UC Berkeley
UC Berkeley Electronic Theses and Dissertations

Title
Learning Transferable Representations across Domains

Permalink
https://escholarship.org/uc/item/1gm8r9m0

Author
Yue, Xiangyu

Publication Date
2022

Peer reviewed|Thesis/dissertation

eScholarship.org Powered by the California Digital Library
University of California

Learning Transferable Representations across Domains

by

Xiangyu Yue

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Engineering — Electrical Engineering and Computer Sciences

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Alberto Sangiovanni-Vincentelli, Chair
Professor Kurt Keutzer
Professor Sanjit A. Seshia
Professor Francesco Borrelli

Summer 2022

Learning Transferable Representations across Domains

Copyright 2022
by
Xiangyu Yue

Abstract

Learning Transferable Representations across Domains

by

Xiangyu Yue

Doctor of Philosophy in Engineering — Electrical Engineering and Computer Sciences

University of California, Berkeley

Professor Alberto Sangiovanni-Vincentelli, Chair

Deep neural networks have achieved great success in learning representations on a given dataset. However, in many cases, the learned representations are dataset-dependent and cannot be transferred to datasets with different distributions, even for the same task. How to deal with domain shift is crucial to improve the generalization capability of models. Domain adaptation offers a potential solution, allowing us to transfer networks from a source domain with abundant labels onto target domains with only limited or no labels.

In this dissertation, I will present the many ways that we can learn transferable representations under different scenarios, including 1) when the source domain has only limited labels, even only one label per class, 2) when there are multiple labeled source domains, 3) when there are multiple unseen unlabeled target domains. These approaches are general across different data modalities (e.g. vision and language) and can be easily combined to solve other similar domain transfer settings (e.g. adapting from multiple sources with limited labels), enabling models to generalize beyond the source domains. Many of the works transfer knowledge from simulation data to real-world data in order to alleviate the need for expensive manual annotations. Finally, I present our pioneering work on building a LiDAR point cloud simulator, which has further enabled a large amount of domain adaptation work on LiDAR point cloud segmentation adaptation.

To my parents, Xiaoli Qin and Youxi Yue

Contents

List of Figures

List of Tables

Acknowledgments

First and foremost, I want to thank my wonderful advisor, Prof. Alberto Sangiovanni-Vincentelli. Alberto is probably the most visionary, open-minded, and supportive advisor that one can ever have. I am so grateful that Alberto is flexible with my research direction, and is always available whenever I need guidance and help throughout my PhD study. I would also like to give my deepest gratitude to Prof. Kurt Keutzer, for his guidance, encouragement and support during five years of my PhD. Without the advice and support from Alberto and Kurt, this dissertation would not have been possible.

I would like to thank Prof. Sanjit Seshia for years of collaboration at the beginning of my PhD journey. I also want to thank Prof. Francesco Borrelli for his feedback on my qualification exam and dissertation. Thanks again to my committee members above.

Sincerely, I would like to thank mentors Boqing Gong, Bichen Wu, Sicheng Zhao, who provided large amounts of guidance and help during the beginning of my PhD and shaped my research philosophy today to a significant degree. I am thankful to these mentors for investing time in me: Boqing, for his essential guidance and mentorship during my study at Berekely and during internship at Tencent and Google; Bichen, for demonstrating how to do research with high impact, guiding me through the very first research papers I am involved in at Berkeley; Sicheng, for his guidance and help on various projects, teaching me how to approach a new research area and how to mentor junior students.

