the available data is minimum. If the domain shift between existing buffer samples and the incoming batch samples is high, the estimated pseudo label quality degrades and hence the adaptation performance also degrades.

Clustering

At time t_i, the network utilizes a new batch of samples $\mathcal{X}_{t_i}$ and the existing buffer samples $\mathcal{B}_{t_{i-1}}$ from the previous time step. The combined data $\mathcal{X}_{t_i} \bigcup \mathcal{B}_{t_{i-1}}$ produces n_b i.i.d. minibatches that are passed through the feature extraction network g_t, and the features are accumulated to perform clustering. We adopted weighted k-means clustering encouraged from [14, 125, 198] that provides the pseudo labels and cluster centers.

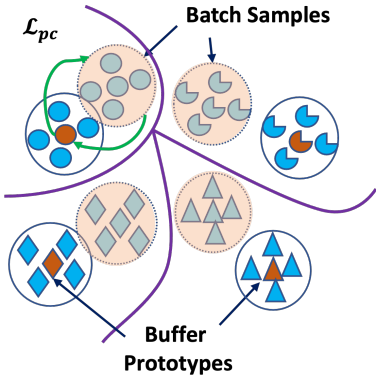


Figure 4.9: Application of contrastive loss using the buffer prototypes (cluster centers) and the batch samples, for better clustering.

Contrastive Alignment

Since the domain gap between two consecutive data batches is small (due to the gradually varying domains), we propose to align the feature representations of the incoming batch and buffer samples using a contrastive loss. Such alignment between the buffer and batch features complements the clustering process and generates better pseudo labels. We compute a cosine distance based contrastive loss from the buffer prototypes to the batch samples, as shown in Fig. 4.9. The buffer prototypes (cluster centers) are computed with the current state of the feature extractor $\hat{g}_t$, using the pseudo-labels $\hat{y}_t \in \hat{\mathcal{Y}}_t$ for the samples in the buffer and the

incoming batch samples, $\mathcal{B}_{t_{i-1}} \bigcup \mathcal{X}_{t_i}$, as follows [125].

$$\mathbf{z}_k = \frac{\sum_{x_t \in \mathcal{B}_{t_{i-1}}} \mathbb{1}(\hat{y}_t = k)\hat{g}_t(x_t)}{\sum_{x_t \in \mathcal{B}_{t_{i-1}}} \mathbb{1}(\hat{y}_t = k)} \tag{4.12}$$

In our experiments, $\mathbf{z}_k \in \mathbb{R}^{|\mathcal{C}_t| \times 256}$. The batch features are computed as follows.

$$\mathbf{z} = \hat{g}_t(x_t), \forall x_t \in \mathcal{X}_{t_i} \tag{4.13}$$

Both the batch features and the buffer features are normalized.

$$\hat{\mathbf{z}}_k = \frac{\mathbf{z}_k}{||\mathbf{z}_k||}, \hat{\mathbf{z}} = \frac{\mathbf{z}}{||\mathbf{z}||} \tag{4.14}$$

The normalized features are used to compute the prototypical contrastive (PC) loss $\mathcal{L}_{pc}$ [116, 117].

$$\mathcal{L}_{pc} = -\log \frac{\exp\left(\hat{\mathbf{z}}_i \cdot \hat{\mathbf{z}}_{k=\hat{y}_t^i}\right)}{\sum_{c=1}^{|\mathcal{C}_t|} \exp\left(\hat{\mathbf{z}}_i \cdot \hat{\mathbf{z}}_{k=c}\right)} \tag{4.15}$$

We minimize the PC loss in conjunction with the other loss functions.

Overall Loss Function

We adopt the Information Maximization (IM) [103, 125] loss, according to the formulation in ConDA that minimizes the entropy $\mathcal{L}_{ent}$ and equal diversity loss $\mathcal{L}_{eq}$. With the pseudo labels computed in the overall clustering, we compute the cross-entropy loss below.

$$\mathcal{L}_{ce} = \mathbb{E}_{x_t \in \mathcal{B}_{t_{i-1}} \bigcup \mathcal{X}_{t_i}, \hat{y}_t \in \hat{\mathcal{Y}}_t} - \log \sigma_k(f_t(x_t)) \tag{4.16}$$

where, σ_k is the softmax function. The overall loss function is written as follows.

$$\mathcal{L}(g_t) = \mathcal{L}_{ent} + \gamma_1 \mathcal{L}_{eqdiv} + \gamma_2 \mathcal{L}_{ce} + \gamma_3 \mathcal{L}_{pc} \tag{4.17}$$

where γ_1, γ_2, and γ_3 are hyper-parameters. The overall process is presented in Algorithm 2.

Algorithm 2: UCL-GV algorithm

Input : A source trained model $f_s = h_s \cdot g_s : \mathcal{X}_s \rightarrow \mathcal{Y}_s$, evolving data batches $\{\mathcal{X}_{t_1}, \mathcal{X}_{t_2}, \cdots \mathcal{X}_{t_m}\}$ from $\mathcal{D}_t$.

Output : A model continually adapted on $\mathcal{D}_t$ and the corresponding predicted labels for $\mathcal{X}_{tar}$.

Init. : Initialize the target network g_t with g_s and set the hypothesis network $h_t = h_s$ and keep it frozen during adaptation.

for $i \leftarrow 1$ **to** m **do**
  if $i = 1$ **then**
    $X \leftarrow \mathcal{X}_{t_i}$;
  else
    $X \leftarrow \mathcal{X}_{t_i} \bigcup \mathcal{B}_{t_{i-1}}$;
  end
  $\hat{Y} \leftarrow$ Compute psuedo labels for X;
  for $j \leftarrow 1$ **to** n_b **do**
    Get i.i.d batch samples from $(X, \hat{Y})$;
    Compute $\mathcal{L}_{ent}$, $\mathcal{L}_{eq}$, and $\mathcal{L}_{ce}$;
    if $i = 1$ **then**
      $\mathcal{L}_{pc} \leftarrow 0$;
    else
      $\mathcal{L}_{pc} \leftarrow$ Compute the PC loss using Equation (4.15);
    end
    Compute $\mathcal{L}(g_t)$ using Equation (4.17);
    Optimize g_t with $\mathcal{L}(g_t)$;
  end
  $\mathcal{B}_{t_i} \leftarrow$ Fill buffer using $g_t$and $(\mathcal{X}_{t_i}, \mathcal{B}_{t_{i-1}})$;
end

4.4.2 Datasets and experiments

We used two datasets, rotating MNIST and CORe50, for evaluation. We adopt the **rotating MNIST** [105] which has 50,000 training and 10,000 test images. It is created to mimic an evolving domain where the first 20,000 images are used for training our source model and are rotated between $[0°, 10°]$. The next 30,000 images from the training set form the intermediate domain and are rotated between $[10°, 50°]$. The 10,000 test images are selected as the target domain and are rotated between $[50°, 60°]$. Following [125], we consider the entire target domain for evaluation after adaptation on the intermediate and the target domains. Examples of the rotating MNIST dataset are shown in Fig. 4.10.

Further, we restructure the **CORe50** [137] dataset to evaluate UCL-GV under the continually evolving domain adaptation setting. CORe50 dataset is specifically designed for CL research and has 50 domestic objects from 10 categories collected on 11 sessions. We found that choosing eight sessions makes the dataset suitable for gradually varying domains where the backgrounds of the images vary gradually in appearance. Additionally, there are pose and illumination changes among various sessions. We used the samples from session 's1' as the source domain, 's2', 's3', and 's8' as the unlabeled intermediate domain, and 's9', 's11', 's4', and 's10' as the target domain where the samples are appended according to the order mentioned here. Examples of the CORe50 dataset are shown in Fig. 4.11.

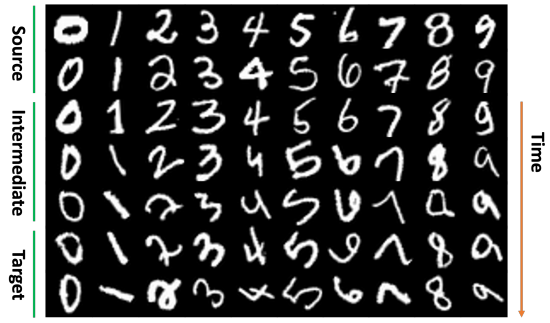


Figure 4.10: Rotating MNIST dataset.

The source model is trained with randomly sampled data from the entire source domain. Following the setting in [105], the intermediate domain is chosen to implement a gradual change, rather than a drastic change from the source domain to the target domain. The intermediate domain and the target domain are provided to the network sequentially, however, the classes are randomly mixed. For the rotating MNIST dataset, we utilize a LeNet backbone [113] with two convolutional layers. For the CORe50 dataset, we choose a ResNet18 backbone [63]. We normalize the rotating MNIST samples to have 0.5 mean and 0.5 standard deviation. CORe50 samples undergo resizing to 256×256 pixels, and random cropping to size 224×224, random horizontal flipping, and normalization for adaptation. The starting learning rate for

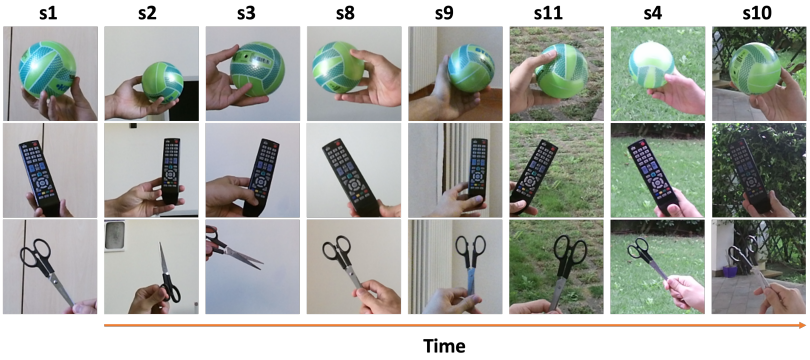


Figure 4.11: CORe50 [137] dataset in a gradual time varying setting.

Table 4.7: Percent accuracy of UCL-GV and comparison with other methods. The experiments on rotating MNIST are performed with a continual batch size of 128 and buffer size of 512. CORe50 experiments are performed with a continual batch size of 16 and buffer size of 32. All evaluations are conducted on the target domain $\mathcal{D}_{tar}$.

Method	Adaptation domain	Domain availability	Rotating MNIST	CORe50
Baseline [125]	None (No adaptation)	Full	45.16	74.59
Baseline [125]	Target only	Full	67.88	90.19
Baseline [125]	Intermediate + Target	Full	96.20	91.49
Gradual ST [105]	Intermediate + Target	Continual	92.03	N/A
Baseline [125]	Intermediate + Target	Continual	94.20	87.14
UCL-GV	Intermediate + Target	Continual	95.66	89.07

rotating MNIST is 0.01 and for CORe50 is 0.001, and are varied according to the setup of [125].

4.4.3 Results

Performance on Full Target Domain

We computed the domain adaptation performance with our baseline method [125] using the full target dataset, as shown in Table 4.7. For all settings, the model is evaluated only on the target dataset, $\mathcal{X}_{tar}$. The model with only source training (without adaptation on the intermediate or the target domain) evaluated on the target domain indicates the domain gap between the source and the target domain. On the rotating MNIST dataset, the low classification score of 45.16% of the source trained model indicates a large domain gap between the source domain and the target domain. On the other hand, the performance of the source trained model on CORe50 dataset is 74.59%, which shows a smaller domain gap between the source and the target domains. The CORe50 dataset contains slight changes among the three domains in the background.

The target-only model is the case where the model is trained on the source dataset and adapted to the target dataset, $\mathcal{X}_{tar}$ without any intermediate domain data. After adapting to the target domain with the baseline method, performance on both datasets improves significantly. For the rotating MNIST dataset, the performance improves by 22.72% and for the CORe50 dataset, the performance improves by 15.6%.

With the availability of the intermediate domain, the shift between the source and the adaptation domains is much smaller. This leads to significant performance gains compared to the target-only adapted baseline model, even for the cases of continual learning from small incoming batches.

Performance on Gradually Varying Domains

UCL-GV shows significant improvement over the existing baseline [125] and Gradual ST [105], as shown in Table 4.7. The results on Gradual ST [105] are obtained by running the publicly available codebase on our dataset settings. In the continual adaptation setting, the performance of the baseline [125] method degrades by 2% on the rotating MNIST dataset and by 4.35% on the CORe50 dataset, compared to the adaptation on the full intermediate and target domains simultaneously. UCL-GV outperforms Gradual ST by 3.63% and the baseline method by 1.46% on rotating MNIST dataset on the continual settings. On the CORe50 dataset, UCL-GV outperforms the continual baseline method by 1.93%.

To further illustrate the continual learning capability of our method, we evaluate the classification perfor-

mance on all of the target samples $\mathcal{X}_{tar}$ of the rotating MNIST dataset after each incoming batch $\mathcal{X}_{t_i}$ from $\mathcal{D}_t$, as shown in Fig. 4.12. Our method shows consistent performance gains while learning on new batches of data.

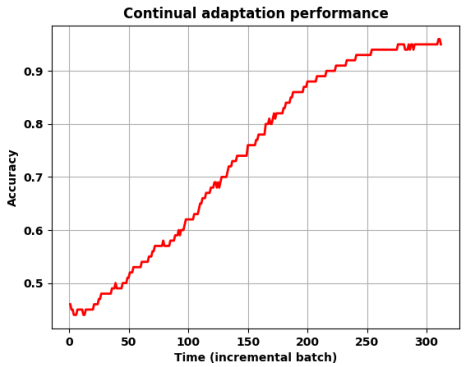


Figure 4.12: Performance of UCL-GV on the rotating MNIST target domain $\mathcal{D}_{tar}$ during continual adaptation on each incremental batch from the combined intermediate and target domain $\mathcal{D}_t$.

Effects of Batch and Buffer Sizes

To understand the impact of batch and buffer sizes on continual adaptation, we conducted ablation studies on the rotating MNIST dataset. Fig. 4.13 shows the results obtained when varying the buffer size (left) and batch size (right). The results in Fig. 4.13 (left) also demonstrate the effectiveness of the first-in, first-out queue. Additionally, we observe that the performance increases with increase in the buffer size. This observation is consistent with the existing supervised streaming learning scenario [59]. Based on intuition,

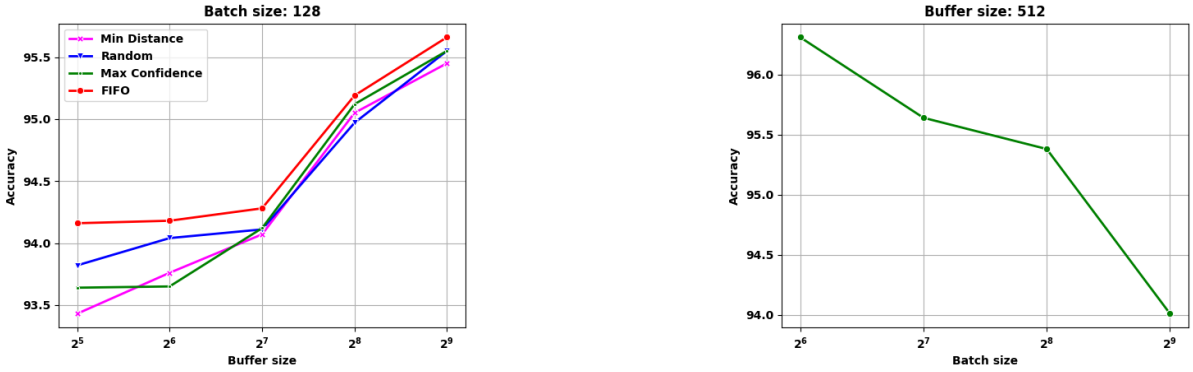


Figure 4.13: Impact of varying the buffer size (left) and batch size (right) of UCL-GV on the rotating MNIST dataset.

increasing the buffer size provides access to more samples, which improves unsupervised clustering and prototype representation from the buffer samples. However, this comes at the cost of larger memory footprint, and the buffer size selection will depend on the available system resources.

The results for various batch sizes, while the buffer size is kept fixed, are shown in Fig. 4.13 (right). When the incoming batch size is varied from 64 to 512, the performance degrades with increase in batch size, which may appear counter intuitive. However, since the target domain data is varying gradually, that is, the class-wise data distribution is continuously changing, having a larger batch size might cause overlap between different class distributions across the varying domain. This can potentially lead to incorrect pseudo-label assignments and eventually result in negative adaptation and lower performance.

Ablation Studies

We demonstrate the effectiveness of various aspects of UCL-GV by performing ablation studies on the rotating MNIST dataset. We performed each experiment three times and report the average in Table 4.8. The UDA baseline [125] method achieves 94.20% accuracy in continual adaptation across varying domains. After adding the buffer, we observe $\sim$1% improvement in performance, which corresponds to 14.8% reduction in error, validating the effectiveness of including the memory buffer. With the introduction of contrastive alignment between the buffer prototypes and the batch samples, the final performance of UCL-GV is 95.66%, which is a 1.46% total improvement over the baseline, or 25.2% reduction in error .

Table 4.8: Ablation studies of UCL-GV on the rotating MNIST dataset. Experiments are performed with a continual batch size of 128 and buffer size of 512.

Method	Percent Accuracy
Baseline	94.20
Baseline+Buffer	95.06
UCL-GV: Baseline+Buffer+$\mathcal{L}_{pc}$	95.66

4.5 Continual domain adaptation on aerial images under gradually degrading weather

4.5.1 Benchmark Datasets

To the best of our knowledge, no existing dataset meets our criteria for evaluating continual domain adaptation under gradually changing weather conditions. We therefore utilize two existing aerial datasets AID [213] and UCM [223] to generate gradually varying weather conditions using the *imgaug* Python library [92]. We use all 30 classes for AID, and all 21 classes for UCM. We use two augmenters *CloudLayer* and *SnowflakesLayer* from *imgaug.augmenters.weather* library to synthesize cloudy, and snowfall weather conditions on real AID, and UCM images. With two augmentations on AID, and two augmentations on UCM, we get a total of four datasets with gradually degrading weather conditions. We call the new AID dataset with cloud cover distortion, and with snowfall distortion AID-CC and AID-SF, respectively. Similarly, we name the new UCM dataset with cloud cover distortion, and with snowfall distortion UCM-CC and UCM-SF, respectively.

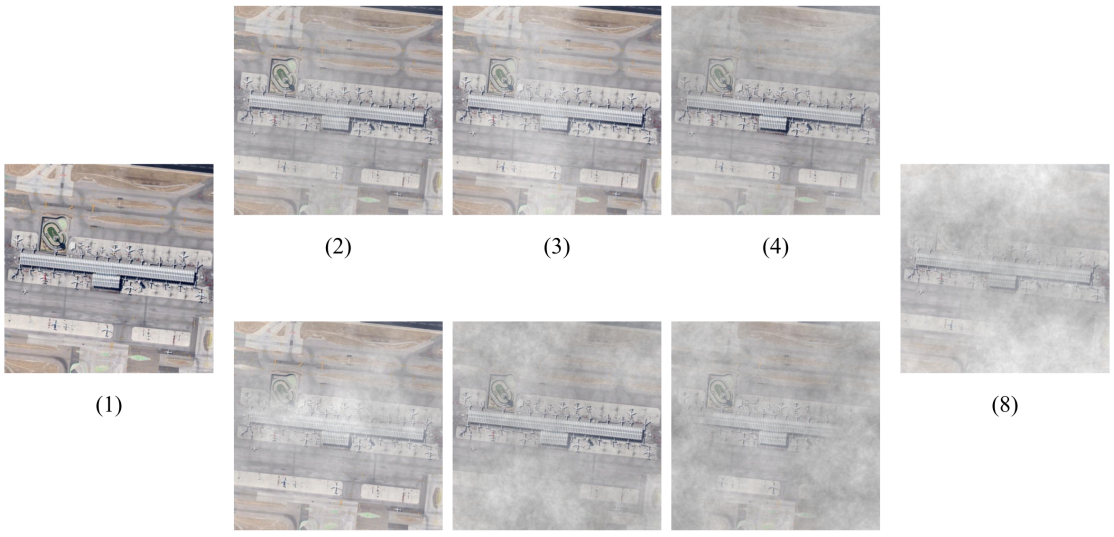


Figure 4.14: AID-CC dataset with cloud cover degradation. (1) is the source domain, (8) is the target domain, and (2-7) are progressively degrading intermediate domains.

We take the clear weather images from AID, and UCM as the source data for each respective dataset. The

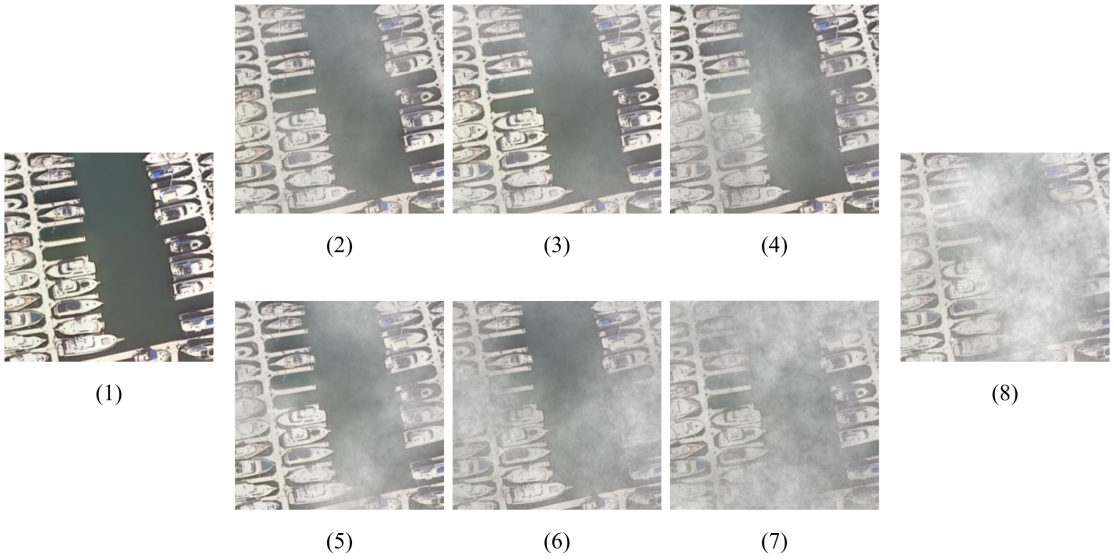


Figure 4.15: UCM-CC dataset with cloud cover degradation. (1) is the source domain, (8) is the target domain, and (2-7) are progressively degrading intermediate domains.

seven levels of cloud cover degradation are made by varying the density and size of clouds. The five levels of snowfall degradation are made by varying the density of the snowflakes, and overall brightness of the scene. The data with the highest level of degradation for both types are taken as the respective target domain data, while the rest are treated as gradually varying intermediate domains, depending on the intensity of weather degradation. We therefore have six intermediate domains for cloud cover, and four intermediate domains for snowfall.

We present a few examples of our newly created synthetic datasets in Figures 4.14, 4.15, 4.16, and 4.17. Figures 4.14 and 4.15 show examples of the seven stages of gradually worsening cloud coverage on AID and UCM, respectively. Figures 4.16 and 4.17 show examples of the five gradually degrading snowfall conditions on AID and UCM, respectively.

4.5.2 Implementation details

All three models we evaluated consist of a backbone or feature extractor, followed by a bottleneck layer, and finally a classifier layer. The buffer sizes for ConDA and UCL-GV are fixed at 420 samples, while Continual SHOT does not contain a buffer. The models are adapted with an SGD optimizer with momentum of 0.9.