I am fortunate to have worked on different areas during my PhD, and I would like to thank all other great collaborators along the journey. I want to thank my labmates and collaborators in Alberto's group: Baihong Jin, Inigo Incer, Antonio Iannopollo, Shromona Ghosh, Edward Kim, Zheng Liang, Tung-Wei Lin, Sheng-Jung Yu, Matteo Guarrera, Yingshui Tan, Florian Hofer, Piergiuseppe Mallozzi, Zangwei Zheng, Xiangyu Peng, Qilong Wu, Min Liu, Zhichao Chen, Zheng Zhang; in Kurt's group: Amir Gholami, Peter Jin, Alvin Wan, Noah Golmant, Kiseok Kwon, Zhewei Yao, Zhen Dong, Sheng Shen, Ravi Krishna, Tianjun Zhang, Xuanyu Zhou, Bo Li, Shanghang Zhang, Colorado Reed, Chenfeng Xu, Sehoon Kim, Shijia Yang, Bohan Zhai, Jiachen Lian; and in Sanjit's group: Tommaso Dreossi, Daniel Fremont, Kevin Cheang. Many thanks to my collaborators during my internships at Google Research: Hang Qi, Matthew Brown; at Google [X] Robotics: Yunfei Bai, Daniel Ho; at Baidu AI Lab: Liangjun Zhang; and at Tencent AI Lab: Yang Zhang. Big thanks to other collaborators: Hari Prasanna Das, Yuhui Yuan, Pengfei Xu, Yang Gu, Hanchen Wang, Qi Liu, Sayna Ebrahimi, Yang Gao, Taesung Park, Yaoqing Yang, Zhuang Liu, Alex Zhao, Elicia Ye, Jessica Newman, William Mullen, Prof. Matt Kusner, Prof. Yuxin Chen, Prof. Masayoshi Tomizuka, Prof. Joseph Gonzalez, Prof. Trevor Darrell.

I owe huge thanks to my partner, Yiyi He, for accompanying me during the years at Berkeley through the ups and downs.

Finally, my deepest gratitude to my parents, Youxi Yue and Xiaoli Qin, for their unconditionally love and support throughout my life. I want to thank them for encouraging and supporting me to pursue my dream since I was a child, for teaching me how to live and how to love. I feel so lucky to be their son and I believe whatever I achieve also belongs to them.

Chapter 1

Introduction

1.1 Chapter Overview

This chapter introduces the motivation and summarizes the key contributions of the dissertation. The organization of the content in this dissertation is also presented.

1.2 Motivation

In recent years, deep neural networks have achieved huge success across wide range of artificial intelligence problems, including computer vision, natural language process, and robotics. At the heart of these progress is the paradigm of supervised learning, under which a labeled dataset is collected and models are trained to predict the labels give the corresponding input. Deep neural networks usually contains a large amount of parameters in order to solve complex real-world problems; and in order for the models not to become overfitting, the labeled datasets are usually in quite large scale.

However, in many real-world applications, collecting large-scale labeled data is always available, as labeling can be expensive, time-consuming, and difficult. For example, in fine-grained recognition only the labels provided by experts are reliable; for semantic segmentation, labeling each image of the Cityscapes dataset takes about 1.5 hours; point-wise 3D LiDAR point clouds are very difficult to label. One essential way to alleviate the need for labeled data is transferring knowledge from one dataset to another. However, due to *domain shift* or *dataset bias*, directly applying a model trained on one dataset to another dataset collected from a different source often leads to significant performance drop. Domain shift does exist. For example, images taken under different weather and time of day; hand-writing from different people; simulation images and real-world images, etc.

Domain adaptation seeks to tackle the learning problem in which the test data do not come from the same distribution as the training data. The training distribution is referred to as the *source domain*, while the test distribution is referred to as the *target domain*. It is often assumed that the source domain contains a large amount of labels, while the target

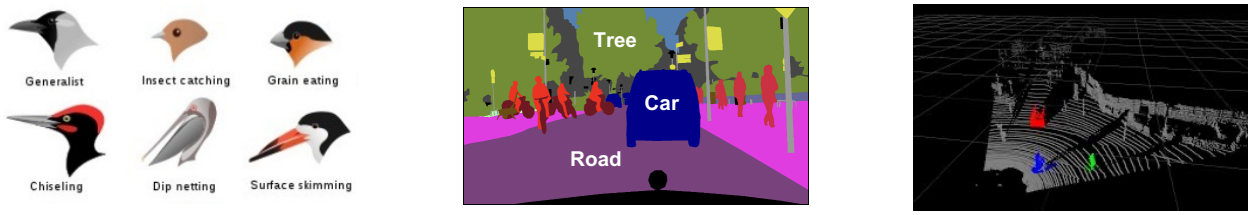


Figure 1.1: Large-scale labeled datasets are not always available, as annotation can be expensive, time-consuming and difficult. (1) For fine-grained recognition, only labels provided by experts are reliable [9]; (2) For semantic segmentation, labeling each image of the cityscapes dataset takes around 90 minutes [41]; (3) Point-wise 3D LiDAR point clouds are very difficult to label [241].