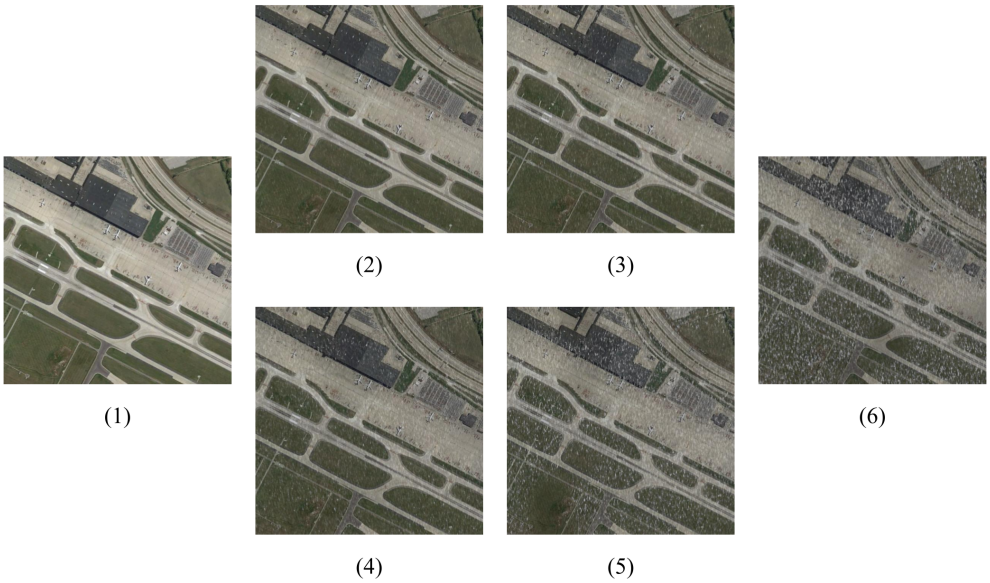


Figure 4.16: AID-SF dataset with snowfall degradation. (1) is the source domain, (6) is the target domain, and (2-5) are progressively degrading intermediate domains.

The initial learning rates $\eta_0 = 0.02$ and $\eta_0 = 0.002$ are used with a learning rate scheduler such that the learning rate $\eta = \eta_0 \cdot (1 + 10 \cdot p)^{-0.75}$, where $p = \frac{i}{i_T}$ changes from 0 to 1 for each incoming batch. Here i is the current iteration, and i_T is the total number of iterations for each incoming batch of intermediate/target data.

4.5.3 Results and discussion

Table 4.9: Initial results on the gradually degrading AID and UCM datasets with ResNet-50 backbone on the final target domain. Source-trained refers to the model trained on the source data only, without any adaptation. The top accuracy is in bold and the second best is underlined.

Method	Cloud Cover		Snowfall	
	AID-CC	UCM-CC	AID-SF	UCM-SF
Source-trained	12.29	32.81	42.65	58.38
Continual-SHOT [125]	**84.14**	80.10	94.21	95.00
ConDA	<u>80.41</u>	**85.54**	<u>95.38</u>	<u>95.90</u>
UCL-GV	79.40	<u>85.19</u>	**95.49**	**95.95**

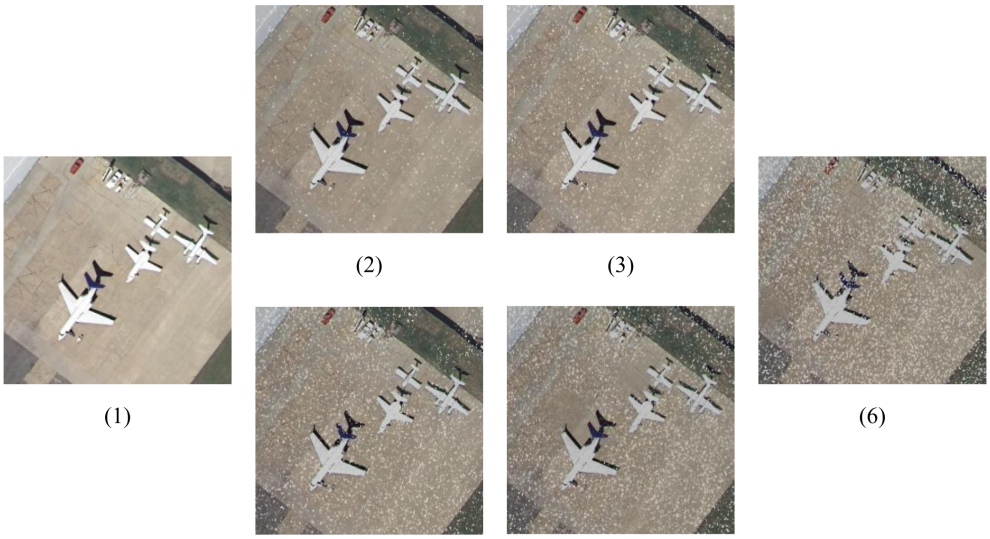
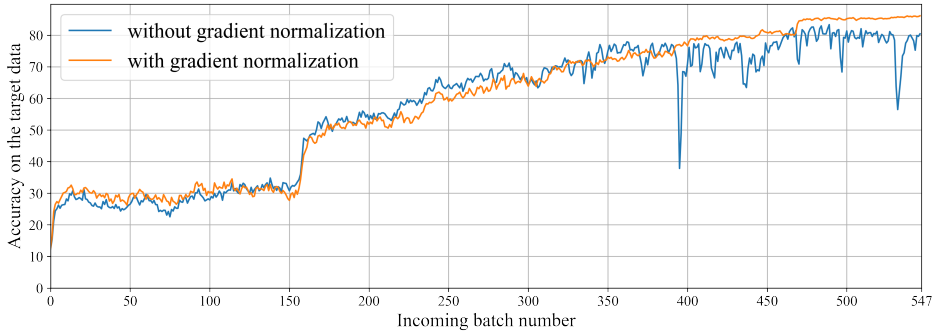


Figure 4.17: UCM-SF dataset with snowfall degradation. (1) is the source domain, (6) is the target domain, and (2-5) are progressively degrading intermediate domains.

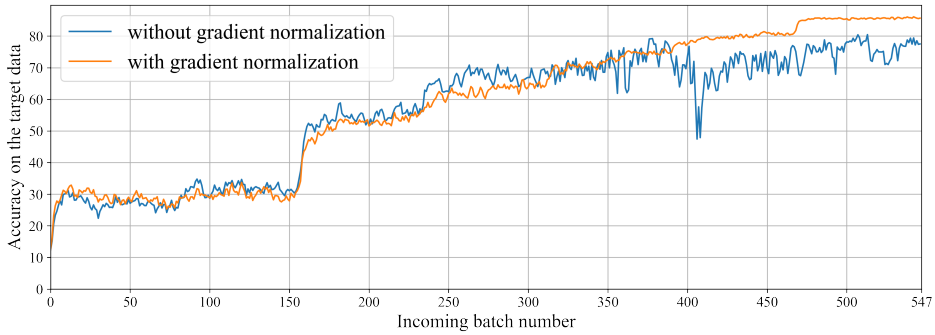
Initial results for all three methods considered using a ResNet-50 backbone are shown in Table 4.9. The two continual DA models, UCL-GV and ConDA, outperform Continual-SHOT for snowfall degradation, but the results for cloud cover are mixed. For AID-CC, we see significant drops in performance for the ConDA and UCL-GV, compared to continual-SHOT. With additional examination of the results, we found occasional lack of stability that ConDA and UCL-GV may encounter for certain batches during adaptation. The continual batches within a domain do not have any particular order in which they are received, and the performance drops can happen at any time during the adaptation process. Such adaptation instability needs to be addressed to improve performance, since continual DA does not revisit batches of images that have already been processed or seen by the model.

We propose gradient normalization to help stabilize the adaptation process for the continual models, and improve their performance. Empirically, we conduct L2-normalization of all the gradients after backpropagation through the model and before optimization for each adaptation iteration. In Figure 4.18, we plot the continual adaptation performance for each incoming batch, with and without gradient normalization, for the continual models on AID-CC.

From Fig. 4.18, we can see that ConDA and UCL-GV may face significant stability issues at times, and model performance may drop significantly (up to $\sim 40\%$) from one incoming batch to the next. Although



(a) Accuracy of ConDA with *ResNet-50* backbone on the final target data, with and without gradient normalization, as it continually adapts to the incoming batches of intermediate and target domains of AID-CC.



(b) Accuracy of UCL-GV with *ResNet-50* backbone on the final target data, with and without gradient normalization, as it continually adapts to the incoming batches of intermediate and target domains of AID-CC.

Figure 4.18: Effect of gradient normalization on adaptation stability for the continual models ConDA and UCL-GV.

the continual models start to recover in the subsequent incoming batches for AID-CC, they may not adapt optimally. In our experiments, it is evident that gradient normalization greatly mitigates the drops in performance for some of the batches, that are observed in the original forms of the models. With gradient normalization, both ConDA and UCL-GV gradually continue to better adapt to the target domain as the adaptation process progresses.

The results for all three methods with gradient normalization, and different learning rates are tabulated in Table 4.10. In all cases, ConDA and UCL-GV outperform continual SHOT, but the best results depend on the learning rate. It has to be noted that for optimal performance, continual DA necessitates the models to undergo fast optimization, as revisits to earlier data batches are not allowed, and the models need to adapt to the target domain over single passes of the continual data stream. On the smaller UCM-CC and UCM-

Table 4.10: Results on the gradually degrading AID and UCM datasets, using ResNet-50 backbone, gradient normalization, and initial learning rate of $\eta_0 = 0.002$ and $\eta_0 = 0.02$. Source-trained refers to the model trained on the source data only, without any continual target adaptation over the intermediate and target domains. The top accuracy is in bold and the second best is underlined.

Method	AID-CC		UCM-CC		AID-SF		UCM-SF	
	$\eta_0 =$ 0.002	$\eta_0 =$ 0.02	$\eta_0 =$ 0.002	$\eta_0 =$ 0.02	$\eta_0 =$ 0.002	$\eta_0 =$ 0.02	$\eta_0 =$ 0.002	$\eta_0 =$ 0.02
Source-trained	12.29	12.29	32.81	32.81	42.65	42.65	58.38	58.38
Continual-SHOT [125]	80.37	79.36	67.95	81.71	88.71	94.50	78.29	95.57
ConDA	**86.13**	58.91	79.43	<u>81.33</u>	<u>94.87</u>	93.12	90.05	**97.38**
UCL-GV	<u>85.67</u>	65.09	78.76	**82.10**	**95.05**	93.68	90.48	<u>96.71</u>

SF datasets, both ConDA and UCL-GV adapt better with a higher learning rate, due to faster optimization afforded by the higher initial learning rate. On the other hand, the models cannot adequately adapt to the target domain for UCM-CC and UCM-SF at the smaller initial learning rate of $\eta = 0.002$, due to comparatively slower optimization. Therefore, for UCM-CC and UCM-SF, best results are obtained for higher learning rates. It also has to be noted that, the continual adaptation process becomes more susceptible to instability at higher learning rates, the effects of which can be seen in the results for AID-CC at $\eta_0 = 0.02$. On the larger AID-CC and AID-SF datasets, better performance is obtained with the smaller initial learning rate of $\eta_0 = 0.002$. As AID-CC and AID-SF have large number of samples to process, continual models can reach an optimal solution with a slower optimization at smaller learning rates, while the pitfalls of higher instability at higher learning rates can be avoided.

We further evaluate the continual models with two transformer backbones: Vision Transformer (ViT) [37] and Swin-V2 [133, 135]. To keep the computational load tractable, we choose the base versions of the transformer backbones for our experiments. We report the adaptation performance of the three models, with and without gradient normalization, on the AID-CC and AID-SF datasets at the lower initial learning rate of $\eta_0 = 0.002$ in Table 4.11, and on the UCM-CC and UCM-SF datasets at the higher initial learning rate of $\eta_0 = 0.02$ in Table 4.12.

We can see from the results in Table 4.11 that while gradient normalization may prevent optimal adaptation performance for the continual-SHOT model, adaptation stability increases for the two buffer-fed continual models ConDA and UCL-GV, and both models generally adapt better to the target domains at the end of the adaptation process. Having been stabilized with gradient normalization, the models ConDA and UCL-GV can be considered comparable in adaptation performance. Accuracies of ConDA and UCL-GV on the target domain for continual adaptation for AID-CC, with Swin-B as the backbone are plotted in Figure 4.19, to inspect the increase of adaptation stability due to gradient normalization.

Table 4.11: Results on AID-CC, and AID-SF with ResNet-50, ViT-B, and Swin-B backbones, at initial learning rate of 0.002. Source-trained method refers to the model trained on the source data only, without any continual target adaptation over the intermediate and target domains. The top accuracy is in bold and the second best is underlined. respectively.

Method	Backbone (# params)	AID-CC		AID-SF	
		w/o. Grad Norm	w. Grad Norm	w/o. Grad Norm	w. Grad Norm
Source-trained	ResNet-50 (23M)	12.29	12.29	42.65	42.65
Continual-SHOT [125]		84.14	80.37	94.21	88.71
ConDA		80.41	86.13	95.38	94.87
UCL-GV		79.40	85.67	95.49	95.05
Source-trained	ViT-B (86M)	11.50	11.50	55.85	55.85
Continual-SHOT [125]		80.43	74.18	90.41	88.79
ConDA		78.28	79.94	89.65	89.77
UCL-GV		79.55	79.84	88.56	90.29
Source-trained	Swin-B (88M)	19.96	19.96	67.34	67.34
Continual-SHOT [125]		89.76	91.20	96.29	94.12
ConDA		81.82	**93.20**	95.82	<u>97.77</u>
UCL-GV		81.67	<u>92.82</u>	93.22	**97.84**

The results in Table 4.12, particularly those for UCM-CC show the dangers of instability during adaptation. At the higher initial learning rate, and without stabilization by gradient normalization, the adaptation process may completely collapse. The impact is more severe for UCM-CC due to the higher degree of degradation of cloud cover in our datasets. But when the gradients are normalized and the adaptation process is stabilized, the models show promising performance. This clearly shows the necessity and effectiveness of the improvement over no gradient normalization in the continual methods we propose in this dissertation. Similar to results for AID-CC and AID-SF, ConDA and UCL-GV with Swin-B backbone beat the other two backbone architectures.

Overall, the two continual DA models ConDA and UCL-GV beat the standard SHOT model under continual setting. This shows the efficacy of selectively storing samples in a memory buffer and replaying these samples from earlier batches mixed with the samples from new incoming batch. Such memory replay helps in retaining knowledge gained from earlier batches, and results in a better domain adaptation to the target domain. Gradient normalization with smaller learning rates, despite preventing optimal adaptation in certain cases for smaller datasets, significantly increases adaptation stability, and prevents the models from potentially collapsing.

In terms of backbones, Swin generally outperform the CNN-based ResNet-50 model for continual DA on

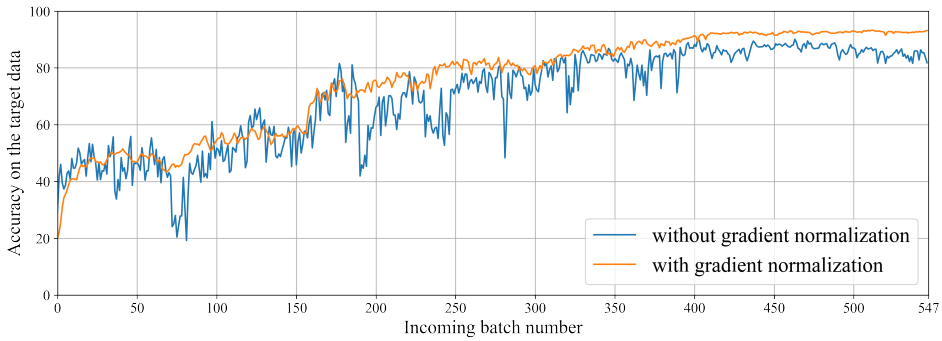
Table 4.12: Results on UCM-CC, and UCM-SF with ResNet-50, ViT-B, and Swin-B backbones, at initial learning rate of 0.02. Source-trained method refers to the model trained on the source data only, without any continual target adaptation over the intermediate and target domains. The top accuracy is in bold and the second best is underlined. respectively.

Method	Backbone (# params)	UCM-CC		UCM-SF	
		w/o. Grad Norm	w. Grad Norm	w/o. Grad Norm	w. Grad Norm
Source-trained	ResNet-50 (23M)	32.81	32.81	58.38	58.38
Continual-SHOT [125]		9.33	81.71	73.81	95.57
ConDA		6.95	81.33	96.00	**97.38**
UCL-GV		8.52	82.10	92.14	96.71
Source-trained	ViT-B (86M)	48.00	48.00	59.24	59.24
Continual-SHOT [125]		82.00	79.38	95.10	94.14
ConDA		56.29	77.81	82.71	95.71
UCL-GV		25.52	71.95	76.19	94.52
Source-trained	Swin-B (88M)	14.10	14.10	72.67	72.67
Continual-SHOT [125]		55.62	<u>86.05</u>	87.10	<u>97.05</u>
ConDA		3.76	**87.29**	69.43	96.95
UCL-GV		7.38	85.81	46.38	**97.38**

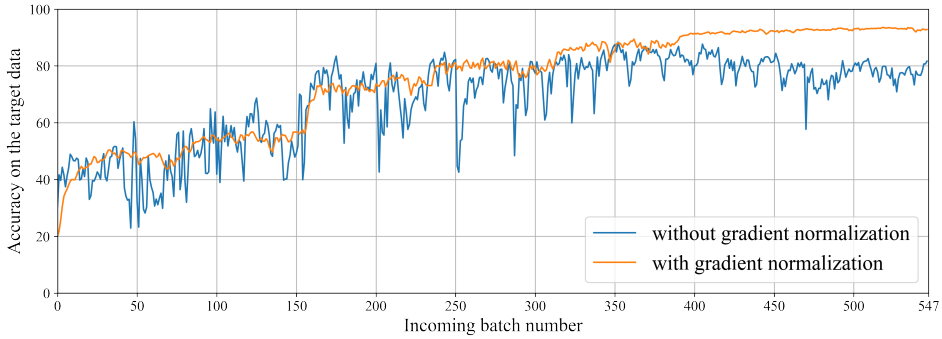
our benchmark evaluations. This can be attributed to its attention mechanism, as well as the increased ability of transformers to capture global feature representations at the lower layers, compared to CNN-based architectures. Between the two attention models evaluated, Swin consistently outperforms ViT. ResNet-50, with much lower number of parameters, beat ViT at well. This can be attributed to the weak inductive bias observed in ViT [190], leading to increased overfitting on the source data. Raghu *et al.* [169] showed that the lower layer effective receptive fields for ViT are larger than those in ResNets, and by design also than those in Swin transformer. This however results in weaker inductive bias, and therefore requires a large amount of data to effectively train on. Swin transformer is a hierarchical transformer where self-attention is calculated within a local sliding window, leading to stronger inductive bias and requires comparatively less data for training. This makes ViT as the feature extractor/backbone worse suited for continual DA with single pass network updates on limited amount of data, compared to both ResNet-50 and Swin.

4.6 Conclusions

This work introduces a new paradigm of unsupervised CL for domain adaptation where a source-trained model adapts to target domain data that are received continually in small batches. We tackle this problem



(a) Accuracy of ConDA with *Swin* backbone, both with, and without gradient normalization on the final target data as it continually adapts to the incoming batches of intermediate and target domains of AID-CC.



(b) Accuracy of UCL-GV with Swin backbone, both with, and without gradient normalization on the final target data as it continually adapts to the incoming batches of intermediate and target domains of AID-CC.

Figure 4.19: Continual models ConDA and UCL-GV with Swin backbone, showing increase in adaptation stability and final accuracy due to gradient normalization.

by combining source-free DA with buffer management and sample replay inspired from CL research. We introduce ConDA as the first DA method to address such a setting. In ConDA, we selectively store samples in a buffer and replay them with the incoming batches to improve our network's generalization capabilities for the overall target domain. We also propose a novel loss function that improves the overall performance of our network. Our results demonstrate that ConDA outperforms existing SOTA DA methods under continual settings on various datasets at a fraction of the standard DA data storage requirements. We extend ConDA for multiple target domains and our method beats the baseline and continual SOTA methods. We further explore continual learning under gradually varying domains, and propose UCL-GV that utilizes a memory buffer for sample replay in a manner similar to ConDA, but with a first-in-first-out strategy, while utilizing a contrastive loss for domain alignment between the buffer and the incoming samples in each iteration. UCL-GV outperforms SOTA continual DA on gradually varying domains on two benchmark datasets. We propose four datasets for evaluating continual DA models under gradually degrading weather conditions

for domain adaptation from a clean weather domain to a severely worsened weather domain. We identify stability issues in continual DA models, and propose a simple trick of gradient normalization for increasing stability.

Chapter 5

Curriculum-Guided Domain Adaptation in the Dark

This chapter is based on the paper titled ”Curriculum Guided Domain Adaptation in the Dark” [82], which has been published in the IEEE Transactions on Artificial Intelligence journal. Addressing the rising concerns of privacy and security, domain adaptation in the dark aims to adapt a black-box source trained model to an unlabeled target domain without access to any source data or source model parameters. The need for domain adaptation of black-box predictors becomes even more pronounced to protect intellectual property as deep learning based solutions are becoming increasingly commercialized. Current methods distill noisy predictions on the target data obtained from the source model to the target model, and/or separate clean/noisy target samples before adapting using traditional noisy label learning algorithms. However, these methods do not utilize the easy-to-hard learning nature of the clean/noisy data splits. Also, none of the existing methods are end-to-end, and require a separate fine-tuning stage and an initial warmup stage. In this work, we present **C**urriculum **A**daptation for **B**lack-**B**ox (**CABB**), which provides a curriculum guided adaptation approach to gradually train the target model, first on target data with high confidence (clean) labels, and later on target data with noisy labels. CABB utilizes Jensen-Shannon divergence as a better criterion for clean-noisy sample separation, compared to the traditional criterion of cross entropy loss. Our method utilizes co-training of a dual-branch network to suppress error accumulation resulting from confirmation bias. The proposed approach is end-to-end trainable and does not require any extra finetuning stage, unlike existing methods. Empirical results on standard domain adaptation datasets show that CABB outperforms existing state-of-the-art black-box DA models and is comparable to white-box domain adaptation models.

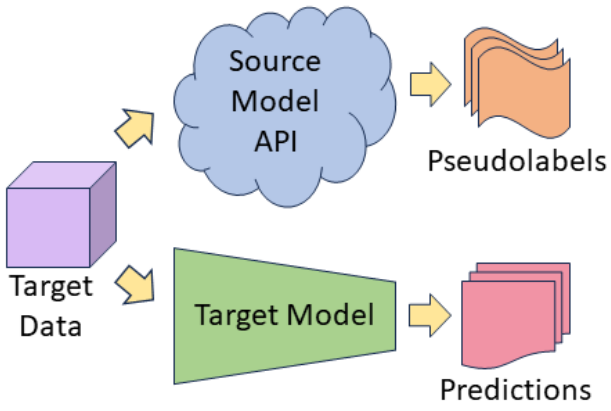


Figure 5.1: Overview of BBDA, where the source model parameters are not available during adaptation. The source model may only be accessed as a black box to generate pseudolabels for the unlabeled target data. These pseudolabels may be used to adapt the target model on the target domain without true labels.

5.1 Introduction

Source-Free UDA [125, 221] has recently emerged to address cases where the adaptation process utilizes only a model trained on the source data, without accessing the source data. Such methods still fail to adequately alleviate data privacy and security concerns as model attacks may potentially retrieve the raw source data or corrupt the model. Moreover, with the commercialization of deep learning based solutions, companies may be reluctant to share their proprietary model parameters with the end users. These issues brought forth a newer UDA paradigm called black-box domain adaptation (BBDA) that adapts without accessing neither the source data, nor the source model parameters [126]. Practically, a vendor can have the source trained model as an API in the cloud, and the end user can access the black-box source model to generate predictions for each unlabelled target instance to adapt on the target domain.