has no labels or very limited labels. A large effort from the research community has been placed on the problem of *Unsupervised Domain Adaptation (UDA)*, where there is a fully labeled source domain and an unlabeled target domain; and the goal is to train a model with the labeled source and unlabeled target and achieve high performance on the target domain. However, the problem setting is too simple and not practical enough:

- Fully labeled source. Although this is reasonable in some cases, e.g. simulation data as source where labeling is almost free; in many applications e.g. medical imaging, even a fully labeled source is hard to get.
- Single target domain. This requires that the target domain be known and available during training. Yet, in the real-world, we often want to train a model and expect it could generalize well to various unseen domains.
- Single source domain. This can underutilize the resources available, since we might have access to multiple source domains. For instance, we have two simulation environments from different companies.

Under diverse scenarios in the real world, the desired domain adaptation setting can be quite different from and more complicated than unsupervised domain adaptation.

1.3 Research Contributions

This dissertation presents a series of studies on learning transferable representations under various scenarios. Specifically, we propose the following methods: a framework that performs adaptation to an unlabeled target domain from a source with limited labels [265], a method that learns image semantic segmentation models that generalize well to various

unseen domains [263], a method that adapts semantic segmentation models from multiple source domains to a target domain [279], a method that adapts models for textural sentiment analysis from multiple sources to a target domain [275], a framework that adapts to an unlabeled target domain from multiple sources with limited labels [264], a LiDAR point cloud simulation environment that enables further research on LiDAR point cloud segmentation adaptation [262].

The contributions of the thesis are as follows:

Chapter 2 considers the task of few-shot unsupervised domain adaptation, adapting to a target domain from a source domain with only few annotations. To cope with this problem, previous work performed instance-wise cross-domain self-supervised learning, followed by an additional fine-tuning stage. However, the instance-wise self-supervised learning only learns and aligns low-level discriminative features, and the cross-domain matching is not robust. We instead propose an end-to-end Prototypical Cross-domain Self-Supervised Learning framework (PCS) for few-shot unsupervised domain adaptation. PCS not only performs cross-domain low-level feature alignment, but it also encodes and aligns semantic structures in the shared embedding space across domains. Our framework captures category-wise semantic structures of the data by in-domain prototypical contrastive learning; and performs feature alignment through cross-domain prototypical self-supervision.

Chapter 3 focus on the task of segmentation domain generalization, learning a segmentation model that generalizes well to various unseen domains. The segmentation network is trained without any data of target domains and tested on the unseen target domains. We propose a new approach of domain randomization and pyramid consistency to learn a model with high generalizability. First, we propose to randomize the synthetic images with the styles of real images in terms of visual appearances using auxiliary datasets, in order to effectively learn domain-invariant representations. Second, we further enforce pyramid consistency across different "stylized" images and within an image, in order to learn domaininvariant and scale-invariant features, respectively.

Chapter 4 targets the task of multi-source adaptation for semantic segmentation, adapting a segmentation model to a given target domain from multiple source domains. Existing methods mainly focus on a single-source setting, which cannot easily handle a more practical scenario of multiple sources with different distributions. In this chapter, we propose to investigate multi-source domain adaptation for semantic segmentation. Specifically, we design a novel framework, termed Multi-source Adversarial Domain Aggregation Network (MADAN), which can be trained in an end-to-end manner. First, we generate an adapted domain for each source with dynamic semantic consistency while aligning at the pixel-level cycle-consistently towards the target. Second, we propose sub-domain aggregation discriminator and cross-domain cycle discriminator to make different adapted domains more closely aggregated. Finally, feature-level alignment is performed between the aggregated domain and target domain while training the segmentation network.

Chapter 5 further investigates multi-source adaptation for textual sentiment analysis, beyond computer vision tasks. Existing multi-source domain adaptation (MDA) methods either fail to extract some discriminative features in the target domain that are related to sentiment,

neglect the correlations of different sources and the distribution difference among different sub-domains even in the same source, or cannot reflect the varying optimal weighting during different training stages. In this chapter, we propose a novel instance-level MDA framework, named curriculum cycle-consistent generative adversarial network (C-CycleGAN), to address the above issues. Specifically, C-CycleGAN consists of three components: (1) pre-trained text encoder which encodes textual input from different domains into a continuous representation space, (2) intermediate domain generator with curriculum instance-level adaptation which bridges the gap across source and target domains, and (3) task classifier trained on the intermediate domain for final sentiment classification. C-CycleGAN transfers source samples at instance-level to an intermediate domain that is closer to the target domain with sentiment semantics preserved and without losing discriminative features. Further, our dynamic instance-level weighting mechanisms can assign the optimal weights to different source samples in each training stage.

Chapter 6 considers both the few-shot and multi-source settings; investigating Multi-source Few-shot Domain Adaptation (MFDA), a new domain adaptation scenario with limited multi-source labels and unlabeled target data. As we show, existing methods often fail to learn discriminative features for both source and target domains in the MFDA setting. Therefore, we propose a novel framework, termed Multi-Source Few-shot Adaptation Network (MSFAN), which can be trained end-to-end in a non-adversarial manner. MSFAN operates by first using a type of prototypical, multi-domain, self-supervised learning to learn features that are not only domain-invariant but also class-discriminative. Second, MSFAN uses a small, labeled support set to enforce feature consistency and domain invariance across domains. Finally, prototypes from multiple sources are leveraged to learn better classifiers.

Chapter 7 considers the synthetic labeled data generalization for LiDAR segmentation adaptation. We present a framework to rapidly create point clouds with accurate pointlevel labels from a computer game. To our best knowledge, this is the first work on LiDAR point cloud simulation framework for autonomous driving. The framework supports data collection from both auto-driving scenes and user-configured scenes. Point clouds from auto-driving scenes can be used as training data for deep learning algorithms, while point clouds from user-configured scenes can be used to systematically test the vulnerability of a neural network, and use the falsifying examples to make the neural network more robust through retraining. In addition, the scene images can be captured simultaneously for sensor fusion tasks, with a method proposed for automatic registration between the point clouds and scene images. This enables a line of research work on point cloud segmentation adaptation.

Finally, Chapter 8 summarizes the whole dissertation and discusses future directions.

Chapter 2

Adapting with Few Source Labels

2.1 Chapter Overview

This chapter targets the problem of Few-shot Unsupervised Domain adaptation, where there are only limited labels on source domain data. We propose a framework targeting this problem with prototypical cross-domain self-supervised learning and adaptive prototype-classifier learning.

2.2 Introduction

Deep Learning has achieved remarkable performance in various computer vision tasks, such as image classification [94, 106] and semantic segmentation [144, 273, 27]. Despite high accuracy, deep neural networks trained on specific datasets often fail to generalize to new domains owing to the *domain shift* problem [217, 54, 221]. Unsupervised domain adaptation (UDA) transfers predictive models from a fully-labeled source domain to an unlabeled target domain. Although it is challenging with no label information in the target domain, many UDA methods [221, 100, 147, 69] could achieve high accuracy on the target domain using the abundant explicit supervision in source domain, together with the unlabeled target samples for domain alignment.

In some real-world applications, however, providing large-scale annotations even in the source domain is often challenging due to the high cost and difficulty of annotation. Taking medical imaging for instance, each image of the Diabetic Retinopathy dataset [89] is annotated by a panel of 7 or 8 U.S. board-certified ophthalmologists, with a total group of 54 doctors. Thus practically it is too stringent to assume the availability of source data with abundant labels.