Existing BBDA methods transfer knowledge from the source trained model predictions to the target model, and then finetune the target model on the target data [126, 219]. The approach in [219] utilizes a noisy label learning (NLL) algorithm [115] to separate the target domain into an easy-to-adapt subdomain with cleaner pseudolabels, and a hard-to-adapt subdomain with noisier pseudolables using low cross-entropy (CE) loss criterion as the separator [56], and then applies supervised and semi-supervised learning strategies on the easy- and hard-to-adapt subdomains, respectively.

In this work, we propose ***C**urriculum **A**daptation for **B**lack-**B**ox **(CABB)*** as an unsupervised domain adaptation framework for black-box predictors. We present Jensen-Shannon distance (JSD) as a better criterion to separate clean and noisy samples using pseudolabels generated by the source model. JSD can be modelled

using a two-component Gaussian Mixture Model (GMM), where the distribution with the lower distance can be considered to be consisting of cleaner samples and that with the higher distance contains noisier samples. As opposed to traditional low loss criterion for clean-noisy separation, low JSD criterion produces a more conservative, but more accurate clean sample set. To reduce error accumulation from confirmation bias, CABB employs co-training [56, 115] two identical networks and adapts one network on the clean-noisy separated sets generated by the other, and vice versa. CABB introduces a curriculum learning strategy to adaptively learn from the clean samples first, and the noisy samples later during the adaptation process. CABB foregoes the finetuning stage of existing methods by utilizing mutual information maximization [125, 198] within its curriculum, making it end-to-end adaptable. The main contributions of our work are as follows.

- We introduce CABB as a curriculum guided domain adaptation model that progressively learns from the clean target set and the noisy target set, while utilizing co-training of a dual-branch network to suppress error accumulation resulting from confirmation bias.
- We identify Jensen-Shannon divergence loss as a better criterion than cross-entropy loss for separation of clean and noisy samples for BBDA.
- CABB incorporates mutual information maximization within its curriculum and makes the adaptation process end-to-end without the need for any separate finetuning stage.
- CABB produces robust pseudolabels from the mean of an ensemble of predictions generated by the two branches of the network on a set of augmentations.

5.2 Related Work

Table 5.1: Methodology comparison between CABB and existing BBDA methods.

Model	Distillation	Co-teaching	Sample splitting	Curriculum learning	Fine-tuning
DINE [126]	✓	✗	✗	✗	✓
BETA [219]	✓	✓	CE loss	✗	✓
CABB	✓	✓	Jensen-Shannon distance	✓	✗

5.2.1 Unsupervised domain adaptation

Domain gap or domain shift occurs when the data distribution of the training data (source domain) is considerably different from that of the testing data (target domain) [200]. Long *et al.* [138], and Tzeng *et*

al. [202] proposed to mitigate this distribution shift by minimizing the maximum mean discrepancy (MMD) between the two distributions, while Zellinger *et al.* [229] proposed to match the higher order central moments of source and target probability distributions, and thus minimize central moment discrepancy (CMD) for UDA. Sun and Saenko [192] devised Deep CORAL to minimize second-order distribution statistics to mitigate domain shift. Ganin *et al.* [49] utilized a domain discriminator module, and introduced gradient reversal layer (GRL) to adversarially align the two distributions. Many methods followed since then that have utilized adversarial alignment on the latent feature space. [140, 166]. While [49] uses a common encoder for the source and target data, Tzeng *et al.* [201] proposed to decouple the encoders by first training an encoder and a classifier on the labelled source data, followed by training a separate target data encoder using a domain discriminator, and finally deploying the same source classifier as the target classifier. Hoffman *et al.* [69] produced source-like images using generative image-to-image translation [239] and adversarially-aligned source and target data distributions at the low-level or pixel-level. Global domain-wise adversarial alignment, however, may cause loss of intrinsic target class discrimination in the embedding space, and lead to suboptimal performance. To preserve class-wise feature discrimination, Li *et al.* [120] simultaneously aligned the domain-wise and class-wise distributions across the source and target data by solving two complementary domain-specific and class-specific minimax problems. In a non-adversarial approach, Pan *et al.* [163] proposed to calculate the source class prototypes for the labelled source data, and target class prototypes from the pseudo-labelled target data, and then enforce consistency on the prototypes in the embedding space. Tang *et al.* [194] similarly bases structural domain similarity to enforce structural source regularization and conducts discriminative clustering of target data without any domain alignment.

5.2.2 Source-free domain adaptation

Although domain divergence minimization [200, 202, 229], adversarial adaptation [49, 69], and optimal transport [19,217] are widely used techniques for UDA, they require access to both the source and target data during adaptation. Addressing situations where source data is unavailable, several source-free DA (SFDA) methods have been proposed recently. Chidlovskii *et al.* [27] proposed to use a few source prototypes or representatives in place of the entire source data for semi-supervised domain adaptation. Liang *et al.* [124] proposed to conduct target adaptation using source-free distant supervision to iteratively find taret pseudo-labels, a domain invariant subspace where the source and target data centroids are only moderately shifted, and finally target centroids/prototypes by implementing an alternating minimization strategy. Liang *et al.* [125] introduced SHOT as an SFDA framework which transfers the source hypothesis or classifier to the target model, and adapts via self-training with information maximization [74, 103, 187] and class centroid-based pseudolabel refinement. Yang *et al.* [221] proposed G-SFDA which refines the pseudolabels further via consistency regularization among neighboring target samples. Ding *et al.* [36] introduced SFDA-DE

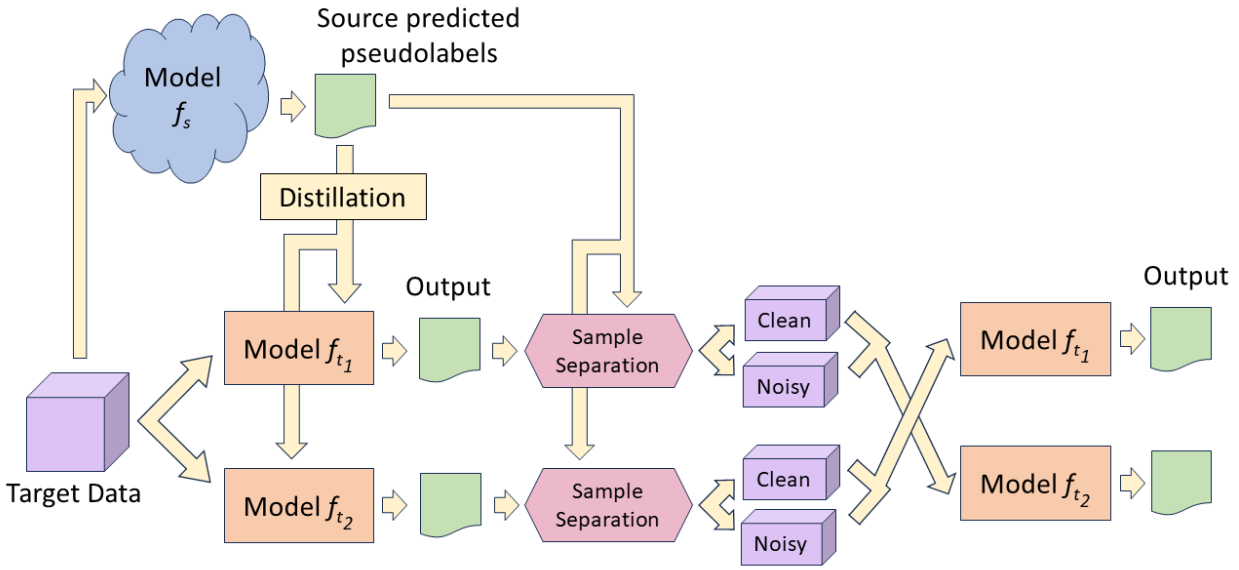


Figure 5.2: UDA pipeline in CABB. The target data is fed to the source model f_s and the knowledge generated from f_s is transferred to both target branches f_{t_1} and f_{t_2}. The source predicted pseudolabels are also used to calculate JSD and produce clean-noisy sample sets. In subsequent co-training of f_{t_1} and f_{t_2}, the samples sets created by one branch are used to update the other branch, using curriculum guided losses to progressively adapt to clean samples first, and the noisy samples later.

which samples from an estimated source data distribution, and conducts contrastive alignment between the estimated source and target distributions. Yang *et al.* [220] proposed BAIT that utilizes maximum classifier discrepancy [182] for SFDA after separating the target samples into certain and uncertain sets using entropy as the criterion. This approach is similar to identifying novel class samples in the open set versions of [125, 222]. However, BAIT [220] did not conduct any data distribution modeling, and simply split the target samples into half for each set without considering the possible high noise rate.

5.2.3 Black box domain adaptation

Extending the premise of SFDA further, Liang *et al.* [126] introduced a newer paradigm of black box DA where, in addition to the source data, the source model parameters are also unavailable during adaptation. This new challenging scenario is important to protect intellectual property (source model parameters) from the end users. Liang *et al.* [126] proposed DINE which distills knowledge from the black-box source model to the target model in the first stage, followed by finetuning with target pseudolabels in the second stage. Yang *et al.* [219] proposed BETA as a method that separates easy- and hard-to-learn pseudolabels using a conventional noisy label learning technique [56], and applies a twin-network co-training strategy similar

to [115], and adversarial alignment during adaptation.

5.2.4 Curriculum learning

Bengio *et al.* [8] introduced curriculum learning as a method of training a model with increasingly complex data samples, to mimic the human learning process. In practice, a difficulty criterion is utilized to rank the training samples from easy to hard. Model training begins with the easy-to-learn samples, and a scheduler decides when to update the curriculum, i.e when to incorporate harder-to-learn samples in the training objective during the process. This method results in faster convergence, and achieves better local minima, as evidenced by its superior performance compared to training a model with the standard random sampling approach [8].

While [8] treated the complexity of geometric shapes (basic vs. intricate) as the measure for selecting easy-to-hard samples and applied curriculum learning strategy for image classification task, Spitkovsky *et al.* [189] considered the length of sentences (short vs long) as the data separation criterion for curriculum learning in Natural Language Processing (NLP) tasks, and Braun *et al.* [12] used signal-to-noise ratio for the sample ranking criterion in speech recognition.

In terms of curriculum learning for domain adaptation, Roy *et al.* [177] utilized mean entropy as the domain ranking measure in a multi-target domain adaptation setting, and applied curriculum learning strategy with a graph convolution network to consecutively adapt to easier target domains first, and harder target domains later. Zhang *et al.* [231] and Zhan *et al.* [234] used curriculum learning for training on unlabelled *auxiliary* data for semi-supervised domain adaptation for neural machine translation.

5.2.5 CABB comparison with other BBDA methods

In relation to other works, CABB adopts the process of source model distillation to the target model and subsequent exponential moving average updates for pseudo-label refinement during adaptation, as done in DINE [126]. CABB also uses co-teaching of a dual-branch network [56, 115] for reducing confirmation bias in sample separation. We identify Jensen-Shannon distance (JSD) as a more appropriate criterion for clean-noisy sample separation for the unbounded noise rate in UDA, compared to existing BBDA method BETA [219] that uses the low CE loss for the bounded noise rate in NLL, and no such clean-noisy separation as in DINE [126].

As opposed to BETA which makes no distinction between the weights given to losses from clean or noisy

samples, we acknowledge the varying impacts of clean and noisy samples and formulate a curriculum learning strategy to train the target model end-to-end with cleaner samples first, and progressively with noisy samples later. CABB is also end-to-end trainable as it foregoes any final fine-tuning stage used in DINE and BETA. In addition, we use ensemble-based pseudolabeling using a series of weak and strong augmentations using *AutoAugment* [30], and utilize a mix of active-passive losses (normalized cross-entropy and reverse cross-entropy) [144] for adaptation on the noisy sample subset. A brief comparison of our CABB method against existing BBDA methods is presented in Table 5.1.

5.3 Methodology

The black-box source model $f_s(\theta_s) : \mathcal{X}_s \rightarrow \mathcal{Y}_s$ with model parameters θ_s, maps the multiclass source data $x_s \in \mathcal{X}_s$ of source domain $\mathcal{D}_s$, to the label space $y_s \in \mathcal{Y}_s$. For BBDA, we however do not have access to θ_s, but only the hard predictions $(\hat{y}_t \in \mathcal{Y}_t) = f_s(\theta_s, x_t)$ from f_s on the target data $x_t \in \mathcal{X}_t$ of target domain $\mathcal{D}_t$. There exists a domain shift between the source data distribution $\mathcal{D}_s$ and the target data distribution $\mathcal{D}_t$, while the label space is shared, i,e $\mathcal{Y}_s = \mathcal{Y}_t$. Due to this domain shift, a large number of predictions $\hat{y}_t$ may be incorrect and could result in a set of noisy pseudolabels generated by the source model. Our objective for DA is to learn a mapping function $f_t(\theta_t) : \mathcal{X}_t \rightarrow \mathcal{Y}_t$.

Research has shown that when deep networks are trained with noisy labels, the resulting models tend to memorize the wrongly-labelled samples owing to confirmation bias, as the training progresses [232]. Furthermore, in regular training of a single-branch network with noisy labels, the error from one training mini-batch flows back into the network itself for the next mini-batch, and thus the error increasingly accumulates [56]. In this work, during adaptation, we employ co-teaching [56] of a dual-branch network [115, 219] to mitigate error accumulation, resulting from the confirmation bias. In co-teaching, due to the difference in branch parameters of the dual-branch design, error introduced by the noisy pseudolabels in one branch can be filtered out by the other branch. In practice, one branch conducts the clean-noisy sample separation for the other branch, and vice versa. Since each branch generates different sets of clean and noisy samples, co-teaching breaks the flow of error through the network, and thus error accumulation attenuates. To simplify notation, the dual target branches/models f_{t_1} and f_{t_2} may be represented by f_t in later parts of this paper. Both networks are trained/adapted, and the final inference can be taken from either one.

We follow [126] to distill knowledge from the source model predictions to the target model in a teacher-student manner via Kullback-Leibler (KL) divergence loss and information maximization loss [125] (equations 6.10 and 6.11), at the beginning of each epoch throughout the adaptation process. However, unlike [126], we only have access to the hard predictions from the source model. Similar to [126], the source

model predictions $\hat{y}_t^i$ are updated during adaptation at certain intervals via temporal ensembling by exponential moving average (EMA) between the source model predicted pseudolabels $\hat{y}_t^i$ and the target model predicted pseudolabels y_t^i.

It is to be noted that, the purpose of distillation is not to initialize the target model in the absence of source model parameters; rather distillation acts as an *anchor* from the source model to the target model during adaptation. Distillation continues to take place from the temporally-ensembled source (teacher) generated pseuduolabels to the target (student) model, which has been shown to improve generalization [196]. The process of generating y_t^i is described in section 5.3.2.

5.3.1 Clean-noisy separation

The predictions $\hat{y}_t$ generated by the black box source model f_s are noisy and unreliable due to domain shift between $\mathcal{D}_s$ and $\mathcal{D}_t$. Research on learning with noisy labels shows that deep learning models tend to fit on the clean samples first, and on the noisy samples later during training [5, 115]. We follow this insight and separate the target domain data into a clean sample set $\mathcal{X}_{tc}$ with reliable predictions, and a noisy sample set $\mathcal{X}_{tn}$ with unreliable predictions. In traditional noisy label settings, the noisy labels are either caused by wrong annotations from humans or from image search engines. The noise rate is, therefore, bounded. However, as the noisy labels in UDA are generated by the source model, the noise rate in this case is unbounded and can approach unity [225]. We propose Jensen-Shannon distance (JSD) [40] between the source predicted hard labels $\hat{y}_t^i$ and the target model class probabilities as the criterion for clean-noisy sample separation under unbounded noise rate. JSD is calculated as,

$$JSD(\hat{y}_t^i, p_t^i) = \frac{1}{2} KL(\hat{y}_t^i, \frac{\hat{y}_t^i + p_t^i}{2}) + \frac{1}{2} KL(p_t^i, \frac{p_t^i + \hat{y}_t^i}{2}) \tag{5.1}$$

where, $KL(a, b)$ is the Kullback-Leibler divergence between a and b, and p_t^i is the target model output probability for target sample x_t^i. Compared to cross-entropy loss, JSD is symmetric by design, and ranges between 0 and 1, thus becoming less susceptible to noise. When applied to the network response, JSD produces a bimodal distribution, which is modelled by a two-component Gaussian Mixture Model (GMM) with equal priors. In DA, the target model may *confidently* categorize an image as the wrong class with very high prediction probability. Therefore, this is a poor criterion for identifying whether a sample is clean or noisy. For the potentially unbounded pseudolabel noise rate in BBDA, we take the probability of belonging to the JSD Gaussian distribution with the lower mean value as the confidence metric of being a clean sample in our clean-noisy sample separation stage. Empirically, we apply a threshold δ_t on our confidence score of belonging to the lower-mean GMM distribution to select our clean sample set $\mathcal{X}_{tc}$, at the beginning of each

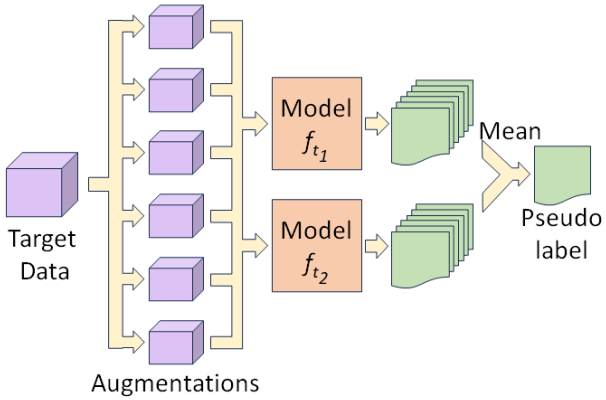


Figure 5.3: Ensemble-based pseudolabeling in CABB. Each sample is augmented to produce 6 different views that are fed through both branches f_{t_1} and f_{t_2} to create a total of 12 output predictions, which are then averaged to produce the soft pseudolabel for co-training f_{t_1} and f_{t_2}.

epoch for adaptation. The remaining target samples are included in the noisy label set $\mathcal{X}_{tn}$.

5.3.2 Ensemble based pseudolabeling

In order to produce robust target model pseudolabels y_t^i, we apply a series of augmentations on the target samples based on *AutoAugment* [30] and produce an ensemble of output prediction probabilities from our two target models. We give equal weights to each output prediction and take the mean of the outputs as the soft pseudolabel as follows.

$$y_t^i = \frac{1}{2M} \sum_{0}^{M} f_{t_1}(x_{t_m}^i) + f_{t_2}(x_{t_m}^i) \tag{5.2}$$

where M is the number of augmentations for the i-th target sample. The predictions are further sharpened with a temperature factor $T(0 < T < 1)$ and then normalized as follows.

$$y_t^i = \frac{(y_t^i)^{\frac{1}{T}}}{\sum_C (y_t^{iC})^{\frac{1}{T}}} \tag{5.3}$$

where y_t^{iC} is the C-th dimensional value of the pseudolabel vector y_t^i.

5.3.3 Curriculum-guided noisy learning

In order to mitigate early training time memorization [5] induced from noisy labels during the adaptation of deep models, we introduce a curriculum-guided learning to train the target model on the clean samples first, and on the noisy samples later. As the adaptation/training progresses, more noisy samples are reclassified as clean samples.

We employ separate training losses for the clean and noisy sample set. The clean set is trained with standard cross-entropy (CE) loss as follows.

$$\mathcal{L}_{tc}(f_t; \mathcal{X}_{tc}) = -\mathbb{E}_{x_t^i \in \mathcal{X}_{tc}} \sum_{k=1}^{C} y_{t_k}^i log(\sigma_k(f_t(x_t^i))) \tag{5.4}$$

where $\sigma_k(a) = \frac{exp(a_k)}{\sum_i exp(a_i)}$ is the softmax function and C is the number of classes. For the noisy set, we minimize a combination of active-passive losses [144] constructed of normalized cross-entropy loss $\mathcal{L}_{tn_{NCE}}$ and reverse cross-entropy loss $\mathcal{L}_{tn_{RCE}}$. [144] showed that such normalization makes a model robust to noisy data. Reverse cross-entropy loss is applied to avoid any underfitting on the noisy set. Due to the unbounded nature of noise rate in UDA and conservative clean-noisy separation criteria in CABB, we employ this particular combination of active-passive losses as our noisy set loss $\mathcal{L}_{tn}$ to make target training/adaptation robust and comprehensive on the noisy sample set. The loss function is expressed as follows.

$$\mathcal{L}_{tn_{NCE}}(f_t; \mathcal{X}_{tn}) = -\mathbb{E}_{x_t^i \in \mathcal{X}_{tn}} \frac{\sum_{k=1}^{C} y_{t_k}^i log(\sigma_k(f_t(x_t^i)))}{\sum_{j=1}^{C} \sum_{k=1}^{C} y_{t_j}^i log(\sigma_k(f_t(x_t^i)))} \tag{5.5}$$

$$\mathcal{L}_{tn_{RCE}}(f_t; \mathcal{X}_{tn}) = -\mathbb{E}_{x_t^i \in \mathcal{X}_{tn}} \sum_{k=1}^{C} \sigma_k(f_t(x_t^i)) log(y_{t_k}^i) \tag{5.6}$$

$$\mathcal{L}_{tn} = \mathcal{L}_{tn_{NCE}} + \beta \mathcal{L}_{tn_{RCE}} \tag{5.7}$$

where β is a hyperparameter.

To promote learning of clean samples first and to mitigate noisy label memorization, target training is done under curriculum guidance [8]. Based on the success of the clean-noisy sample separation, the pseudolabels

in the clean sample set $\mathcal{X}_{tc}$ are more likely to be correct, while those in the noisy sample set $\mathcal{X}_{tn}$ have a much higher noise rate. Therefore, a deep network tends to easily learn from the unambiguous $\mathcal{X}_{tc}$ set. We set a curriculum factor γ_n according to the following equation.

$$\gamma_n = \gamma_{n-1}(1 - \alpha\epsilon^{-L_{tcn}/L_{tc_{n-1}}}) \tag{5.8}$$

where, α is a hyperparameter and n is the iteration number. γ_{n-1} is the curriculum factor for the previous iteration. The ratio $L_{tc_n}/L_{tc_{n-1}}$ determines how much the curriculum factor decreases from iteration $n-1$ to n. If the CE loss on the clean set increases, γ decreases by a small value to allow for further training on the clean set in the subsequent iterations. But if the CE loss decreases by a large margin, γ decreases accordingly to accommodate learning from the noisy sample set in the coming iterations. Our curriculum guidance balances the supervised and unsupervised losses on the respective clean and noisy sets as follows.

$$\mathcal{L}_t = \gamma_n\mathcal{L}_{tc} + (1 - \gamma_n)\mathcal{L}_{tn} \tag{5.9}$$

We adopt the formulation of information maximization (IM) loss [103, 125, 187] from [198] to help our model produce precise predictions, while maintaining a global diversity across all classes in the output predictions. The IM loss is a combination of the following entropy loss $\mathcal{L}_{ent}$ and equal diversity loss $\mathcal{L}_{eqdiv}$.

$$\mathcal{L}_{ent}(f_t; \mathcal{X}_t) = -\mathbb{E}_{x_t^i \in \mathcal{X}_t} \sum_{k=1}^{C} \sigma_k(f_t(x_t^i))log(\sigma_k(f_t(x_t^i))) \tag{5.10}$$

$$\mathcal{L}_{eqdiv}(f_t; \mathcal{X}_t) = \sum_{k=1}^{C} q_k log\left(\frac{q_k}{\hat{q}_k}\right) \tag{5.11}$$

where $\hat{q}_k = \mathbb{E}_{x_t^i \in \mathcal{X}_t}[\sigma(f_t(x_t))]$ is the mean of the softmax of the target network output response. $\mathcal{L}_{eqdiv}$ conducts KL divergence between $\hat{q}_k$ and the ideal uniform response q_k. Our curriculum guided IM loss is as follows.

$$\mathcal{L}_{IM} = \mathcal{L}_{eqdiv} + (1 - \gamma_n)\mathcal{L}_{ent} \tag{5.12}$$

Minimization of entropy loss $\mathcal{L}_{ent}$ is gradually activated as the model sufficiently adapts to the clean sample. Such curriculum guidance ensures that the potentially erroneous predictions produced in the early stages of self-training are not accumulated. The $\mathcal{L}_{eqdiv}$ loss enforces diversity in the output predictions throughtout

the training process. The overall objective function is,

$$\mathcal{L}_{tot} = \mathcal{L}_t + \mathcal{L}_{IM} \tag{5.13}$$

A brief demonstration of the CABB pipeline can be found in Algorithm 4.