In this chapter, to cope with the labeling costs of the source domain, we instead consider a few-shot unsupervised domain adaptation (FUDA) setting, where only an extremely small fraction of source samples are labeled, while all the rest source and target samples remain unlabeled. Most state-of-the-art UDA methods align source and target features by mini-

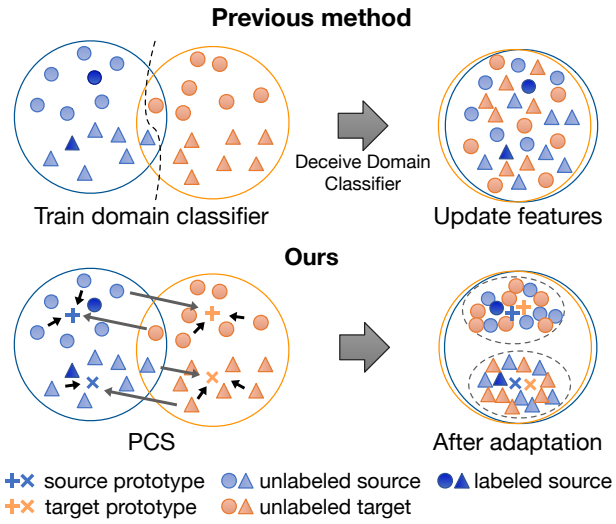


Figure 2.1: We address the task of few-shot unsupervised domain adaptation. Top: Existing domain-classifier based methods align source and target distributions but fail to extract discriminative features due to lack of labeled data. Bottom: Our method estimates prototypes for in-domain and cross-domain self-supervised learning to extract domain-aligned discriminative features.

mizing some form of distribution distances [147, 145, 205, 69], and learn the discriminative representation by minimizing the supervision loss on fully-labeled source domain data. In FUDA, however, since we have a very limited number of labeled source samples, it is much harder to learn discriminative features in the source domain, not to mention in the target domain.

Several recent papers [95, 30, 88, 163, 245] on self-supervised learning (SSL) present promising representation learning results on images from a single domain and [119] further extended to perform SSL across two domains for better domain adaptation performance. Despite the improved performance, the instance-based method in [119] has some fundamental weaknesses. First, the semantic structure of the data is not encoded by the learned structure. This is because the in-domain self-supervision in [119] treats two instances as negative pairs as long as they are from different samples, regardless of the semantic similarity. Consequently, many instances sharing the same semantic are undesirably pushed apart in the feature space. Second, the cross-domain instance-to-instance matching in [119] is very sensitive to abnormal samples. Imagine a case where the embeddings of source and target samples are far apart (i.e. the domain gap is large) and one abnormal source sample is mapped closer to all target samples than any other source sample. Then the method in [119] would match all target samples to the same source sample (cf. Figure 2.3). For a given sample, the matched sample in the other domain may change drastically during the training procedure, making the optimization harder to converge. Third, the two-stage pipeline (i.e. SSL followed by

domain adaptation) is complicated and experiments show that the optimal DA methods for different datasets are different. As a result, the training is rather cumbersome and it is unclear how to choose the optimal DA method in the second stage for different datasets.

In this chapter, we propose *Prototypical Cross-domain Self-supervised learning*, a novel single-stage framework for FUDA that unifies representation learning and domain alignment with few-shot labeled source samples. PCS contains three major components to learn both discriminative and domain-invariant features. First, PCS performs in-domain prototypical self-supervision to implicitly encode the semantic structure of data into the embedding space. This is motivated by [134], but we further leverage the known semantic information of the task and learn better semantic structure in each domain. Second, PCS performs cross-domain instance-to-prototype matching to transfer knowledge from source to the target in a more robust manner. Instead of instance-to-instance matching, matching a sample to a prototype (*i.e.* representative embedding for a group of instances that are semantically similar) is more robust to abnormal instances in the other domain and makes the optimization converge faster and more smoothly. Third, PCS unifies prototype learning with cosine classifier and update cosine classifier adaptively with source and target prototypes. transfers from source prototypes to target prototypes for better performance on the target domain. In order to further mitigate the effect of cross-domain mismatching, we perform entropy maximization to obtain a more diversified output. We show that together with entropy minimization, this is equivalent to maximizing the mutual information (MI) between input image and the network prediction.