Algorithm 3: Pseudocode for CABB

Input: Black-box source trained model f_s and target data $x_t^i \in \mathcal{X}_t$
Output: Target adapted model f_t
Initialization: Dual target models f_{t_1} and f_{t_2}

```
for epoch = 1 to epoch_total do
    while m ≤ iter_distill do
        Distill from teacher f_s to students f_t1 and f_t2 following [126]
    end
    Conduct clean(X_tc)-noisy(X_tn) sample separation using JSD from model f_t1 for f_t2 and vice-versa
    for f_t ∈ f_t1, f_t2 do
        while n ≤ iter_adapt do
            Get ensemble averaged pseudolabels y_t^i ∈ Y_t from equations 6.2 and 5.3
            Calculate L_tc on X_tc, L_tn on X_tn, and L_ent and L_eqdiv on (X_tc, X_tn) ∈ X_t using equations
              5.4, 5.7, 6.10, and 6.11 respectively
            Calculate γ_n using equation 5.8
            Calculate L_t and L_IM using equations 5.9 and 5.12
            Optimize f_t with loss L_tot using equation 5.13
        end
    end
end
```

5.4 Experimental setup

5.4.1 Datasets

We evaluate CABB on three popular domain adaptation datasets viz. Office-31 [180], Office-Home [204], and VisDA-C [167]. These datasets have been described in Chapter 4.

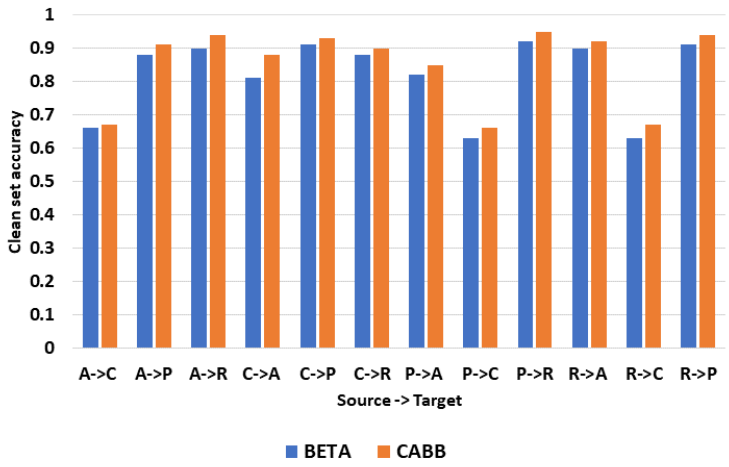


Figure 5.4: Accuracy on the clean sample set achieved via clean-noisy sample separation using low JSD (CABB) vs low CE (BETA), after distillation from the source teacher at the first epoch.

Table 5.2: Mean accuracy on the Office31. 'SF' refers to source-free and 'BB' means black-box. The top performing results among the BBDA methods are in bold letters.

Method	SF	BB	A→D	A→W	D→A	D→W	W→A	W→D	Mean
DANN [49]	✗	✗	79.7	82.0	68.2	96.9	67.4	99.1	82.2
ALDA [21]	✗	✗	94.0	95.6	72.2	97.7	72.5	100.0	88.7
GVB-GD [31]	✗	✗	95.0	94.8	73.4	98.7	73.7	100.0	89.4
SRDC [194]	✗	✗	95.8	95.7	76.7	99.2	77.1	100.0	90.9
SHOT [125]	✓	✗	94.0	90.1	74.7	98.4	74.3	99.9	88.6
A^2Net [214]	✓	✗	94.5	94.0	76.7	99.2	76.1	100	90.1
SFDA-DE [36]	✓	✗	96.0	94.2	76.6	98.5	75.5	99.8	90.1
LNL-OT [226]	✓	✓	88.8	85.5	64.6	95.1	66.7	98.7	83.2
LNL-KL [233]	✓	✓	89.4	86.8	65.1	94.8	67.1	98.7	83.6
HD-SHOT [127]	✓	✓	86.5	83.1	66.1	95.1	68.9	98.1	83.0
SD-SHOT [127]	✓	✓	89.2	83.7	67.9	95.3	71.1	97.1	84.1
DINE [126]	✓	✓	91.6	86.8	72.2	96.2	73.3	98.6	86.4
BETA [219]	✓	✓	93.6	88.3	**76.1**	95.5	**76.5**	99.0	88.2
CABB (Ours)	✓	✓	**94.0**	**88.6**	**76.0**	**97.9**	76.0	**99.6**	**88.7**

Table 5.3: Mean accuracy on the Office-Home dataset. 'SF' refers to source-free and 'BB' means black-box. The top performing results among the BBDA methods are in bold letters.

Method	SF	BB	A→C	A→P	A→R	C→A	C→P	C→R	P→A	P→C	P→R	R→A	R→C	R→P	Mean
DANN [49]	✗	✗	45.6	59.3	70.1	47.0	58.5	60.9	46.1	43.7	68.5	63.2	51.8	76.8	57.6
ALDA [21]	✗	✗	53.7	70.1	76.4	60.2	72.6	71.5	56.8	51.9	77.1	70.2	56.3	82.1	66.6
GVB-GD [31]	✗	✗	57.0	74.7	79.8	64.6	74.1	74.6	65.2	55.1	81.0	74.6	59.7	84.3	70.4
SRDC [194]	✗	✗	52.3	76.3	81.0	69.5	76.2	78.0	68.7	53.8	81.7	76.3	57.1	85.0	71.3
FixBi [158]	✗	✗	58.1	77.3	80.4	67.7	79.5	78.1	65.8	57.9	81.7	76.4	62.9	86.7	72.7
G-SFDA [221]	✓	✗	57.9	78.6	81.0	66.7	77.2	77.2	65.6	56.0	82.2	72.0	57.8	83.4	71.3
SHOT [125]	✓	✗	57.1	78.1	81.5	68.0	78.2	78.1	67.4	54.9	82.2	73.3	58.8	84.3	71.8
HCL [75]	✓	✗	64.0	78.6	82.4	64.5	73.1	80.1	64.8	59.8	75.3	78.1	69.3	81.5	72.6
A^2Net [214]	✓	✗	58.4	79.0	82.4	67.5	79.3	78.9	68.0	56.2	82.9	74.1	60.5	85.0	72.8
SFDA-DE [36]	✓	✗	59.7	79.5	82.4	69.7	78.6	79.2	66.1	57.2	82.6	73.9	60.8	85.5	72.9
LNL-OT [226]	✓	✓	49.1	71.7	77.3	60.2	68.7	73.1	57.0	46.5	76.8	67.1	52.3	79.5	64.9
LNL-KL [233]	✓	✓	49.0	71.5	77.1	59.0	68.7	72.9	56.4	46.9	76.6	66.2	52.3	79.1	64.6
HD-SHOT [127]	✓	✓	48.6	72.8	77.0	60.7	70.0	73.2	56.6	47.0	76.7	67.5	52.6	80.2	65.3
SD-SHOT [127]	✓	✓	50.1	75.0	78.8	63.2	72.9	76.4	60.0	48.0	79.4	69.2	54.2	81.6	67.4
DINE [126]	✓	✓	52.2	78.4	81.3	65.3	76.6	78.7	62.7	49.6	82.2	69.8	55.8	84.2	69.7
BETA [219]	✓	✓	57.2	78.5	**82.1**	**68.0**	78.6	**79.7**	67.5	56.0	**83.0**	71.9	58.9	84.2	72.1
CABB (Ours)	✓	✓	**57.4**	**79.5**	**82.0**	**68.1**	**79.3**	78.8	**68.2**	**57.9**	82.7	**73.6**	**60.0**	**86.4**	**72.8**

5.4.2 Implementation details

We follow the same protocol in [126, 219] for source training to ensure fairness for comparison. Our target models are initialized with ImageNet pretrained weights, since source model parameters are inaccessible. For Office-31 and Office-Home, we use ResNet50, and for VisDA-C we use ResNet101 as the backbone [63], on top of which we attach an MLP-based classifier, similar to [126, 219]. The target models are trained with SGD optimizer with 0.9 momentum and weight decay $1e^{-3}$. The learning rate for the backbone is set to $1e^{-3}$, while that of the classifier is set to $1e^{-2}$. α in the curriculum factor is set to $2e^{-3}$ for Office-31, and $2e^{-4}$ for Office-Home and VisDA-C, depending on the size of the dataset. The model is adapted for 50 epochs for Office-31 and Office-Home datasets, and for five epochs for the VisDA-C dataset. Temperature sharpening factor T is set to 0.5. We implement our method using the PyTorch library on an NVIDIA-A100 GPU.

Table 5.4: Mean per-class accuracy on the VisDA-C dataset. 'SF' refers to source-free and 'BB' means black-box. The top performing results among the BBDA methods are in bold letters.

Method	SF	BB	plane	bcycl	bus	car	horse	knife	mcycle	person	plant	sktbrd	train	truck	Per-class
DANN [49]	✗	✗	81.9	77.7	82.8	44.3	81.2	29.5	65.2	28.6	51.9	54.6	82.8	7.8	57.6
ALDA [21]	✗	✗	93.8	74.1	82.4	69.4	90.6	87.2	89.0	67.6	93.4	76.1	87.7	22.2	77.8
SHOT [125]	✓	✗	94.3	88.5	80.1	57.3	93.1	94.9	80.7	80.3	91.5	89.1	86.3	58.2	82.9
A^2Net [214]	✓	✗	94.0	87.8	85.6	66.8	93.7	95.1	85.8	81.2	91.6	88.2	86.5	56.0	84.3
SFDA-DE [36]	✓	✗	95.3	91.2	77.5	72.1	95.7	97.8	85.5	86.1	95.5	93.0	86.3	61.6	86.5
LNL-OT [226]	✓	✓	82.6	84.1	76.2	44.8	90.8	39.1	76.7	72.0	82.6	81.2	82.7	50.6	72.0
LNL-KL [233]	✓	✓	82.7	83.4	76.7	44.9	90.9	38.5	78.4	71.6	82.4	80.3	82.9	50.4	71.9
HD-SHOT [127]	✓	✓	75.8	85.8	78.0	43.1	92.0	41.0	79.9	78.1	84.2	86.4	81.0	65.5	74.2
SD-SHOT [127]	✓	✓	79.1	85.8	77.2	43.4	91.6	41.0	80.0	78.3	84.7	86.8	81.1	65.1	74.5
DINE [126]	✓	✓	81.4	86.7	77.9	55.1	92.2	34.6	80.8	79.9	87.3	87.9	84.3	58.7	75.6
BETA [219]	✓	✓	**96.2**	83.9	82.3	71.0	**95.3**	73.1	**88.4**	80.6	**95.5**	90.9	**88.3**	45.1	82.6
CABB (Ours)	✓	✓	95.1	**87.0**	**82.6**	**71.5**	94.5	**89.7**	87.5	**81.5**	93.8	**92.4**	87.3	**55.5**	**84.9**

Table 5.5: Performance evaluation of curriculum adaptation involving different parts of CABB on the VisDA-C dataset. The 'tick' marks mean the part is present in the model, and the 'cross' mark means that part is absent. When curriculum is absent and $\mathcal{L}_{tn}$ is present, γ_n is set to 0.5. $*$ refers to replacement of our active-passive loss with standard cross-entropy loss for noisy samples.

Curriculum	$\mathcal{L}_{tn}$	$\mathcal{L}_{ent}$	plane	bcycl	bus	car	horse	knife	mcycle	person	plant	sktbrd	train	truck	Per-class
✗	✗	✗	98.0	93.1	79.1	41.8	97.1	81.6	79.5	79.9	93.3	91.1	90.3	49.5	81.2
✗	✗	✓	98.2	89.2	82.2	58.1	97.2	83.5	84.3	71.3	95.8	92.2	90.4	18.1	80.0
✗	$*$	✓	96.2	86.6	83.2	71.1	95.5	90.1	85.1	80.1	93.0	91.9	84.3	40.1	83.1
✗	✓	✓	97.1	82.3	85.0	79.1	91.7	93.2	89.0	77.7	94.4	92.5	83.9	1.2	80.6
✓	✗	✓	97.3	89.9	78.3	60.1	96.4	76.1	80.2	77.3	93.5	90.0	88.7	52.8	81.7
✓	$*$	✓	96.0	85.2	80.9	68.2	95.0	85.0	86.0	79.6	93.2	92.1	88.7	55.8	83.8
✓	✓	✗	95.2	85.9	83.5	68.9	93.8	88.6	83.6	80.7	95.1	92.0	86.0	56.7	84.2
✓	✓	✓	95.1	87.0	82.6	71.5	94.5	89.7	87.5	81.5	93.8	92.4	87.3	55.5	84.9

Table 5.6: Performance evaluation of CABB with different kinds of pseudolabeling schemes, and the impact of dual-branched co-training on error accumulation for VisDA-C dataset.

Dual-branch co-training	Pseudolabeling	plane	bcycl	bus	car	horse	knife	mcycle	person	plant	sktbrd	train	truck	Per-class
✓	Model Predictions	91.2	68.5	68.0	48.1	87.0	1.2	41.7	24.1	73.1	56.5	78.3	46.2	57.0
✓	Clustering [14, 125]	95.0	86.0	83.4	60.7	93.7	20.8	85.4	81.0	89.5	81.5	86.6	50.7	76.2
✓	Ensemble	95.1	87.0	82.6	71.5	94.5	89.7	87.5	81.5	93.8	92.4	87.3	55.5	84.9
✗	Ensemble	97.8	91.6	78.8	48.1	96.3	77.7	81.2	79.1	94.8	90.3	89.4	51.4	81.4

5.5 Results

5.5.1 Overall evaluation

Liang *et al.* [126] pioneered this area and formulated the problem statement. They also presented a number of baselines for comparison. Among them **NLL-KD** and **NLL-OT** are inspired by noisy label learning and utilize KL divergence and optimal transport respectively for refining pseudolabels. **HD-SHOT** and **SD-SHOT** are based on the SHOT [125] model and treat the source model predictions as hard labels and soft labels, respectively. In addition to these baselines, we compare CABB against state-of-the-art black-box DA models **DINE** [126] and **BETA** [219]. We further compare against a number of standard DA methods, such as **DANN** [49], **ALDA** [21], **GVB-GD** [31], **SRDC** [194], **SHOT** [125], **A^2-Net** [214], **SFDA-DE** [36] etc.

Table 5.7: Performance evaluation of CABB with various values for the hyperparameter α for VisDA-C dataset.

α	plane	bcycl	bus	car	horse	knife	mcycle	person	plant	sktbrd	train	truck	Per-class
$2e^{-3}$	97.6	89.1	80.0	57.9	96.7	79.8	85.5	81.1	93.4	91.0	89.4	49.5	82.6
$2e^{-4}$	95.1	87.0	82.6	71.5	94.5	89.7	87.5	81.5	93.8	92.4	87.3	55.5	84.9
$2e^{-5}$	98.0	88.4	83.8	51.0	96.9	75.2	84.6	80.1	93.8	92.9	88.4	48.0	81.8

In Figure 5.4, we present the accuracy of the clean sample set after clean-noisy sample separation for the first epoch after distillation from the source teacher model to the target student model. We can see that our choice of low JSD separation criterion in CABB consistently outperforms the low CE loss criterion used in BETA by 1-7% across all 12 source-target domain pairs for Office-Home dataset.

The classification accuracies after adaptation across the 6 domain pairs for Office-31 dataset are shown in Table 5.2. CABB outperforms BETA and DINE on average by 0.5% and 2.3%, respectively. While CABB beats DINE across all the domain pairs, it only underperforms BETA for **Webcam-Amazon** adaptation by 0.5%. Overall, CABB is on-par with *white-box source-free* model SHOT and *non-source-free* model ALDA.

The results for Office-Home dataset are presented in Table 5.3. CABB outperforms BETA and DINE by 0.7% and 3.1%, respectively. Moreover, CABB outperforms several standard *non-source-free* DA methods such as SRDC and FixBi, and is either better than, or on par with existing state-of-the-art *white-box source-free* DA models like HCL, A^2Net, and SFDA-DE.

A comparative evaluation of CABB against other state-of-the-art DA methods and BBDA baselines on the VisDA-C dataset is shown in Table 5.4. CABB surpasses both DINE and BETA by 9.3% and 2.3%, respectively in terms of mean-per-class accuracy. CABB beats BETA in the most challenging catergory

truck by 10.4%. CABB also outperforms *white-box source-free* models SHOT and A^2Net comfortably.

5.5.2 Ablation study

A detailed ablation study on the efficacy of our curriculum adaptation method is given in Table 5.5. The impact of curriculum on the noisy set loss $\mathcal{L}_{tn}$ and entropy loss $\mathcal{L}_{ent}$ is shown, as curriculum is applied to these two components. In this table, in the absence of curriculum adaptation, γ_n is set to 0.5. In rows two and five, $\mathcal{L}_{tn}$ is set to 0. In addition, we further compare our active-passive loss against standard cross-entropy (CE) loss for the unreliable sample subset, and present results with CE loss for noisy samples in rows three and six (denoted by $*$ for $\mathcal{L}_{tn}$).

The results clearly indicate the benefit of a guided adaptation framework that progressively learns from the clean samples first and the noisy samples later. We see in the first four rows in Table 5.5 that without curriculum guidance, adaptation performance suffers significantly. In the absence of curriculum guidance, we see that leaving out learning from the noisy samples during the adaptation process is better than adapting to the noisy samples with active-passive $\mathcal{L}_{tn}$ loss, and further enforcing the wrong predictions with $\mathcal{L}_{ent}$ loss.

The drawback of blindly adapting to noisy samples becomes evident in the second and fourth rows, particularly in the most challenging *truck* class. By adapting to unrefined noisy samples from the beginning, the model performance drastically deteriorates and accuracy on *truck* can fall to as low as $\sim 1\%$. Adaptation with CE loss on noisy samples is however robust, even without curriculum, particularly due to significantly higher accuracy for *truck* class.

The results in the 5th through 8th rows in Table 5.5 show the necessity for curriculum guidance during adaptation. In the presence of curriculum learning, CABB outperforms existing state-of-the-art BBDA methods. Curriculum guidance progressively refines the noisy sample pseudolabels. While enforcing the refined predictions by minimizing the $\mathcal{L}_{ent}$ loss produces improved results, learning from the noisy pseudolabels by minimizing the $\mathcal{L}_{tn}$ loss significantly boosts the model performance. Minimizing losses $\mathcal{L}_{tn}$ and $\mathcal{L}_{ent}$ on the refined pseudolabels together produce the strongest results. Comparing between rows 6 and 8, it can be seen that adaptation with active-passive loss on noisy samples outperforms that with standard CE loss on noisy samples by $\sim 1\%$.

Table 5.6 illustrates the impact of dual branch co-teaching and the pseudolabeling process on our CABB framework. *Model Predictions* refer to pseudolabeling based on the model prediction probabilities on the non-augmented images. *Clustering* refers to the self-supervised pseudolabeling technique used in

SHOT [125], based on DeepCluster [14]. To get soft-pseudolabels, we convert the cosine distances into class probabilities using *SoftMin* ($\frac{exp(-x_i)}{\sum_j exp(-x_j)}$). The results in Table 5.6 clearly demonstrate that our ensemble-based pseudolabeling is much more robust compared to other pseudolabeling methods. *Model Predictions* produces high rate of wrong pseudolabels due to domain gap on non-augmented samples, while clustering-based pseudolabeling tends to preserve feature clusters but disregard class boundaries. Ensemble-based pseudolabeling preserves class boundaries and produces robust pseudolabels due to averaging over predictions generated by both the branches on a number of augmented views.

We can further see that co-training with two branches helps to reduce error accumulation and outperforms training with a single branch. We also assess the impact of the value of hyperparameter α (curriculum factor) on CABB in Table 5.7. For VisDA-C, $\alpha = 2e^{-}4$ produces the best results. The hyperparameter α determines the balance between the clean set loss and the noisy set loss, and is dependent on the size of the dataset and domain gap between the source and target domains. Higher domain gaps will necessitate lower values of α to delay learning from the noisy set. Smaller datasets would require higher α to avoid overfitting on the clean set.

5.6 Conclusion

In this paper we present a curriculum-guided self-training based domain adaptation method called CABB to adapt a black-box source model/predictor to the target domain. Without access to the source data or the source model parameters during adaptation, we draw inspiration from noisy label learning algorithms. We employ a co-training scheme and propose to use Jensen-Shannon distance or JSD as the criterion to filter clean and reliable samples from noisy and unreliable samples. JSD calculated between the source model predicted pseudolabels and target model predictions is modelled using a mixture of Gaussian distributions. The samples with high probability of lying on the distribution with the lower mean JSD are taken as clean samples, and the target model is trained under a curriculum schedule first on the clean samples and progressively on the noisy samples. The dual-branch design of CABB also allows robust ensemble-based pseudolabeling. CABB consistently outperforms existing black-box domain adaptation models on three popular domain adaptation benchmarks, and is on par with other white-box source free models.

Chapter 6

Unknown Sample Discovery for Source Free Open Set Domain Adaptation

This chapter is based on a paper titled "Unknown Sample Discovery for Source Free Open Set Domain Adaptation" [83], which was accepted to the 1st Workshop on Test-Time Adaptation: Model, Adapt Thyself! (MAT) at The IEEE / CVF Computer Vision and Pattern Recognition Conference (CVPR) 2024. Open Set Domain Adaptation (OSDA) aims to adapt a model trained on a source domain to a target domain that undergoes distribution shift and contains samples from novel classes outside the source domain. Source-free OSDA (SF-OSDA) techniques eliminate the need to access source domain samples, but current SF-OSDA methods utilize only the known classes in the target domain for adaptation, and require access to the entire target domain even during inference after adaptation, to make the distinction between known and unknown samples. In this paper, we introduce **U**nknown **S**ample **D**iscovery (USD) as an SF-OSDA method that utilizes a temporally ensembled teacher model to conduct known-unknown target sample separation and adapts the student model to the target domain over all classes using co-training and temporal consistency between the teacher and the student. USD promotes Jensen-Shannon distance (JSD) as an effective measure for known-unknown sample separation. Our teacher-student framework significantly reduces error accumulation resulting from imperfect known-unknown sample separation, while curriculum guidance helps to reliably learn the distinction between target known and target unknown subspaces. USD appends the target model with an unknown class node, thus readily classifying a target sample into any of the known or unknown classes in subsequent post-adaptation inference stages. Empirical results show that USD is superior to existing SF-OSDA methods and is competitive with current OSDA models that utilize both source and target domains during adaptation.

6.1 Introduction

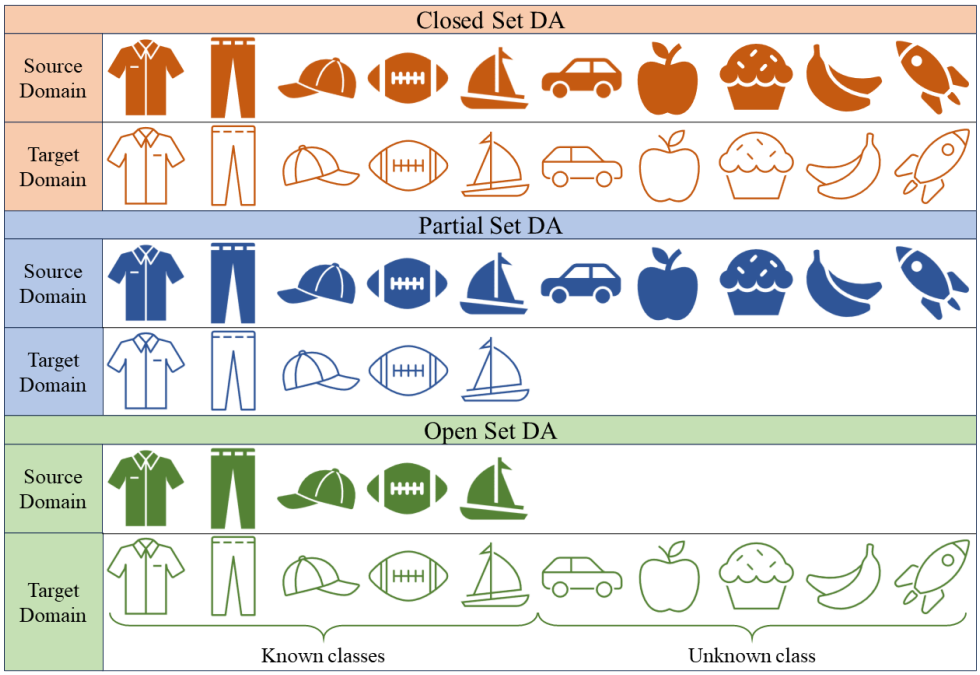


Figure 6.1: Different domain adaptation settings depending on the classes present in the source and target domains. For open-set DA, the classes novel to the target domain are grouped into a single unknown class during adaptation.

While the vast majority of existing UDA literature deals with closed-set domain adaptation, where the target domain and source domain share the same classes, a more realistic scenario is open-set domain adaptation (OSDA) [130, 183] where the target domain contains samples belonging to novel classes that are absent in the source domain (Figure 6.1). In the OSDA setting, closed-set UDA solutions would enforce alignment of the source and target feature spaces under the unknown category mismatch, leading to negative transfer [42] and deteriorating performance. The majority of the existing OSDA methods [130, 183] utilize domain adversarial learning techniques to align the source domain with only the known classes in the target domain, leaving out the target-unknown classes. Such methods fail to properly learn the features for the unknown classes, and hence no clear decision boundary between the known classes and the unknown class in the target domain is realized. Some universal domain adaptation methods, i.e. UDA methods designed to work in both closed and open-set settings [114, 181], have attempted to conduct self-supervised learning (SSL) to discover latent target domain features without explicit distribution matching. However, such methods fail under large domain gaps. More recently, [88] proposed a three-way domain adversarial feature space alignment between the source domain and the known and the unknown target subdomains, thus segregating

the known and unknown classes in the target domain.

In this work, we introduce **U**nknown **S**ample **D**iscovery (**USD**) as a source-free OSDA (SF-OSDA) method that utilizes an ensemble-based pseudolabeling strategy for the target data, and generates known and unknown target subsets based on Jensen-Shannon distance (JSD) between the pseudolabels and the predictions from a teacher model. USD uses two-component Gaussian Mixture Model (GMM) to model the target domain JSD, where the distribution with the lower mean JSD is considered to be of the known class samples and that with the greater mean JSD is taken as that consisting of unknown class samples. The known-unknown target subsets are used to adapt the student model. The student model is updated with gradient descent, while the teacher model is updated by exponential moving averages (EMA) of the teacher and student models. The teacher-student framework in USD helps to mitigate error accumulation induced from any possibly faulty known-unknown sample separation.

USD introduces an unknown class output node in the target model. The adapted target model infers new target samples in one one of the known classes or the unknown class, without operating on the entire target dataset first to identify known and unknown samples. The main contributions of this work are as follows.

- We introduce USD as an SF-OSDA model that co-trains a dual-branch teacher-student framework to split the target domain into known and unknown class subsets.
- USD proposes the Jensen-Shannon distance between the target pseudolabels and teacher model predictions as an effective criterion for separating target samples in known and unknown classes.
- Co-training in USD, aided by weak-strong consistency between the teacher and student outputs, significantly mitigates error accumulation resulting from imperfect known-unknown separation, and sustains the adaptation performance.
- USD generates reliable pseudolabels from the student model outputs on an ensemble of weak and strong target data augmentations.
- USD utilizes curriculum adaptation to progressively learn the known class feature space first, and the unknown class feature space later, thus enabling robust alignment of the entire target space with the source domain.
- Extensive experiments on 3 popular UDA benchmarks demonstrate the superiority of USD over existing SF-OSDA methods.

6.2 Related Work

6.2.1 Unsupervised domain adaptation

Domain gap originates from the distribution shift between the source domain where a deep network model is trained, and the target domain where the model is deployed [200]. This domain gap may be reduced by minimizing the maximum mean discrepancy (MMD) [138, 202], or the central moment discrepancy (CMD) [229] between the distributions in the source and target domains. Deep CORAL [192] mitigated domain shift by matching second-order distribution statistics. [49] introduced the Gradient Reversal Layer (GRL) and made use of a domain discriminator to adversarially align the source and target distributions in a common feature space using a common feature encoder. The Adversarial Discriminative Domain Adaptation (ADDA) [201] method decoupled the feature extraction process by learning two separate feature encoders for the two domains and aligned them adversarially to perform classification with a common classifier.

Generative adversarial networks (GANs) have been utilized to produce images in an intermediate domain between the source and target to facilitate easier and smoother adaptation [69]. Domain-wise global adversarial alignment in the absence of target annotations may lead to loss of class discrimination in the target embeddings. To align the domain-wise and class-wise distributions across the source and target data while maintaining target class feature discrimination, [120] simultaneously solved two complementary domain-specific and class-specific minimax objectives. The non-adversarial alignment approach in [163] imposed a consistency constraint between the labeled source prototypes and the pseudo-labeled target prototypes in the feature space.

6.2.2 Source free domain adaptation

UDA methods that adversarially align the embedding space [49, 69, 201] or minimize the source-target domain divergence [138,202,229] require access to both the source and target data during adaptation, rendering them unusable in situations where the source data is private or restricted. A semi-supervised UDA method involving a few source representatives or prototypes instead of the full source data was proposed in [27]. Distant supervision for SFDA [124] iteratively assigned pseudo-labels to the target data and used them to learn a domain invariant feature space and obtain the target class centroids. Liang *et al.* [125] introduced SHOT which adapts the source-pretrained feature encoder to the target domain via self-training with information maximization [103, 187] and self-supervised clustering for pseudolabeling, while transferring the source hypothesis (classifier model) to the target. Ahmed *et al.* [3] proposed to calculate more than one class prototype for each class in the target domain, as a single prototype may fail to fully characterize the class

in the latent space. To further refine the pseudolabels, [221] proposed to enforce neighborhood consistency regularization among the target samples. To generate compact target clusters, [222] considered minimizing the distance among K-nearest neighbors for each target sample and dispersing the rest by retrieving target features stored in a memory bank. Noisy label learning (NLL) is also another promising avenue for filtering pseudo-labels, as domain shift may cause significant number of incorrect pseudolabels. Kim *et al.* [100] proposed to calculate multiple prototypes for each class and recognize a sample as reliable when the Hausdorff distance from the sample to the its most similar class prototypes is smaller than that between the sample and the second-most similar class prototypes. Chu *et al.* [28] built on Arpit's *et al.* [5] claim that weak models are less prone to memorization, and proposed to use an additional untrained model to identify incorrect pseudolabels in the target domain. SHOT++ [127] utilized MixMatch [9] between high confidence samples and low confidence samples to increase the fidelity of the low confidence ones. CABB [82] employed NLL techniques to identify noisy target samples in black-box SFDA, and made use of curriculum guidance for progressive adaptation.

6.2.3 Open set domain adaptation

In addition to aligning the source and target subspaces, a critical step in OSDA is detecting target samples from novel or unknown categories that are absent in the source domain. [87] applied a simple class-wise confidence threshold to reject those samples with lower confidence as unknown. [183] adversarially aligned the source domain and known target subdomain, where the unknown target samples were identified based on a preset threshold. Alignment for only the known classes however results in subpar performance in identifying the unknown samples. The adversarial alignment objective was modified in [130] with an instance weighting procedure, where higher weights were given to known target samples and lower weight to unknown samples. This somewhat smoothened the known-unknown distinction, but lower weights produced less contributions in the objective loss from the unknown samples, leading to suboptimal performance. A three-way domain adversarial alignment between source, known target, and unknown target in the feature space was proposed in [88] such that the source and known target are aligned while the target-unknown gets segregated. [125] and [222] are SF-UDA methods that also conduct SF-OSDA by separating the known and unknown samples based on clustering the sample entropies into two clusters, and taking the cluster with lower mean entropy as the known subset.

6.3 Method

For unsupervised OSDA, we have n_s labeled samples $\{x_s^i, y_s^i\}_{i=1}^{n_s} \in \mathcal{X}_s, \mathcal{Y}_s$ belonging to the source domain $\mathcal{D}_s$, and n_t unlabeled samples $\{x_t^i\}_{i=1}^{n_t} \in \mathcal{X}_t$ belonging to the target domain $\mathcal{D}_t$. The task of SF-OSDA is to take the source model $f_s(\theta_s) : \mathcal{X}_s \rightarrow \mathcal{Y}_s$ with model parameters θ_s trained on the C_s-multiclass source data $\{x_s^i, y_s^i\}_{i=1}^{n_s} \in \mathcal{X}_s, \mathcal{Y}_s$, and adapt it to $f_t(\theta_t) : \mathcal{X}_t \rightarrow \mathcal{Y}_t$ with model parameters θ_t that can map the $\{x_t^i\}_{i=1}^{n_t} \in \mathcal{X}_t$ to the C_t classes, where $C_t = C_s + 1$. The additional class in the target domain is a catch-all class for all samples in the target domain that do not belong to any of the classes in the source domain.

We follow [125] for Source model training follows [125] to ensure fair comparison with other source-free UDA models. The source model is trained by minimizing the standard cross entropy loss with label smoothing [156] as follows.

$$\mathcal{L}_s(f_s; \mathcal{X}_s, \mathcal{Y}_s) = -\mathbb{E}_{x_s \in \mathcal{X}_s, y_s \in \mathcal{Y}_s} \sum_{k=1}^{C_s} q_k^{ls} \log(\sigma_k(f_s(x_s))) \tag{6.1}$$

where $\sigma_k(a) = \frac{exp(a_k)}{\sum_i exp(a_i)}$ is the k-th element in its softmax output of a C_s-dimensional vector a, and q^{ls} is the one-hot encoded and smoothed C_s-dimensional vector for sample label y_s^i, such that $q_k^{ls} = (1-\alpha)q_k + \alpha/C_s$, where q_k is 1 for the correct class and 0 for all other classes, and α is the smoothing factor set at 0.1.

The source model f_s consists of a feature extractor $g_s : \mathcal{X}_s \rightarrow \mathbb{R}^d$ and a C_s-class classifier $h_s : \mathbb{R}^d \rightarrow \mathbb{R}^{C_s}$, such that $f_s(x) = h_s(g_s(x))$. USD consists of a student target model $f_t^S(\theta_t^S)$ and a teacher target model $f_t^T(\theta_t^T)$. The feature extractors g_t^S and g_t^T, in the student and teacher networks respectively, are initialized with the source model feature extractor, i.e., $g_t^S = g_t^T = g_s$. To account for the novel class samples in the target domain, the source classifier h_s is expanded in the student and teacher models to include an additional trainable output node for the unknown class. The known class nodes in the target classifiers h_t^S and h_t^T, for the student and teacher respectively, are initialized with h_s, and remain frozen during adaptation. The unknown class nodes in h_t^S, h_t^T and the feature extractors g_t^S, g_t^T are adapted using only the unlabeled target samples.

6.3.1 Known-unknown sample separation

The first step for target adaptation is to reliably separate the known class samples and the novel class samples in the target data. This step is visually depicted in Figure 6.2. In order to generate pseudolabels $\hat{y}_t$, the target data undergoes $M = 6$ number of weak and strong augmentations (1 weak and 5 strong) based on *AutoAugment* [30] policy for ImageNet. The softmax output over C_s classes for each augmented view x_t^{iM}

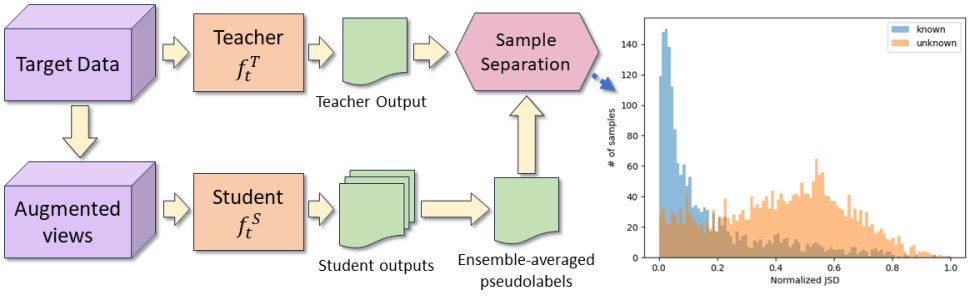


Figure 6.2: Pseudolabel generation for the target samples and known-unknown sample separation based on JSD

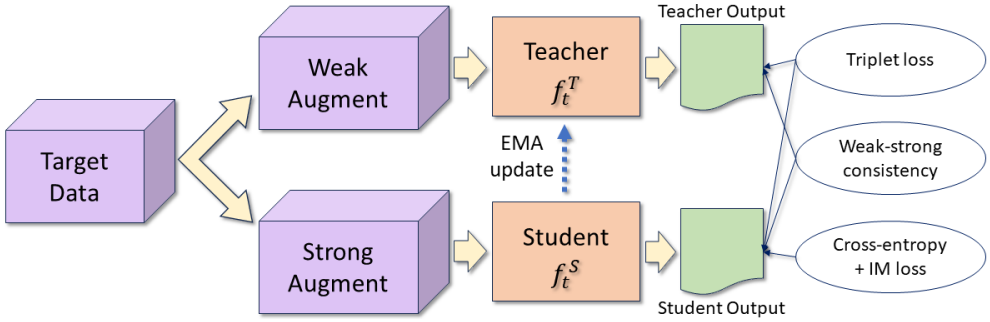


Figure 6.3: Adaptation process for USD using co-training. The student model receives pseudolabels for the target samples (see Figure 6.2) and is optimized using a combination of triplet, weak-strong consistency, information maximization (IM) and cross-entropy losses. The teacher model is updated via exponential moving averages (EMA) at the end of each epoch.

is taken from the student model f_t^S, and then averaged over the augmentations, as follows.

$$\hat{y}_t^i = \arg\max \frac{1}{M} \sum_{1}^{M} f_t^S(x_t^{iM}) \tag{6.2}$$

The index corresponding to the maximum averaged softmax output is taken as the hard pseudolabel $\hat{y}_t^i$ for each target sample x_t^i. These pseudolabels are however only over the C_s known classes, and therefore the samples need to be split into known class subset $\mathcal{X}_t^K$ and unknown class subset $\mathcal{X}_t^U$. Existing SF-OSDA methods [125, 222] identify unknown class samples by utilizing the output entropy of the target data. Entropies for all samples are calculated at the beginning of each epoch and then normalized in the range of $[0, 1]$ by dividing the each sample entropy by $\log C_s$. The normalized entropies are then clustered by two-class k-means clustering. The cluster with the higher mean entropy or uncertainty is considered to be the one containing unknown samples, while the other cluster with lower mean entropy is taken as containing known class samples.

Sample separation is a critical component for noisy label learning (NLL) algorithms where clean and noisy samples are separated for robust supervised training of a model. Traditionally, NLL calculates the cross-entropy loss on the whole dataset and then uses low cross-entropy loss as the criterion to identify clean samples [4, 115, 119]. In USD, we conduct known-unknown sample separation for SF-OSDA based on JSD between the network outputs and their corresponding pseudolabels, which is calculated as follows.

$$JSD(\hat{y}_t^i, p_t^i) = \frac{1}{2} KL\left(\hat{y}_t^i, \frac{\hat{y}_t^i + p_t^i}{2}\right) + \frac{1}{2} KL\left(p_t^i, \frac{p_t^i + \hat{y}_t^i}{2}\right) \tag{6.3}$$

where, $KL(a, b)$ is the Kullback-Leibler divergence between a and b, and $p_t^i = \sigma(f_t^T(x_t^i))$ is the output softmax probability for target sample x_t^i from the target teacher model f_t^T.

We consider the unknown class samples in the target domain as noisy samples when predictions are made over only the known C_s classes. In comparison to entropy or cross-entropy loss, JSD is symmetric by design and ranges between 0 and 1. As shown in Figure 6.2, when plotted against the number of samples, JSD produces a bimodal histogram. It is possible to set a threshold on the JSD histogram and split the target samples into known and unknown subsets based on the threshold value. However, such a threshold would depend on the location of the modes in the histogram, and therefore a fixed threshold cannot be applied across all source-target domain pairs. We therefore model the JSD distribution with two-component Gaussian Mixture Model (GMM) with equal priors, resulting in probabilities $[w_{t_L}^i, w_{t_H}^i]$ for each target sample x_t^i to belong to either of the two components. We consider the samples with higher probability of belonging to the distribution with the lower-mean Gaussian component as samples from one of the known classes, and conversely consider those samples with higher probability of belonging to the higher-mean Gaussian component as coming from the unknown target class.

Practically, we take the probability $w_{t_L}^i$ of belonging to the lower-mean GMM component for each target sample x_t^i, and in order to be conservative in our sample splitting, we set a lower-bound/threshold δ_t on $w_{t_L}^i$ to select the known sample subset $\mathcal{X}_t^K$. The remaining target samples are included in the unknown subset $\mathcal{X}_t^U$. The pseudolabels $\hat{y}_t^i$ are updated accordingly, where the known subset retain their earlier assigned pseudolabel from among the C_s classes, and the unknown subset of target samples get the new unknown class pseudolabel $|C_t|$. It has to be noted that during adaptation, the teacher network conducts the known-unknown sample separation at the beginning of each epoch, and the student network is adapted over the C_t classes with the target data.

6.3.2 Teacher-student co-training and regularization

USD simultaneously adapts the student and teacher target models, such that the student model parameters θ_t^S are updated based on the minibatch gradient descent, and the teacher network parameters θ_t^T are updated as temporally ensembled version of the student network [196] at the end of each epoch as follows.

$$\theta_{t_N}^T = m\theta_{t_{N-1}}^T + (1-m)\theta_{t_N}^S \tag{6.4}$$

where, m is the momentum parameter for weight ensembling, and $N = 2, 3, .., E$ is the epoch number. Such co-training and cross-network sample splitting by the teacher for the student work to lessen error accumulation from imperfect known-unknown sample separation and stabilizes the adaptation process. As a means of training regularization, USD further maintains weak-strong temporal consistency between the teacher network outputs and the student network outputs by minimizing the following consistency loss.

$$\mathcal{L}_t^{con}(f_t^S, f_t^T; \mathcal{X}_t) = KL\left(p_t^{iS}, p_t^{iT}\right) = \sum_{k=1}^{C_t} p_t^{iT} \log\left(\frac{p_t^{iT}}{p_t^{iS}}\right) \tag{6.5}$$

where, $p_t^{iS} = \sigma(f_t^S(x_t^{iS}))$ is the softmax output from the student on an strongly augmented target sample x_t^{iS}, and $p_t^{iT} = \sigma(f_t^T(x_t^{iW}))$ is the softmax output from the teacher on the weakly augmented version x_t^{iW} of the same target instance. The strong and weak augmentations are done following the *AutoAugment* [30] ImageNet policy.

Contrastive learning is a viable method to learn discriminative representations by minimizing the distance between an anchor and a corresponding positive instance and by maximizing the distance between the anchor and a corresponding negative instance. In a source-free setting, it is exceedingly difficult to identify positive and negative instances for a target anchor. USD deftly utilizes the teacher-student framework and weak-strong augmentations, and employs a triplet loss [185] to effectively learn the decision boundary between known and unknown classes. The output $z_T^{ia} = [f_t^T(x_t^{iW})]^a$ of the teacher model on an weakly augmented known class sample is taken as the anchor, and the corresponding output $z_S^{i+} = [f_t^S(x_t^{iS})]^+$ on the strongly augmented version of the same sample from the student model is taken as the positive instance. The negative instance is the student model output $z_S^{i-} = [f_t^S(x_t^{iS})]^-$ on a randomly chosen unknown class sample. Cosine distance is taken as the distance metric, and is calculated as follows.

$$\mathbf{D}(z_1, z_2) = 1 - \frac{z_1 . z_2}{||z_1||_2 ||z_2||_2} \tag{6.6}$$

where z_1 and z_2 are any two network outputs. Triplet loss is in turn calculated as follows.

$$\mathcal{L}_t^{trip}(f_t^S, f_t^T; \mathcal{X}_t) = \max(\mathbf{D}(z_T^{ia}, z_S^{i+}) - \mathbf{D}(z_T^{ia}, z_S^{i-}) + 1, 0) \tag{6.7}$$

In addition, the student network is trained with the instance-weighted standard cross-entropy loss with label smoothing [156], as follows.

$$\mathcal{L}_t^{ce}(f_t^S; \mathcal{X}_t) = -\mathbb{E}_{x_t^i \in \mathcal{X}_t} \omega^i \sum_{k=1}^{C_t} \hat{y}_{t_k}^i log(\sigma_k(f_t^S(x_t^{iS}))) \tag{6.8}$$

The instance weights ω^i are the probability $w_{t_L}^i$ for known target samples $x_i^t \in \mathcal{X}_t^K$ of belonging to the lower-mean JSD component, and probability $w_{t_H}^i$ for unknown target samples $x_i^t \in \mathcal{X}_t^U$ of belonging to the higher-mean JSD component, during the known-unknown sample separation. In order to promote adaptation to the known samples first and to progressively learn the unknown class feature space, USD utilizes cross-entropy loss under curriculum guidance, dictated by the curriculum factor γ_r as follows.

$$\mathcal{L}_t^{ce}(f_t^S; \mathcal{X}_t^K, \mathcal{X}_t^U) = \gamma_n \mathcal{L}_{tK}^{ce}(f_t^S; \mathcal{X}_t^K) + (1 - \gamma_r)\mathcal{L}_{tU}^{ce}(f_t^S; \mathcal{X}_t^U) \tag{6.9}$$

where $\gamma_r = \max(0.5, \gamma_{r-1}(1 - \beta\epsilon^{-\mathcal{L}_{tK_r}^{ce}/\mathcal{L}_{tK_{r-1}}^{ce}}))$ such that, β is a hyperparameter and r is the current iteration number. The ratio $\mathcal{L}_{tK_r}^{ce}/\mathcal{L}_{tK_{r-1}}^{ce}$ dictates the degree by which the curriculum factor decreases from the earlier $(r-1)$-th iteration to the current r-th iteration. When loss $\mathcal{L}_{tK}^{ce}$ on the known sample subset increases, γ marginally decreases to accommodate further adaptation on the known samples in the subsequent iterations. But if $\mathcal{L}_{tK}^{ce}$ decreases by a large margin, γ decreases accordingly to progressively adapt to the unknown samples in the following iterations. Curriculum guidance balances the adaptation of the target model to the known and unknown subsets.

To encourage individually precise and globally diverse predictions, USD further minimizes the information maximization (IM) [125] loss as formulated in [198], [197].

$$\mathcal{L}_t^{ent}(f_t^S; \mathcal{X}_t^K) = -\mathbb{E}_{x_t^i \in \mathcal{X}_t^K} \sum_{k=1}^{C_t} \sigma_k(f_t^S(x_t^{iS}))log(\sigma_k(f_t^S(x_t^{iS}))) \tag{6.10}$$

$$\mathcal{L}_t^{eqdiv}(f_t^S; \mathcal{X}_t^K) = \sum_{k=1}^{C_t} p_t^{iS} log\left(\frac{p_t^{iS}}{\overline{p_t^{iS}}}\right) \tag{6.11}$$

$$\mathcal{L}_t^{IM}(f_t^S; \mathcal{X}_t^K) = \mathcal{L}_t^{ent}(f_t^S; \mathcal{X}_t^K) + \mathcal{L}_t^{eqdiv}(f_t^S; \mathcal{X}_t^K) \tag{6.12}$$

where $\overline{p_t^{iS}} = \mathbb{E}_{x_t^i \in \mathcal{X}_t^K}[\sigma(f_t^S(x_t^{iS}))]$ is the mean softmax output vector over known target samples in a mini-batch. The overall objective function is therefore,

$$\mathcal{L}_t^{tot} = \mathcal{L}_t^{ce} + \mathcal{L}_t^{IM} + \zeta_1 \mathcal{L}_t^{trip} + \zeta_2 \mathcal{L}_t^{con} \tag{6.13}$$

where ζ_1 and ζ_2 are two hyperparameters.

A brief demonstration of the USD domain adaptation pipeline is presented in Algorithm 4.

Algorithm 4: Pseudocode for USD

Input: Source trained model f_s and n_t unlabled target data samples $x_t^i \in \mathcal{X}_t$
Output: Target adapted student model f_t^S
Initialization: Teacher target model f_t^T and student target model f_t^S, are both initialized with parameters θ_s from f_s

for *epoch* = 1 *to* E **do**
  Conduct $M = 6$ weak-strong augmentations and assign ensemble averaged pseudolabels $\hat{y}_t^i$ using eq. (6.2)
  Conduct known ($\mathcal{X}_t^K$) - unknown ($\mathcal{X}_t^U$) target sample separation using JSD between $\hat{y}_t^i$ and teacher softmax output $p_t^i = \sigma(f_t^T(x_t^i))$
  for i = 1 *to* n_t **do**
    Optimize, for each minibatch, student model f_t^S with loss $\mathcal{L}_{tot}$ using eq. (6.13) and get new student model parameters θ_t^S
  end
  Update teacher model f_t^T using new student model weights θ_t^S and current teacher model weights θ_t^T using eq. (6.4)
end

6.4 Experimental Setup

6.4.1 Datasets

We evaluate USD on three popular domain adaptation benchmarks: Office-31 [180], Office-Home [204], and VisDA-C [167]. These datasets have been previously described in Chapter 4. We follow [183] for splitting the data into shared (known) and target-private (unknown) classes for all three datasets evaluated.

Table 6.1: Evaluation of USD on Office-31 dataset. * are results computed for the methods using publicly released code.

Method	SF	Office																				
		A → D			A → W			D → A			D → W			W → A			W → D			Avg.		
		OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS
DANN [49]	✗	90.8	59.2	71.5	87.4	55.7	68.1	72.9	74.5	73.7	99.3	77.0	86.7	72.1	73.1	72.6	100.0	70.2	82.5	87.1	68.3	75.9
CDAN [139]	✗	92.2	52.4	66.8	90.3	50.7	64.9	74.9	70.6	72.7	99.6	73.2	84.3	72.8	69.3	71.0	100.0	67.3	80.5	88.3	63.9	73.4
STA [130]	✗	91.0	63.9	75.0	86.7	67.6	75.9	83.1	65.9	73.2	94.1	55.5	69.8	66.2	68.0	66.1	84.9	67.8	75.2	84.3	64.8	72.5
OSBP [183]	✗	90.5	75.5	82.4	86.8	79.2	82.7	76.1	72.3	75.1	97.7	96.7	97.2	73.0	74.4	73.7	99.1	84.2	91.1	87.2	80.4	83.7
PGL [143]	✗	82.1	65.4	72.8	82.7	67.9	74.6	80.6	61.2	69.5	87.5	68.1	76.5	80.8	61.8	70.1	82.8	64.0	72.2	82.7	64.7	72.6
OSLPP [209]	✗	92.6	90.4	91.5	89.5	88.4	89.0	82.1	76.6	79.3	96.9	88.0	92.3	78.9	78.5	78.7	95.8	91.5	93.6	89.3	85.6	87.4
UADAL [88]	✗	85.1	87.0	86.0	84.3	94.5	89.1	73.3	87.3	79.7	99.3	96.3	97.8	67.4	88.4	76.5	99.5	99.4	99.5	84.8	92.1	88.1
SHOT* [125]	✓	94.0	46.3	62.0	95.6	42.3	58.7	83.3	39.1	53.3	100.0	75.7	86.1	82.7	46.6	59.6	100.0	69.7	82.1	92.6	53.3	67.0
AaD* [222]	✓	73.0	84.6	78.3	63.5	89.5	74.3	63.6	88.9	74.2	78.0	98.5	87.0	61.9	88.9	73.0	94.6	96.8	95.7	72.4	91.2	80.4
USD (Ours)	✓	90.7	73.4	81.2	82.8	72.7	77.9	65.7	84.4	73.9	97.9	96.6	97.3	64.6	86.7	74.0	98.0	92.6	95.2	83.3	84.4	83.3

Table 6.2: Evaluation of USD on Office-Home and VisDA-C datasets. * are results computed for the methods using publicly released code.

Method	SF	Office-Home																				
		A → C			A → P			A → R			C → A			C → P			C → R			P → A		
		OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS
DANN [49]	✗	37.1	82.7	51.2	60.0	71.3	65.2	75.1	67.3	71.0	43.8	84.3	57.6	50.1	77.6	60.9	61.1	73.5	66.7	42.4	83.9	56.3
CDAN [139]	✗	39.7	78.9	52.9	61.7	68.8	65.1	75.2	66.7	70.7	44.9	82.8	58.2	51.6	76.8	61.7	61.5	73.7	67.1	45.8	81.2	58.6
STA [130]	✗	46.0	72.3	55.8	68.0	48.4	54.0	78.6	60.4	68.3	51.4	65.0	57.4	61.8	59.1	60.4	67.0	66.7	66.8	54.2	72.4	61.9
OSBP [183]	✗	50.2	61.1	55.1	71.8	59.8	65.2	79.3	67.5	72.9	59.4	70.3	64.3	67.0	62.7	64.7	72.0	69.2	70.6	59.1	68.1	63.2
PGL [143]	✗	63.3	19.1	29.3	78.9	32.1	45.6	87.7	40.9	55.8	85.9	5.3	10.0	73.9	24.5	36.8	70.2	33.8	45.6	73.7	34.7	47.2
OSLPP [209]	✗	55.9	67.1	61.0	72.5	73.1	72.8	80.1	69.4	74.3	49.6	79.0	60.9	61.6	73.3	66.9	67.2	73.9	70.4	54.6	76.2	63.6
UADAL [88]	✗	54.9	74.7	63.2	69.1	72.5	70.8	81.3	73.7	77.4	53.5	80.5	64.2	62.1	78.8	69.5	69.1	78.3	73.4	50.5	83.7	63.0
SHOT [125]	✓	67.0	28.0	39.5	81.8	26.3	39.8	87.5	32.1	47.0	66.8	46.2	54.6	77.5	27.2	40.2	80.0	25.9	39.1	66.3	51.1	57.7
AaD [222]	✓	50.7	66.4	57.6	64.6	69.4	66.9	73.1	66.9	69.9	48.2	81.1	60.5	59.5	63.5	61.4	67.4	68.3	67.8	47.3	82.4	60.1
USD (Ours)	✓	53.3	71.5	61.1	65.7	74.9	70	73.3	79.5	76.3	52.2	70.8	60.1	62.4	68.4	65.2	69.3	68.6	68.9	54.3	73.8	62.6

Method	SF	Office-Home																		VisDA-C		
		P → C			P → R			R → A			R → C			R → P			Avg.					
		OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS
DANN [49]	✗	30.1	86.3	44.6	67.7	72.0	69.8	56.8	77.1	65.4	37.1	80.9	50.9	69.6	67.2	68.4	52.6	77.1	60.7	52.1	-	-
CDAN [139]	✗	33.1	82.4	47.2	69.8	69.7	69.7	59.8	73.6	66.0	40.3	75.8	52.7	70.9	64.6	67.6	54.5	74.6	61.4	-	-	-
STA [130]	✗	44.2	67.1	53.2	76.2	64.3	69.5	67.5	66.7	67.1	49.9	61.1	54.5	77.1	55.4	64.5	61.8	63.3	61.1	62.4	82.4	71.0
OSBP [183]	✗	44.5	66.3	53.2	76.2	71,7	73.9	66.1	67.3	66.7	48.0	63.0	54.5	76.3	68.6	72.3	64.1	66.3	64.7	50.9	81.7	62.7
PGL [143]	✗	59.2	38.4	46.6	84.8	27.6	41.6	81.5	6.1	11.4	68.8	0.0	0.0	84.8	38.0	52.5	76.1	25.0	35.2	-	-	-
OSLPP [209]	✗	53.1	67.1	59.3	77.0	71.2	74.0	60.8	75.0	67.2	54.4	64.3	59.0	78.4	70.8	74.4	63.8	71.7	67.0	-	-	-
UADAL [88]	✗	43.4	81.5	56.6	71.6	83.1	76.9	66.7	78.6	72.1	51.1	74.5	60.6	77.4	76.2	76.8	62.6	78.0	68.7	-	-	-
SHOT [125]	✓	59.3	31.0	40.8	85.8	31.6	46.2	73.5	50.6	59.9	65.3	28.9	40.1	84.4	28.2	42.3	74.6	33.9	45.6	57.5*	12.1*	20.1*
AaD [222]	✓	45.4	72.8	55.9	68.4	72.8	70.6	54.5	79.0	64.6	49.0	69.6	57.5	69.7	70.6	70.1	58.2	71.9	63.6	32.0*	62.9*	42.4*
USD (Ours)	✓	47.3	69.6	56.3	70	74.5	72.2	64.6	71.3	67.8	53.8	65.5	59.1	73.3	69.1	71.1	61.6	71.5	65.9	57.8	86.7	69.4

6.4.2 Implementation details

For source training, we follow the protocol from [125, 222] for fair comparison against existing SF-OSDA methods. The basic structure of the teacher and student models also follow that of [125, 222], that is, the feature extractor is a ResNet-50 [63], followed by a fully-connected (FC) bottleneck layer, a batch normalization layer [80], another FC classifier layer, and finally a weight normalization layer [184], respectively. The student target model is trained with an SGD optimizer with momentum of 0.9 and weight decay of 10^{-3}. Due to the difference in the number of samples in each dataset, USD is adapted for 40 epochs on Office and for 2m Office-Home, and for 5 epochs on VisDA-C, at minibatch size of 64 samples in all cases. Similarly, the hyperparameter β in curriculum factor γ_r is set at 0.01 for Office and Office-Home datasets, while β is set at 0.001 for VisDA-C dataset. The threshold δ_t for known-unknown sample separation is set at 0.8, and the momentum parameter m for temporal ensembling is set according to schedule in [216] with a maximum of 0.9995. Further, $\zeta_1 = 0.01$ and ζ_2 is gradually increased to 0.5 from 0.0 following the schedule in [110]. All experiments are done on a NVIDIA A100 GPU.

6.4.3 Evaluation metrics

The mean-per-class accuracy **OS** over all known classes and the unified unknown class for all the target data may be considered as a metric for evaluating OSDA. However, such a metric is dominated by the accuracy on the known classes, as all the unknown samples are lumped into one unknown class [13]. A better metric is therefore to calculate the mean-per-class accuracy **OS*** over only the known classes, and the accuracy **UNK** for the unknown class, and then take the harmonic mean **HOS** of the two for fair evaluation over the known and the unknown classes. Mathematically, the metrics are formulated as follows.

$$OS^* = \frac{1}{|C_s|}\sum_{i=1}^{|C_s|}\frac{|x_t : x_t \in \mathcal{D}_t^i \cap \tilde{y}_t^i = i|}{|x_t : x_t \in \mathcal{D}_t^i|} \tag{6.14}$$

$$UNK = \frac{|x_t : x_t \in \mathcal{D}_t^{|C_t|} \cap \tilde{y}_t^i = |C_t||}{|x_t : x_t \in \mathcal{D}_t^{|C_t|}|} \tag{6.15}$$

$$HOS = \frac{2 \times OS^* \times UNK}{OS^* + UNK} \tag{6.16}$$

Here, $\tilde{y}_t^i = \arg\max(\sigma(f_t^S(x_t^i)))$ is the prediction from the student model f_t^S and $\mathcal{D}_t^i$ is the target domain data belonging to class i. In this work, we report OS*, UNK, and HOS for the evaluated adaptation tasks.

6.5 Results

6.5.1 Overall results

We compare USD to a number of existing UDA methods: closed-set UDA methods (1) DANN [49], (2) CDAN [139], open-set UDA methods (3) STA [130], (4) OSBP [183], (5) PGL [143], (6) OSLPP [209], and (7) UADAL [88]. These methods however are not source-free. We compare USD to open-set versions of SF-UDA methods SHOT [125] and AaD [222]. The open-set results for SHOT and AaD on Office-Home are provided in their respective publications. We generate results for Office-31 and VisDA-C using their publicly released code.

The results on Office-31 over all 6 domain pairs are presented in Table 6.1. USD outperforms SHOT and AaD by $\sim 16\%$ and $\sim 3\%$, respectively in terms of mean HOS. Distinguishing between known and unknown class samples is crucial in OSDA, and USD strikes the best balance among the other SF-OSDA methods. SHOT clearly adapts primarily to the known classes without good adaptation on the unknown samples. AaD overcompensates in identifying unknown samples at the expense of correctly adapting to the known classes. USD performs equally well over both known and unknown classes, leading to higher HOS. USD also outperforms non-source-free methods STA and PGL, while being comparable to OSBP.

A comparative evaluation for USD against existing UDA methods on Office-Home is given in Table 6.2. USD outperforms SHOT and AaD by $\sim 20\%$ and $\sim 2\%$, respectively in terms of the average HOS over the 12 domain pairs. Similar to Office-31, SHOT adapts better to the known classes, but fails to competently identify unknown samples, while AaD performs worse on the known classes and better on the unknown samples. USD is more balanced across the known and unknown classes and also outperforms non-SF OSDA methods STA, OSBP and PGL.

Results on VisDA-C are given in the bottom right section in Table 6.2. SHOT severely suffers from negative transfer in the unknown class, while AaD fails to learn the target-known feature space. USD greatly outperforms SHOT and AaD, as well as the non-SF method OSBP, while being comparable to STA in terms of mean HOS.

6.5.2 Ablation study

A detailed ablation study was performed on the known-unknown sample selection criterion and on the modeling of the criterion distribution. The results of the ablation study on both Office-31 and VisDA-C

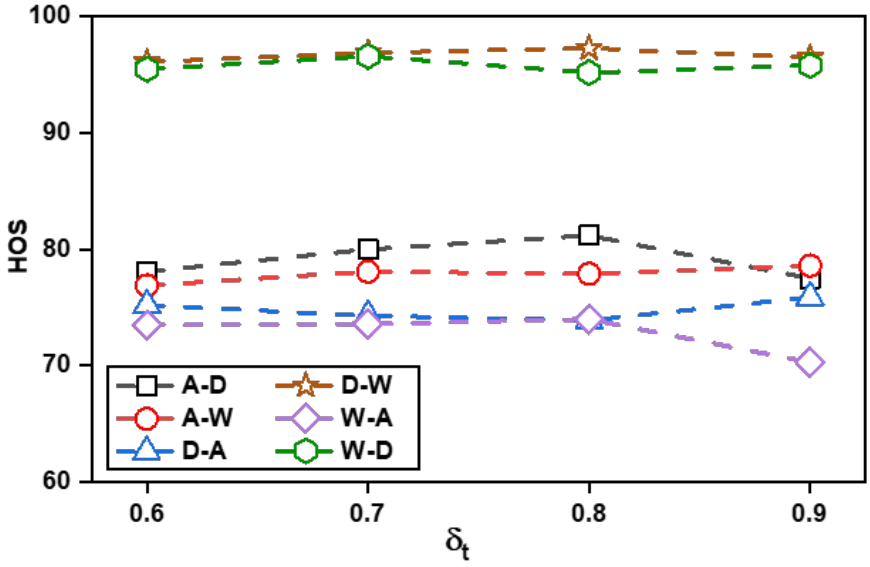


Figure 6.4: Impact of JSD threshold δ_t on HOS for Office dataset.

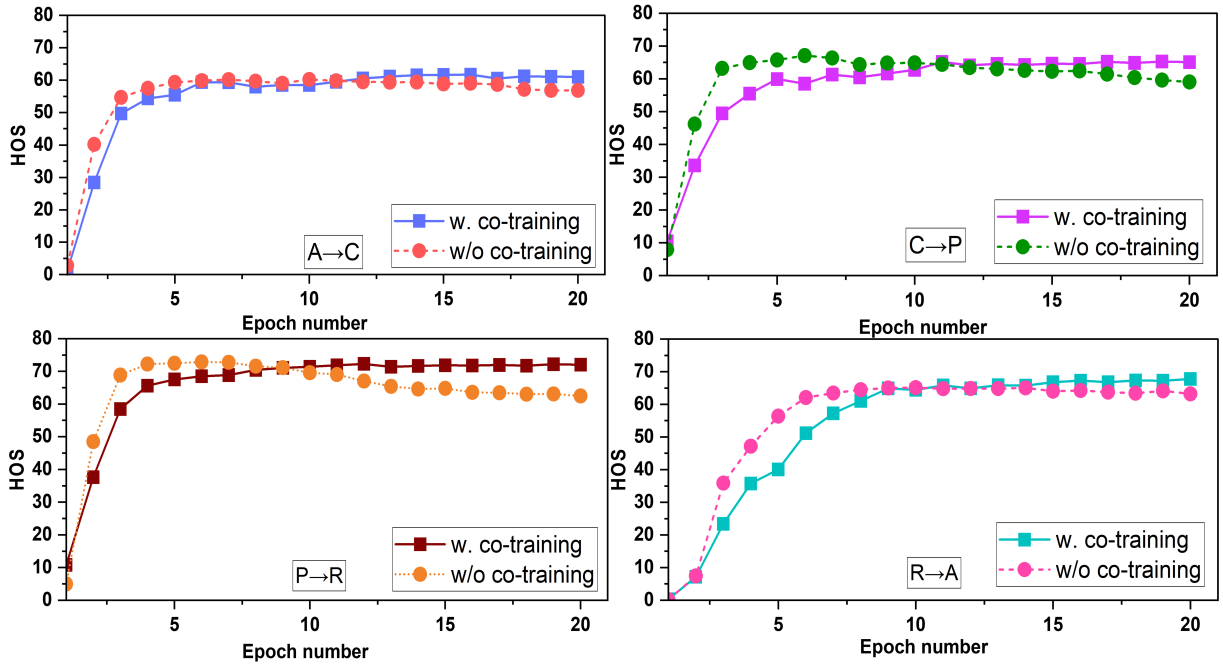


Figure 6.5: Impact of co-training on reducing error accumulation during adaptation on Office-Home dataset.

Table 6.3: Evaluation of separation criterion and distribution modeling for known-unknown sample separation in USD on Office dataset.

Separation criterion	Distribution modeling	Office											
		A → D			A → W			D → A			D → W		
		OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS
JSD	GMM	89.4	70.2	78.6	82.7	73.0	77.6	66.4	85.2	74.6	97.5	97.0	97.2
Entropy	GMM	88.9	70.2	78.4	83.3	74.5	78.6	65.3	90.5	75.9	97.9	93.3	95.5
CE	GMM	90.7	68.6	78.1	90.0	61.8	73.3	69.6	81.0	74.9	98.2	93.3	95.6
JSD	BMM	91.4	53.7	67.7	93.6	53.2	67.8	77.7	72.3	74.9	100.0	82.4	90.3
Entropy	BMM	90.2	60.1	72.1	87.2	78.3	82.5	66.3	88.5	75.8	89.5	92.1	90.8
CE	BMM	96.0	25.0	39.7	93.6	37.8	53.9	81.0	61.2	69.7	100.0	63.3	77.5

Separation criterion	Distribution modeling	Office									VisDA-C		
		W → A			W → D			Avg.					
		OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS
JSD	GMM	68.3	85.4	75.9	98.0	93.6	95.8	83.7	84.1	83.3	57.8	86.7	69.4
Entropy	GMM	60.2	88.5	71.7	98.0	93.1	95.5	82.3	85.0	82.6	57.1	85.4	68.4
CE	GMM	68.5	86.0	76.2	98.0	90.4	94.1	85.8	80.2	82.0	67.3	45.5	54.3
JSD	BMM	77.1	72.5	74.8	100.0	71.3	83.2	90.0	67.6	76.5	67.6	58.3	62.6
Entropy	BMM	60.8	87.1	71.6	100.0	88.8	94.1	82.3	82.5	81.2	42.3	83.4	56.1
CE	BMM	78.4	68.0	72.8	100.0	71.3	83.2	91.5	54.4	66.1	68.6	24.0	35.5

Table 6.4: Ablation study on the objective function, and co-training for USD on Office-Home dataset.

Method	A → C			A → P			A → R			C → A			C → P			C → R		
	OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS
USD (full)	53.3	71.5	61.1	65.7	74.9	70.0	73.3	79.5	76.3	52.2	70.8	60.1	62.4	68.4	65.2	69.3	68.6	68.9
USD w/o $\mathcal{L}_t^{trip}$	52.9	69.9	60.2	66.4	75.1	70.4	73.6	78.9	76.2	52.0	70.0	59.3	62.3	68.5	65.2	68.0	67.8	67.9
USD w/o $\mathcal{L}_t^{con}$	50.5	75.6	60.6	63.3	77.7	69.8	69.6	83.1	75.8	49.4	74.4	59.3	57.9	73.8	64.9	64.3	72.9	68.3
USD w/o $\mathcal{L}_t^{IM}$	50.1	74.7	59.9	64.6	73.5	68.7	73.5	77.7	75.5	51.2	67.9	58.4	60.2	68.3	64.0	66.7	69.0	67.9
USD w/o curriculum	47.5	77.1	58.8	60.8	79.4	68.9	69.3	82.5	75.3	44.7	79.1	57.1	57.8	74.6	65.2	62.2	73.8	67.5
USD w/o co-training	44.0	80.4	56.8	58.5	78.4	67.0	64.1	78.2	70.5	43.4	72.3	54.3	50.5	71.0	59.0	51.4	76.1	61.4

Method	P → A			P → C			P → R			R → A			R → C			R → P			Avg.		
	OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS
USD (full)	54.3	73.8	62.6	47.3	69.6	56.3	70	74.5	72.2	64.6	71.3	67.8	53.8	65.5	59.1	73.3	69.1	71.1	61.6	71.5	65.9
USD w/o $\mathcal{L}_t^{trip}$	51.2	75.9	61.1	47.6	70.4	56.8	69.5	74.2	71.8	63.9	69.9	66.8	51.4	66.9	58.2	73.5	67.6	70.4	61.0	71.3	65.4
USD w/o $\mathcal{L}_t^{con}$	49.4	78.0	60.5	44.9	71.8	55.3	66.0	78.4	71.7	60.3	75.3	67.0	50.1	70.1	58.4	70.6	73.9	72.2	58.0	75.4	65.3
USD w/o $\mathcal{L}_t^{IM}$	50.6	75.4	60.6	45.5	68.1	54.5	68.7	74.1	71.3	63.2	72.8	67.7	49.7	66.5	56.9	73.1	67.3	70.1	59.8	71.3	64.6
USD w/o curriculum	46.1	80.7	58.6	40.4	74.5	52.4	64.7	78.3	70.9	57.9	77.7	66.4	48.0	73.3	58.0	69.0	73.4	71.1	55.7	77.0	64.2
USD w/o co-training	48.3	78.9	59.9	38.5	71.8	50.1	51.6	79.2	62.5	53.9	76.5	63.2	46.6	77.6	58.2	60.7	80.5	69.2	51.0	76.7	61.0

Table 6.5: Ablation study on the pseudolabeling scheme for USD on Office-Home dataset.

Pseudolabel	A → C			A → P			A → R			C → A			C → P			C → R		
	OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS
Ensemble	53.3	71.5	61.1	65.7	74.9	70.0	73.3	79.5	76.3	52.2	70.8	60.1	62.4	68.4	65.2	69.3	68.6	68.9
Clustering	50.8	73.5	60.1	67.0	73.2	69.9	74.8	75.8	75.3	54.3	67.0	60.0	61.5	66.9	64.1	67.2	66.1	66.7
Student Predictions	50.7	74.1	60.2	65.9	73.6	69.5	74.7	78.5	76.5	51.3	68.2	58.6	61.7	67.3	64.4	67.2	69.5	68.3

Pseudolabel	P → A			P → C			P → R			R → A			R → C			R → P			Avg.		
	OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS	OS*	UNK	HOS
Ensemble	54.3	73.8	62.6	47.3	69.6	56.3	70.0	74.5	72.2	64.6	71.3	67.8	53.8	65.5	59.1	73.3	69.1	71.1	61.6	71.5	65.9
Clustering	53.4	73.6	61.9	48.6	69.0	57.1	71.0	71.8	71.4	63.4	72.2	67.5	52.2	68.1	59.1	70.0	68.3	69.2	61.2	70.5	65.2
Student Predictions	53.0	73.2	61.5	46.6	71.3	56.4	68.9	73.8	71.3	61.8	73.0	66.9	53.3	65.8	58.9	72.7	70.6	71.6	60.6	71.6	65.3

are given in Table 6.3. USD uses JSD as the known-unknown sample splitting criterion, while entropy has been extensively used in existing OSDA methods (SHOT, AaD, UADAL etc.) for this purpose. In addition, cross-entropy (CE) loss is a popular criterion for separating clean-noisy samples for noisy label learning (NLL) algorithms [4, 115, 119]. We evaluate all three criteria to find the best performing one. The criterion distribution can be modelled by either Gaussian Mixture Model (GMM) or a Beta Mixture Model (BMM). UADAL models sample entropy distribution using BMM to distinguish between known and unknown samples. Our results in Table 6.3 show that modeling the distribution of the JSD with a GMM outperforms all of the other combinations for unknown sample discovery.

The effect of the JSD threshold δ_t for known-unknown separation on the final HOS is shown in Figure 6.4. The performance is relatively uniform, which suggests robustness of adaptation to the hyperparameter δ_t. Nonetheless, if the threshold is set too high (such as 0.9), too few samples may be denoted as known samples, and this could lead to inferior performance.

Table 6.4 shows the impact of different components of our objective function and the effects of our teacher-student co-training scheme on the final adaptation performance for Office-Home. It is evident that each of our losses ($\mathcal{L}_t^{trip}, \mathcal{L}_t^{con}, \mathcal{L}_t^{IM}$) contributes to the adaptation, and leaving out any one of them hurts performance. We observe that curriculum guidance considerably benefits adaptation and the final average HOS increases by $> 1.5\%$ (from 64.2% to 65.9%) when such guidance is included. Notably, without curriculum, adaptation to the known classes is impacted drastically (OS* falls by $\sim 6\%$), signalling that progressively learning the known class subspace first and then the unknown class subspace later is the superior strategy.

The final row in Table 6.4 presents results in the absence of the teacher network, where the student network conducts the known-unknown sample separation for itself. Both the weakly and strongly augmented samples are fed through the student network, and losses $\mathcal{L}_t^{trip}$, $\mathcal{L}_t^{con}$ are calculated over the student model outputs between the weak and strong augmentations. Empirical results clearly show that co-training in a teacher-student framework is pivotal for mitigating the effect of any imperfect known-unknown separation and average HOS over the 12 domain pairs in Office-Home decreases by $\sim 5\%$ when the teacher network is removed. As seen in Figure 6.5, in the absence of co-training, the student model adapts faster, but its performance drops from its peak during the course of adaptation due to error accumulation. In contrast, adaptation with co-training is slightly slower but maintains its peak performance.

The effect of the pseudolabeling scheme on the adaptation performance for Office-Home is shown in Table 6.5. SHOT and AaD use a self-supervised clustering process built on DeepCluster [14] to get pseudolabels for the known samples. We see that such clustering is not better than taking the hard predictions from the student model as pseudolabels. In open set settings, the unknown samples can drift the known class centroids, leading to faulty clusters. Our multi-view augmentation ensembled pseudolabeling strategy outperforms

both pseudolabeling from clustering or direct student predictions.

6.6 Conclusion

We present Unknown Sample Discovery as a teacher-student co-training framework that conducts source-free open-set domain adaptation. USD calculates the Jensen-Shannon distance between the target model outputs and the pseudolabels and models the distance histogram by a two-component Gaussian mixture model. USD splits the target domain data into known and unknown subsets based on the two Gaussian components. Co-training regularization via contrastive and consistency losses greatly mitigates error accumulation, while curriculum guidance progressively adapts the target model to effectively learn both the known and unknown target feature spaces. The student model in USD has an additional node for the lumped unknown class for all the unknown samples, thus readily classifying any sample as one of the known classes or as the unknown class in subsequent inference stages after adaptation. Empirically, USD outperforms existing SF-OSDA methods and is comparable to non-source-free OSDA techniques.

Chapter 7

Conclusion and Future Work

Transfer learning facilitates the transfer of learned knowledge from one task/domain/modality to another. This dissertation explores two fundamental settings of transfer learning viz. cross-modal supervised transfer learning for guiding training in EO modality to SAR modality and, unsupervised cross-domain transfer learning or domain adaptation from one a source domain to an unlabelled target domain. Our method validates the efficacy of using knowledge distillation from the EO pre-trained network to guide training of a SAR network on an EO-SAR co-registered dataset. For mitigating class bias prevalent in SAR datasets, we further present sampling strategies and multi-stage training schemes, which results in accuracy performance improvements of 2.7%. In future work, denoising the SAR images prior to training can be a viable option to improve model performance. Recent SAR denoising algorithms [55, 168] based on denoising diffusion models are promising in this regard.

In the realm of cross-domain unsupervised adaptation, this dissertation delves into several source-free UDA settings viz. continual DA, black-box DA, and open-set DA. Our methods, with their memory buffers and selective replay, address the problem of catastrophic forgetting seen in continual learning and in continual DA. Our methods are applicable for both static and dynamic (gradually changing) target data distributions, and also across multiple target distributions. ConDA and UCL-GV exhibit strong knowledge retention for the entire target distribution(s) at the end of adaptation. We also introduce four new continual DA aerial datasets based on gradually varying weather conditions, and provide baseline benchmarks on these datasets for ConDA, UCL-GV and continual-SHOT. Continual DA is a practical research problem. Further research in this direction may involve incorporating more generalized large vision models, instead of ResNet-50 backbones for feature extraction. This would lead to stronger performance for the source-trained model, and in turn more robust adaptation to the target distribution. It will also be interesting to explore unique

adaptation algorithms catered towards continual DA of large vision models.

This dissertation also presents a promising method for modeling the target samples using the histogram of the Jensen-Shannon distance between target pseudolabels and network predictions using Gaussian Mixture Models, for both black-box DA and source-free open-set DA. In black-box DA, the unreliable target samples can be separated from the reliable samples using this modeling process, while for open-set DA, the unknown class samples may be separated from the known class samples. Our BBDA method CABB uses a dual-branch network where each branch conducts sample separation for the other, and both are simultaneously adapted on the target data. Our SF-OSDA method USD is based on a teacher-student framework, where the teacher network is updated as a temporally-ensembled version of the student network. The simultaneous adaptation and sample cross-splitting of the two branches in CABB, and temporal consistency loss and triplet contrastive loss between the teacher and student in USD mitigate error accumulation originating from imperfect sample splitting (either into reliable-unreliable samples, or into known-unknown samples). CABB use curriculum guidance to progressively adapt to the reliable samples first, and the unreliable samples later, enabling robust alignment between the source data and the reliable target subset and weakening the adverse effect of outliers or hard-to-adapt target samples on adaptation in the early steps of the process. CABB beats existing SOTA BBDA methods by as much as 2.3%, while USD beats current SOTA SF-OSDA methods by up to $\sim 3\%$. Although our target sample separation method is robust to high noise rates in target pseudolabels, future research may focus on developing more effective sample separation processes. Stronger backbone architectures, such as vision transformers, for the source model would produce more reliable samples and lower the noise rate in the pseudolabels, thereby making the clean-noisy sample separation more accurate. For identifying unknown class samples in OSDA, future research may also focus on cross-validation of pseudolabeling obtained from multiple augmentation methods, in order to filter out unknown target samples that are wrongly but confidently predicted as a known class.

Bibliography

[1] Dmitry Abulkhanov, Ivan Konovalenko, Dmitry Nikolaev, Alexey Savchik, Evgeny Shvets, and Dmitry Sidorchuk. Neural network-based feature point descriptors for registration of optical and SAR images. In *Tenth International Conference on Machine Vision (ICMV 2017)*, volume 10696, page 106960L. International Society for Optics and Photonics, 2018.

[2] AFRL. Moving and stationary target acquisition and recognition.

[3] Waqar Ahmed, Pietro Morerio, and Vittorio Murino. Cleaning noisy labels by negative ensemble learning for source-free unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 1616–1625, January 2022.

[4] Eric Arazo, Diego Ortego, Paul Albert, Noel O'Connor, and Kevin McGuinness. Unsupervised label noise modeling and loss correction. In *International conference on machine learning*, pages 312–321. PMLR, 2019.

[5] Devansh Arpit, Stanislaw Jastrzebski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S. Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, and Simon Lacoste-Julien. A closer look at memorization in deep networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 233–242. PMLR, 06–11 Aug 2017.

[6] Nicolas Audebert, Bertrand Le Saux, and Sébastien Lefèvre. Semantic segmentation of earth observation data using multimodal and multi-scale deep networks. In *Asian conference on computer vision*, pages 180–196. Springer, 2016.

[7] Xueru Bai, Ruihang Xue, Li Wang, and Feng Zhou. Sequence SAR image classification based on bidirectional convolution-recurrent network. *IEEE Transactions on Geoscience and Remote Sensing*, 57(11):9223–9235, 2019.

[8] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48, 2009.

[9] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. *Advances in neural information processing systems*, 32, 2019.

[10] Adeleh Bitarafan, Mahdieh Soleymani Baghshah, and Marzieh Gheisari. Incremental evolving domain adaptation. *IEEE Transactions on Knowledge and Data Engineering*, 28(8):2128–2141, 2016.

[11] Andreea Bobu, Eric Tzeng, Judy Hoffman, and Trevor Darrell. Adapting to continuously shifting domains. *International Conference on Learning Representations*, 2018.

[12] Stefan Braun, Daniel Neil, and Shih-Chii Liu. A curriculum learning method for improved noise robustness in automatic speech recognition. In *2017 25th European Signal Processing Conference (EUSIPCO)*, pages 548–552. IEEE, 2017.

[13] Silvia Bucci, Mohammad Reza Loghmani, and Tatiana Tommasi. On the effectiveness of image rotation for open set domain adaptation. In *European conference on computer vision*, pages 422–438. Springer, 2020.

[14] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European Conference on Computer Vision*, pages 132–149, 2018.

[15] Francisco M Castro, Manuel J Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari. End-to-end incremental learning. In *Proceedings of the European conference on computer vision (ECCV)*, pages 233–248, 2018.

[16] Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Proceedings of the European Conference on Computer Vision*, pages 532–547, 2018.

[17] Defang Chen, Jian-Ping Mei, Yuan Zhang, Can Wang, Zhe Wang, Yan Feng, and Chun Chen. Cross-layer distillation with semantic calibration. *arXiv preprint arXiv:2012.03236*, 2020.

[18] Hong-You Chen and Wei-Lun Chao. Gradual domain adaptation without indexed intermediate domains. *Advances in Neural Information Processing Systems*, 34, 2021.

[19] Liqun Chen, Zhe Gan, Yu Cheng, Linjie Li, Lawrence Carin, and Jingjing Liu. Graph optimal transport for cross-domain alignment. In *International Conference on Machine Learning*, pages 1542–1553. PMLR, 2020.

[20] Minghao Chen, Hongyang Xue, and Deng Cai. Domain adaptation for semantic segmentation with maximum squares loss. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2090–2099, 2019.

[21] Minghao Chen, Shuai Zhao, Haifeng Liu, and Deng Cai. Adversarial-learned loss for domain adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3521–3528, 2020.

[22] Sizhe Chen and Haipeng Wang. SAR target recognition based on deep learning. In *2014 International Conference on Data Science and Advanced Analytics (DSAA)*, pages 541–547. IEEE, 2014.

[23] Sizhe Chen, Haipeng Wang, Feng Xu, and Ya-Qiu Jin. Target classification using the deep convolutional networks for SAR images. *IEEE Transactions on Geoscience and Remote Sensing*, 54(8):4806–4817, 2016.

[24] Yushi Chen, Hanlu Jiang, Chunyang Li, Xiuping Jia, and Pedram Ghamisi. Deep feature extraction and classification of hyperspectral images based on convolutional neural networks. *IEEE Transactions on Geoscience and Remote Sensing*, 54(10):6232–6251, 2016.

[25] Gong Cheng, Peicheng Zhou, and Junwei Han. Learning rotation-invariant convolutional neural networks for object detection in vhr optical remote sensing images. *IEEE Transactions on Geoscience and Remote Sensing*, 54(12):7405–7415, 2016.

[26] Gong Cheng, Peicheng Zhou, and Junwei Han. Rifd-CNN: Rotation-invariant and fisher discriminative convolutional neural networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2884–2893, 2016.

[27] Boris Chidlovskii, Stéphane Clinchant, and Gabriela Csurka. Domain adaptation in the absence of source domain data. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 451–460, 2016.

[28] Tong Chu, Yahao Liu, Jinhong Deng, Wen Li, and Lixin Duan. Denoised maximum classifier discrepancy for source-free unsupervised domain adaptation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pages 472–480, 2022.

[29] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*, 2015.

[30] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation strategies from data. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 113–123, 2019.

[31] Shuhao Cui, Shuhui Wang, Junbao Zhuo, Chi Su, Qingming Huang, and Qi Tian. Gradually vanishing bridge for adversarial domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12455–12464, 2020.

[32] George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.

[33] Clement Dechesne, Sebastien Lefevre, Rodolphe Vadaine, Guillaume Hajduch, and Ronan Fablet. Multi-task deep learning from sentinel-1 sar: ship detection, classification and length estimation. In *BiDS'19: Conference on Big Data from Space*, 2019.

[34] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[35] Jun Ding, Bo Chen, Hongwei Liu, and Mengyuan Huang. Convolutional neural network with data augmentation for SAR target recognition. *IEEE Geoscience and remote sensing letters*, 13(3):364–368, 2016.

[36] Ning Ding, Yixing Xu, Yehui Tang, Chao Xu, Yunhe Wang, and Dacheng Tao. Source-free domain adaptation via distribution estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7212–7222, 2022.

[37] 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 *International Conference on Learning Representations*, 2021.

[38] Kangning Du, Yunkai Deng, Robert Wang, Tuan Zhao, and Ning Li. SAR ATR based on displacement-and rotation-insensitive CNN. *Remote Sensing Letters*, 7(9):895–904, 2016.

[39] Natalia Díaz-Rodríguez, Vincenzo Lomonaco, David Filliat, and Davide Maltoni. Don't forget, there is more than forgetting: new metrics for continual learning, 2018.

[40] D.M. Endres and J.E. Schindelin. A new metric for probability distributions. *IEEE Transactions on Information Theory*, 49(7):1858–1860, 2003.

[41] Haoqi Fan, Bo Xiong, Karttikeya Mangalam, Yanghao Li, Zhicheng Yan, Jitendra Malik, and Christoph Feichtenhofer. Multiscale vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6824–6835, 2021.

[42] Zhen Fang, Jie Lu, Feng Liu, Junyu Xuan, and Guangquan Zhang. Open set domain adaptation: Theoretical bound and algorithm. *IEEE transactions on neural networks and learning systems*, 32(10):4309–4322, 2020.

[43] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[44] Kunihiko Fukushima and Sei Miyake. Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition. In *Competition and cooperation in neural nets*, pages 267–285. Springer, 1982.

[45] Hidetoshi Furukawa. Deep learning for end-to-end automatic target recognition from synthetic aperture radar imagery. *arXiv preprint arXiv:1801.08558*, 2018.

[46] Joao Gama. *Knowledge discovery from data streams*. CRC Press, 2010.

[47] Joao Gama, Raquel Sebastiao, and Pedro Pereira Rodrigues. On evaluating stream learning algorithms. *Machine learning*, 90(3):317–346, 2013.

[48] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *International Conference on Machine Learning*, pages 1180–1189. PMLR, 2015.

[49] 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.

[50] Jie Geng, Hongyu Wang, Jianchao Fan, and Xiaorui Ma. SAR image classification via deep recurrent encoding neural networks. *IEEE Transactions on Geoscience and Remote Sensing*, 56(4):2255–2269, 2017.

[51] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 315–323. JMLR Workshop and Conference Proceedings, 2011.

[52] Ryan Gomes, Andreas Krause, and Pietro Perona. Discriminative clustering by regularized information maximization. In *Proceedings of the 23rd International Conference on Neural Information Processing Systems-Volume 1*, pages 775–783, 2010.

[53] Luis Gómez-Chova, Devis Tuia, Gabriele Moser, and Gustau Camps-Valls. Multimodal classification of remote sensing images: A review and future directions. *Proceedings of the IEEE*, 103(9):1560–1584, 2015.

[54] Boqing Gong, Yuan Shi, Fei Sha, and Kristen Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2066–2073. IEEE, 2012.

[55] Soumee Guha and Scott T. Acton. Sddpm: Speckle denoising diffusion probabilistic models, 2023.

[56] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. *Advances in neural information processing systems*, 31, 2018.

[57] Christopher G Harris, Mike Stephens, et al. A combined corner and edge detector. In *Alvey vision conference*, volume 15, pages 10–5244. Citeseer, 1988.

[58] Tyler L Hayes, Kushal Kafle, Robik Shrestha, Manoj Acharya, and Christopher Kanan. Remind your neural network to prevent catastrophic forgetting. In *European Conference on Computer Vision*, pages 466–483. Springer, 2020.

[59] Tyler L Hayes and Christopher Kanan. Selective replay enhances learning in online continual analogical reasoning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3502–3512, 2021.

[60] Tyler L Hayes, Giri P Krishnan, Maxim Bazhenov, Hava T Siegelmann, Terrence J Sejnowski, and Christopher Kanan. Replay in deep learning: Current approaches and missing biological elements. *Neural Computation*, 33(11):2908–2950, 2021.

[61] Chu He, Shuang Li, Zixian Liao, and Mingsheng Liao. Texture classification of PolSAR data based on sparse coding of wavelet polarization textons. *IEEE Transactions on Geoscience and Remote Sensing*, 51(8):4576–4590, 2013.

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

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

[64] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

[65] Byeongho Heo, Minsik Lee, Sangdoo Yun, and Jin Young Choi. Knowledge transfer via distillation of activation boundaries formed by hidden neurons. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3779–3787, 2019.

[66] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[67] Geoffrey E Hinton and Sam Roweis. Stochastic neighbor embedding. *Advances in neural information processing systems*, 15, 2002.

[68] Judy Hoffman, Trevor Darrell, and Kate Saenko. Continuous manifold based adaptation for evolving visual domains. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 867–874, 2014.

[69] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1989–1998. PMLR, 10–15 Jul 2018.

[70] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Lifelong learning via progressive distillation and retrospection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 437–452, 2018.

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

[72] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[73] Fan Hu, Gui-Song Xia, Jingwen Hu, and Liangpei Zhang. Transferring deep convolutional neural networks for the scene classification of high-resolution remote sensing imagery. *Remote Sensing*, 7(11):14680–14707, 2015.

[74] Weihua Hu, Takeru Miyato, Seiya Tokui, Eiichi Matsumoto, and Masashi Sugiyama. Learning discrete representations via information maximizing self-augmented training. In *International Conference on Machine Learning*, pages 1558–1567. PMLR, 2017.

[75] Jiaxing Huang, Dayan Guan, Aoran Xiao, and Shijian Lu. Model adaptation: Historical contrastive learning for unsupervised domain adaptation without source data. *Advances in Neural Information Processing Systems*, 34:3635–3649, 2021.

[76] Lanqing Huang, Bin Liu, Boying Li, Weiwei Guo, Wenhao Yu, Zenghui Zhang, and Wenxian Yu. OpenSARShip: A dataset dedicated to sentinel-1 ship interpretation. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 11(1):195–208, 2018.

[77] Zhongling Huang, Zongxu Pan, and Bin Lei. What, where, and how to transfer in SAR target recognition based on deep CNNs. *IEEE Transactions on Geoscience and Remote Sensing*, 58(4):2324–2336, 2019.

[78] Lloyd H Hughes, Michael Schmitt, Lichao Mou, Yuanyuan Wang, and Xiao Xiang Zhu. Identifying corresponding patches in SAR and optical images with a pseudo-siamese CNN. *IEEE Geoscience and Remote Sensing Letters*, 15(5):784–788, 2018.

[79] Lloyd Haydn Hughes, Nina Merkle, Tatjana Bürgmann, Stefan Auer, and Michael Schmitt. Deep learning for SAR-optical image matching. In *IGARSS 2019-2019 IEEE International Geoscience and Remote Sensing Symposium*, pages 4877–4880. IEEE, 2019.

[80] 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. PMLR, 2015.

[81] Chowdhury Sadman Jahan and Andreas Savakis. Balanced sampling meets imbalanced datasets for sar image classification. In *Geospatial Informatics XIII*, volume 12525, pages 37–45. SPIE, 2023.

[82] Chowdhury Sadman Jahan and Andreas Savakis. Curriculum guided domain adaptation in the dark. *IEEE Transactions on Artificial Intelligence*, 2023.

[83] Chowdhury Sadman Jahan and Andreas Savakis. Unknown sample discovery for source free open set domain adaptation. *arXiv preprint arXiv:2312.03767*, 2023.

[84] Chowdhury Sadman Jahan and Andreas Savakis. Continual domain adaptation on aerial images under gradually degrading weather. *Journal of Applied Remote Sensing*, 18(1):016504–016504, 2024.

[85] Chowdhury Sadman Jahan, Andreas Savakis, and Erik Blasch. Cross-modal knowledge distillation in deep networks for sar image classification. In *Geospatial Informatics XII*, volume 12099, pages 20–27. SPIE, 2022.

[86] Chowdhury Sadman Jahan, Andreas Savakis, and Erik Blasch. Sar image classification with knowledge distillation and class balancing for long-tailed distributions. In *2022 IEEE 14th Image, Video, and Multidimensional Signal Processing Workshop (IVMSP)*, pages 1–5. IEEE, 2022.

[87] Lalit P Jain, Walter J Scheirer, and Terrance E Boult. Multi-class open set recognition using probability of inclusion. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part III 13*, pages 393–409. Springer, 2014.

[88] JoonHo Jang, Byeonghu Na, Dong Hyeok Shin, Mingi Ji, Kyungwoo Song, and Il-Chul Moon. Unknown-aware domain adversarial learning for open-set domain adaptation. *Advances in Neural Information Processing Systems*, 35:16755–16767, 2022.

[89] Daoyun Ji and Matthew A Wilson. Coordinated memory replay in the visual cortex and hippocampus during sleep. *Nature Neuroscience*, 10(1):100–107, 2007.

[90] Xiang Jiang, Qicheng Lao, Stan Matwin, and Mohammad Havaei. Implicit class-conditioned domain alignment for unsupervised domain adaptation. In *International Conference on Machine Learning*, pages 4816–4827. PMLR, 2020.

[91] Xiao Jin, Baoyun Peng, Yichao Wu, Yu Liu, Jiaheng Liu, Ding Liang, Junjie Yan, and Xiaolin Hu. Knowledge distillation via route constrained optimization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1345–1354, 2019.

[92] Alexander B. Jung, Kentaro Wada, Jon Crall, Satoshi Tanaka, Jake Graving, Christoph Reinders, Sarthak Yadav, Joy Banerjee, Gábor Vecsei, Adam Kraft, Zheng Rui, Jirka Borovec, Christian Vallentin, Semen Zhydenko, Kilian Pfeiffer, Ben Cook, Ismael Fernández, François-Michel De Rainville, Chi-Hung Weng, Abner Ayala-Acevedo, Raphael Meudec, Matias Laporte, et al. imgaug. https://github.com/aleju/imgaug, 2020. Online; accessed 07-July-2023.

[93] Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. Decoupling representation and classifier for long-tailed recognition. *arXiv preprint arXiv:1910.09217*, 2019.

[94] Mattias P Karlsson and Loren M Frank. Awake replay of remote experiences in the hippocampus. *Nature Neuroscience*, 12(7):913–918, 2009.

[95] Samia Kazemi, Bariscan Yonel, and Birsen Yazici. Deep learning for direct automatic target recognition from SAR data. In *2019 IEEE Radar Conference (RadarConf)*, pages 1–6. IEEE, 2019.

[96] Henry J Kelley. Gradient theory of optimal flight paths. *Ars Journal*, 30(10):947–954, 1960.

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

[98] Mehran Khodabandeh, Arash Vahdat, Mani Ranjbar, and William G Macready. A robust learning approach to domain adaptive object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 480–490, 2019.

[99] Jangho Kim, SeoungUK Park, and Nojun Kwak. Paraphrasing complex network: Network compression via factor transfer. *arXiv preprint arXiv:1802.04977*, 2018.

[100] Youngeun Kim, Donghyeon Cho, Kyeongtak Han, Priyadarshini Panda, and Sungeun Hong. Domain adaptation without source data. *IEEE Transactions on Artificial Intelligence*, 2(6):508–518, 2021.

[101] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017.

[102] Günter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. Self-normalizing neural networks. *arXiv preprint arXiv:1706.02515*, 2017.

[103] Andreas Krause, Pietro Perona, and Ryan Gomes. Discriminative clustering by regularized information maximization. *Advances in Neural Information Processing Systems*, 23, 2010.

[104] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.

[105] Ananya Kumar, Tengyu Ma, and Percy Liang. Understanding self-training for gradual domain adaptation. In *International Conference on Machine Learning*, pages 5468–5479. PMLR, 2020.

[106] Jogendra Nath Kundu, Naveen Venkat, R Venkatesh Babu, et al. Universal source-free domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4544–4553, 2020.

[107] Jogendra Nath Kundu, Naveen Venkat, Ambareesh Revanur, R Venkatesh Babu, et al. Towards inheritable models for open-set domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12376–12385, 2020.

[108] Vinod Kumar Kurmi, Shanu Kumar, and Vinay P Namboodiri. Attending to discriminative certainty for domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 491–500, 2019.

[109] Ilja Kuzborskij and Francesco Orabona. Stability and hypothesis transfer learning. In *International Conference on Machine Learning*, pages 942–950. PMLR, 2013.

[110] Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. In *International Conference on Learning Representations*, 2017.

[111] Qicheng Lao, Xiang Jiang, and Mohammad Havaei. Hypothesis disparity regularized mutual information maximization. *arXiv preprint arXiv:2012.08072*, 2020.

[112] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

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

[114] Guangrui Li, Guoliang Kang, Yi Zhu, Yunchao Wei, and Yi Yang. Domain consensus clustering for universal domain adaptation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9757–9766, 2021.

[115] Junnan Li, Richard Socher, and Steven C.H. Hoi. Dividemix: Learning with noisy labels as semi-supervised learning. In *International Conference on Learning Representations*, 2020.

[116] Junnan Li, Caiming Xiong, and Steven CH Hoi. Learning from noisy data with robust representation learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9485–9494, 2021.

[117] Junnan Li, Pan Zhou, Caiming Xiong, and Steven CH Hoi. Prototypical contrastive learning of unsupervised representations. *arXiv preprint arXiv:2005.04966*, 2020.

[118] Ke Li, Gang Wan, Gong Cheng, Liqiu Meng, and Junwei Han. Object detection in optical remote sensing images: A survey and a new benchmark. *ISPRS Journal of Photogrammetry and Remote Sensing*, 159:296–307, 2020.

[119] Shikun Li, Xiaobo Xia, Shiming Ge, and Tongliang Liu. Selective-supervised contrastive learning with noisy labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 316–325, June 2022.

[120] Shuang Li, Chi Harold Liu, Binhui Xie, Limin Su, Zhengming Ding, and Gao Huang. Joint adversarial domain adaptation. In *Proceedings of the 27th ACM International Conference on Multimedia*, pages 729–737, 2019.

[121] Shutao Li, Weiwei Song, Leyuan Fang, Yushi Chen, Pedram Ghamisi, and Jón Atli Benediktsson. Deep learning for hyperspectral image classification: An overview. *IEEE Transactions on Geoscience and Remote Sensing*, 57(9):6690–6709, 2019.

[122] Ying Li, Haokui Zhang, and Qiang Shen. Spectral–spatial classification of hyperspectral imagery with 3d convolutional neural network. *Remote Sensing*, 9(1):67, 2017.

[123] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947, 2017.

[124] Jian Liang, Ran He, Zhenan Sun, and Tieniu Tan. Distant supervised centroid shift: A simple and efficient approach to visual domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2975–2984, 2019.

[125] Jian Liang, Dapeng Hu, and Jiashi Feng. Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In *International Conference on Machine Learning*, pages 6028–6039. PMLR, 2020.

[126] Jian Liang, Dapeng Hu, Jiashi Feng, and Ran He. Dine: Domain adaptation from single and multiple black-box predictors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

[127] Jian Liang, Dapeng Hu, Yunbo Wang, Ran He, and Jiashi Feng. Source data-absent unsupervised domain adaptation through hypothesis transfer and labeling transfer. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11):8602–8617, 2021.

[128] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, pages 740–755. Springer, 2014.

[129] Seppo Linnainmaa. Taylor expansion of the accumulated rounding error. *BIT Numerical Mathematics*, 16(2):146–160, 1976.

[130] Hong Liu, Zhangjie Cao, Mingsheng Long, Jianmin Wang, and Qiang Yang. Separate to adapt: Open set domain adaptation via progressive separation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2927–2936, 2019.

[131] Hong Liu, Mingsheng Long, Jianmin Wang, and Yu Wang. Learning to adapt to evolving domains. In *NeurIPS*, 2020.

[132] Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, Furu Wei, and Baining Guo. Swin transformer v2: Scaling up capacity and resolution. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[133] Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, Furu Wei, and Baining Guo. Swin transformer v2: Scaling up capacity and resolution. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[134] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[135] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[136] MultiMedia LLC. NTIRE 2021 multi-modal aerial view object classification challenge - track 1 (SAR), 2021.

[137] Vincenzo Lomonaco and Davide Maltoni. Core50: a new dataset and benchmark for continuous object recognition. In *Conference on Robot Learning*, pages 17–26. PMLR, 2017.

[138] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning transferable features with deep adaptation networks. In *International Conference on Machine Learning*, pages 97–105. PMLR, 2015.

[139] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Conditional adversarial domain adaptation. In *Advances in Neural Information Processing Systems*, pages 1645–1655, 2018.

[140] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Deep transfer learning with joint adaptation networks. In *International Conference on Machine Learning*, pages 2208–2217. PMLR, 2017.

[141] David G Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[142] Xiaoqiang Lu, Xiangtao Zheng, and Yuan Yuan. Remote sensing scene classification by unsupervised representation learning. *IEEE Transactions on Geoscience and Remote Sensing*, 55(9):5148–5157, 2017.

[143] 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.

[144] Xingjun Ma, Hanxun Huang, Yisen Wang, Simone Romano, Sarah Erfani, and James Bailey. Normalized loss functions for deep learning with noisy labels. In *ICML*, 2020.

[145] Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, page 3. Citeseer, 2013.

[146] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of Learning and Motivation*, volume 24, pages 109–165. Elsevier, 1989.

[147] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[148] W.S. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. In *Bulletin of Mathematical Biophysics 5*, page 115–133, 1943.

[149] Nina Merkle, Stefan Auer, Rupert Müller, and Peter Reinartz. Exploring the potential of conditional adversarial networks for optical and SAR image matching. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 11(6):1811–1820, 2018.

[150] Nina Merkle, Wenjie Luo, Stefan Auer, Rupert Müller, and Raquel Urtasun. Exploiting deep matching and SAR data for the geo-localization accuracy improvement of optical satellite images. *Remote Sensing*, 9(6):586, 2017.

[151] Marvin L Minsky and Seymour A Papert. Perceptrons: expanded edition, 1988.

[152] JH Moon, Debasmit Das, and CS George Lee. Multi-step online unsupervised domain adaptation. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 41172–41576. IEEE, 2020.

[153] Alberto Moreira, Pau Prats-Iraola, Marwan Younis, Gerhard Krieger, Irena Hajnsek, and Konstantinos P Papathanassiou. A tutorial on synthetic aperture radar. *IEEE Geoscience and remote sensing magazine*, 1(1):6–43, 2013.

[154] David AE Morgan. Deep convolutional neural networks for ATR from SAR imagery. In *Algorithms for Synthetic Aperture Radar Imagery XXII*, volume 9475, page 94750F. International Society for Optics and Photonics, 2015.

[155] Lichao Mou, Michael Schmitt, Yuanyuan Wang, and Xiao Xiang Zhu. A CNN for the identification of corresponding patches in SAR and optical imagery of urban scenes. In *2017 Joint Urban Remote Sensing Event (JURSE)*, pages 1–4. IEEE, 2017.

[156] Rafael Müller, Simon Kornblith, and Geoffrey Hinton. When does label smoothing help? *arXiv preprint arXiv:1906.02629*, 2019.

[157] Adugna G Mullissa, Claudio Persello, and Alfred Stein. PolSARNet: A deep fully convolutional network for polarimetric SAR image classification. *IEEE Journal of selected topics in applied earth observations and remote sensing*, 12(12):5300–5309, 2019.

[158] 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.

[159] Navya Nagananda, Abu Md Niamul Taufique, Raaga Madappa, Chowdhury Sadman Jahan, Breton Minnehan, Todd Rovito, and Andreas Savakis. Benchmarking domain adaptation methods on aerial datasets. *Sensors*, 21(23):8070, 2021.

[160] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Icml*, 2010.

[161] Poojan Oza, Vishwanath A Sindagi, Vibashan Vishnukumar Sharmini, and Vishal M Patel. Unsupervised domain adaptation of object detectors: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

[162] Joseph O'Neill, Barty Pleydell-Bouverie, David Dupret, and Jozsef Csicsvari. Play it again: reactivation of waking experience and memory. *Trends in Neurosciences*, 33(5):220–229, 2010.

[163] 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.

[164] German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, 2019.

[165] Nikolaos Passalis and Anastasios Tefas. Learning deep representations with probabilistic knowledge transfer. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 268–284, 2018.

[166] Zhongyi Pei, Zhangjie Cao, Mingsheng Long, and Jianmin Wang. Multi-adversarial domain adaptation. In *Thirty-second AAAI Conference on Artificial Intelligence*, 2018.

[167] Xingchao Peng, Ben Usman, Neela Kaushik, Dequan Wang, Judy Hoffman, and Kate Saenko. Visda: A synthetic-to-real benchmark for visual domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 2021–2026, 2018.

[168] Malsha V. Perera, Nithin Gopalakrishnan Nair, Wele Gedara Chaminda Bandara, and Vishal M. Patel. Sar despeckling using a denoising diffusion probabilistic model. *IEEE Geoscience and Remote Sensing Letters*, 20:1–5, 2023.

[169] Maithra Raghu, Thomas Unterthiner, Simon Kornblith, Chiyuan Zhang, and Alexey Dosovitskiy. Do vision transformers see like convolutional neural networks? *Advances in Neural Information Processing Systems*, 34:12116–12128, 2021.

[170] Sayan Rakshit, Anwesh Mohanty, Ruchika Chavhan, Biplab Banerjee, Gemma Roig, and Subhasis Chaudhuri. FRIDA-Generative feature replay for incremental domain adaptation. *Computer Vision and Image Understanding*, page 103367, 2022.

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

[172] Rhammell. Ships in satellite imagery, Jul 2018.

[173] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. *arXiv preprint arXiv:1412.6550*, 2014.

[174] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[175] Mohammad Rostami. Lifelong domain adaptation via consolidated internal distribution. *Advances in Neural Information Processing Systems*, 34, 2021.

[176] Mohammad Rostami, Soheil Kolouri, Eric Eaton, and Kyungnam Kim. Deep transfer learning for few-shot SAR image classification. *Remote Sensing*, 11(11):1374, 2019.

[177] Subhankar Roy, Evgeny Krivosheev, Zhun Zhong, Nicu Sebe, and Elisa Ricci. Curriculum graph co-teaching for multi-target domain adaptation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5351–5360, 2021.

[178] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *2011 International conference on computer vision*, pages 2564–2571. Ieee, 2011.

[179] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

[180] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *European Conference on Computer Vision*, pages 213–226. Springer, 2010.

[181] Kuniaki Saito, Donghyun Kim, Stan Sclaroff, and Kate Saenko. Universal domain adaptation through self supervision. *Advances in neural information processing systems*, 33:16282–16292, 2020.

[182] Kuniaki Saito, Kohei Watanabe, Yoshitaka Ushiku, and Tatsuya Harada. Maximum classifier discrepancy for unsupervised domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3723–3732, 2018.

[183] Kuniaki Saito, Shohei Yamamoto, Yoshitaka Ushiku, and Tatsuya Harada. Open set domain adaptation by backpropagation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 153–168, 2018.

[184] Tim Salimans and Diederik P Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. *arXiv preprint arXiv:1602.07868*, 2016.

[185] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

[186] Colin P Schwegmann, Waldo Kleynhans, Brian P Salmon, Lizwe W Mdakane, and Rory GV Meyer. Very deep learning for ship discrimination in synthetic aperture radar imagery. In *2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pages 104–107. IEEE, 2016.

[187] Yuan Shi and Fei Sha. Information-theoretical learning of discriminative clusters for unsupervised domain adaptation. *arXiv preprint arXiv:1206.6438*, 2012.

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

[189] Valentin I Spitkovsky, Hiyan Alshawi, and Daniel Jurafsky. Baby steps: How "less is more" in unsupervised dependency parsing. In *Proceedings of NIPS Workshop on Grammar Induction, Representation of Language and Language Learning*, 2009.

[190] Andreas Peter Steiner, Alexander Kolesnikov, Xiaohua Zhai, Ross Wightman, Jakob Uszkoreit, and Lucas Beyer. How to train your vit? data, augmentation, and regularization in vision transformers. *Transactions on Machine Learning Research*, 2022.

[191] Robert Stickgold, J Allen Hobson, Roar Fosse, and Magdalena Fosse. Sleep, learning, and dreams: off-line memory reprocessing. *Science*, 294(5544):1052–1057, 2001.

[192] Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *Computer Vision–ECCV 2016 Workshops: Amsterdam, The Netherlands, October 8-10 and 15-16, 2016, Proceedings, Part III 14*, pages 443–450. Springer, 2016.

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

[194] Hui Tang, Ke Chen, and Kui Jia. Unsupervised domain adaptation via structurally regularized deep clustering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8725–8735, 2020.

[195] Shixiang Tang, Peng Su, Dapeng Chen, and Wanli Ouyang. Gradient regularized contrastive learning for continual domain adaptation. In *Proceedings 35th of the AAAI Conference on Artificial Intelligence*, pages 2–13, 2021.

[196] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. *Advances in neural information processing systems*, 30, 2017.

[197] Abu Md Niamul Taufique, Chowdhury Sadman Jahan, and Andreas Savakis. Unsupervised continual learning for gradually varying domains. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 3739–3749, 2022.

[198] Abu Md Niamul Taufique, Chowdhury Sadman Jahan, and Andreas Savakis. Continual unsupervised domain adaptation in data-constrained environments. *IEEE Transactions on Artificial Intelligence*, 5(1):167–178, 2024.

[199] Marco Toldo, Andrea Maracani, Umberto Michieli, and Pietro Zanuttigh. Unsupervised domain adaptation in semantic segmentation: a review. *Technologies*, 8(2):35, 2020.

[200] Antonio Torralba and Alexei A Efros. Unbiased look at dataset bias. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1521–1528. IEEE, 2011.

[201] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7167–7176, 2017.

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

[203] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[204] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5018–5027, 2017.

[205] Riccardo Volpi, Diane Larlus, and Grégory Rogez. Continual adaptation of visual representations via domain randomization and meta-learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4443–4453, 2021.

[206] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, et al. Deep high-resolution representation learning for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[207] Mei Wang and Weihong Deng. Deep visual domain adaptation: A survey. *Neurocomputing*, 312:135–153, 2018.

[208] Qian Wang and Toby Breckon. Unsupervised domain adaptation via structured prediction based selective pseudo-labeling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 6243–6250, 2020.

[209] Qian Wang, Fanlin Meng, and Toby P Breckon. Progressively select and reject pseudo-labelled samples for open-set domain adaptation. *arXiv preprint arXiv:2110.12635*, 2021.

[210] Michael Wilmanski, Chris Kreucher, and Jim Lauer. Modern approaches in deep learning for SAR ATR. In *Algorithms for synthetic aperture radar imagery XXIII*, volume 9843, page 98430N. International Society for Optics and Photonics, 2016.

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

[212] Markus Wulfmeier, Alex Bewley, and Ingmar Posner. Incremental adversarial domain adaptation for continually changing environments. In *2018 IEEE International conference on robotics and automation (ICRA)*, pages 4489–4495. IEEE, 2018.

[213] Gui-Song Xia, Jingwen Hu, Fan Hu, Baoguang Shi, Xiang Bai, Yanfei Zhong, Liangpei Zhang, and Xiaoqiang Lu. Aid: A benchmark data set for performance evaluation of aerial scene classification. *IEEE Transactions on Geoscience and Remote Sensing*, 55(7):3965–3981, 2017.

[214] Haifeng Xia, Handong Zhao, and Zhengming Ding. Adaptive adversarial network for source-free domain adaptation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9010–9019, 2021.

[215] Mengqiu Xu, Ming Wu, Kaixin Chen, Chuang Zhang, and Jun Guo. The eyes of the gods: A survey of unsupervised domain adaptation methods based on remote sensing data. *Remote Sensing*, 14(17):4380, 2022.

[216] Qinwei Xu, Ruipeng Zhang, Ya Zhang, Yanfeng Wang, and Qi Tian. A fourier-based framework for domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14383–14392, June 2021.

[217] Renjun Xu, Pelen Liu, Liyan Wang, Chao Chen, and Jindong Wang. Reliable weighted optimal transport for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4394–4403, 2020.

[218] Ruijia Xu, Guanbin Li, Jihan Yang, and Liang Lin. Larger norm more transferable: An adaptive feature norm approach for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1426–1435, 2019.

[219] Jianfei Yang, Xiangyu Peng, Kai Wang, Zheng Zhu, Jiashi Feng, Lihua Xie, and Yang You. Divide to adapt: Mitigating confirmation bias for domain adaptation of black-box predictors. In *The Eleventh International Conference on Learning Representations*, 2023.

[220] Shiqi Yang, Yaxing Wang, Luis Herranz, Shangling Jui, and Joost van de Weijer. Casting a bait for offline and online source-free domain adaptation. *Computer Vision and Image Understanding*, page 103747, 2023.

[221] Shiqi Yang, Yaxing Wang, Joost Van De Weijer, Luis Herranz, and Shangling Jui. Generalized source-free domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8978–8987, 2021.

[222] Shiqi Yang, Yaxing Wang, Kai Wang, SHANGLING JUI, and Joost van de weijer. Attracting and dispersing: A simple approach for source-free domain adaptation. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[223] Yi Yang and Shawn Newsam. Bag-of-visual-words and spatial extensions for land-use classification. In *Proceedings of the 18th SIGSPATIAL international conference on advances in geographic information systems*, pages 270–279, 2010.

[224] Wei Yao, Dimitrios Marmanis, and Mihai Datcu. Semantic segmentation using deep neural networks for SAR and optical image pairs. In *BiDS'17: Conference on Big Data from Space*, 2017.

[225] Li Yi, Gezheng Xu, Pengcheng Xu, Jiaqi Li, Ruizhi Pu, Charles Ling, Ian McLeod, and Boyu Wang. When source-free domain adaptation meets learning with noisy labels. In *The Eleventh International Conference on Learning Representations*, 2023.

[226] Asano YM., Rupprecht C., and Vedaldi A. Self-labelling via simultaneous clustering and representation learning. In *International Conference on Learning Representations*, 2020.

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

[228] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. *arXiv preprint arXiv:1612.03928*, 2016.

[229] Werner Zellinger, Thomas Grubinger, Edwin Lughofer, Thomas Natschläger, and Susanne Saminger-Platz. Central moment discrepancy (cmd) for domain-invariant representation learning. *arXiv preprint arXiv:1702.08811*, 2017.

[230] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International Conference on Machine Learning*, pages 3987–3995. PMLR, 2017.

[231] Runzhe Zhan, Xuebo Liu, Derek F Wong, and Lidia S Chao. Meta-curriculum learning for domain adaptation in neural machine translation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14310–14318, 2021.

[232] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations*, 2017.

[233] Haojian Zhang, Yabin Zhang, Kui Jia, and Lei Zhang. Unsupervised domain adaptation of black-box source models. *arXiv preprint arXiv:2101.02839*, 2021.

[234] Xuan Zhang, Pamela Shapiro, Gaurav Kumar, Paul McNamee, Marine Carpuat, and Kevin Duh. Curriculum learning for domain adaptation in neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1903–1915, 2019.

[235] Zhimian Zhang, Haipeng Wang, Feng Xu, and Ya-Qiu Jin. Complex-valued convolutional neural network and its application in polarimetric SAR image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 55(12):7177–7188, 2017.

[236] Chengliang Zhong, Xiaodong Mu, Xiangchen He, Jiaxin Wang, and Ming Zhu. SAR target image classification based on transfer learning and model compression. *IEEE Geoscience and Remote Sensing Letters*, 16(3):412–416, 2018.

[237] Daquan Zhou, Bingyi Kang, Xiaojie Jin, Linjie Yang, Xiaochen Lian, Zihang Jiang, Qibin Hou, and Jiashi Feng. Deepvit: Towards deeper vision transformer. *arXiv preprint arXiv:2103.11886*, 2021.

[238] Guorui Zhou, Ying Fan, Runpeng Cui, Weijie Bian, Xiaoqiang Zhu, and Kun Gai. Rocket launching: A universal and efficient framework for training well-performing light net. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[239] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2223–2232, 2017.