To summarize, our contributions are three-fold:

- We propose a novel Prototypical Cross-domain Self-supervised learning framework (PCS) for few-shot unsupervised Domain Adaptation.
- We propose to leverage prototypes to perform better semantic structure learning, discriminative feature learning, and cross-domain alignment in a unified, unsupervised and adaptive manner.
- While it is hard to choose the optimal domain adaptation method in the complex two-stage framework [119], PCS can be easily trained in an end-to-end matter, and outperforms all state-of-the-art methods by a large margin across multiple benchmark datasets.

2.3 Related Work

Domain Adaptation. Unsupervised Domain Adaptation (UDA) [84] addresses the problem of transferring knowledge from a fully-labeled source domain to an unlabeled target domain. Most UDA methods have focused on feature distribution alignment. Discrepancy-based methods explicitly compute the Maximum Mean Discrepancy (MMD) [87] between the source and the target to align the two domains [147, 222, 148]. Long *et al.* [146] proposed to align the joint distributions using the Joint MMD criterion. Sun *et al.* [205] and Zhuo

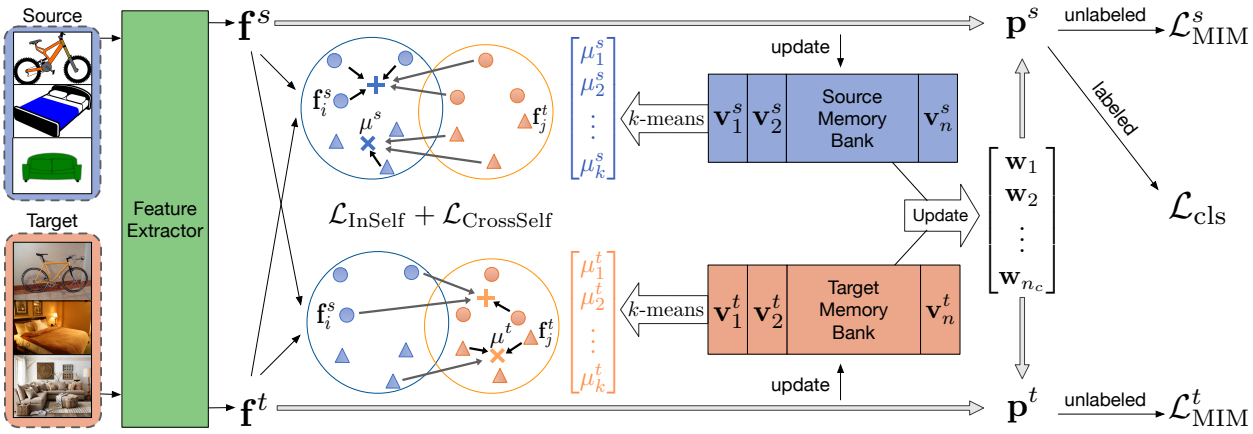


Figure 2.2: An overview of the PCS framework. In-domain and cross-domain self-supervision are performed between normalized feature vectors $\mathbf{f}$ and prototypes μ computed by clustering vectors $\mathbf{v}$ in memory banks. Features with confident predictions ($\mathbf{p}$) are used to adaptively update classifier vectors $\mathbf{w}$. MI maximization and classification loss are further used to extract discriminative features.

et al. [289] further proposed to align second-order statistics of source and target features. With the development of Generative Adversarial Networks [83], additional papers proposed to perform domain alignment in the feature space with adversarial learning [68, 221, 100, 250, 145, 197]. Recently, image translation methods, e.g. [285, 139] have been adopted to further improve domain adaptation by performing pixel-level alignment [100, 17, 184, 159, 263, 189, 199]. Instead of explicit feature alignment, Saito *et al.* [188] proposed minimax entropy for adaptation. While these methods have full supervision on the source domain, similar to [119], we focus on a new adaptation setting with few source labels.

Self-supervised Learning. Self-supervised learning (SSL) is a subset of unsupervised learning methods where supervision is automatically generated from the data [114, 51, 267, 162, 78, 231]. One of the most common strategies for SSL is handcrafting auxiliary pretext tasks predicting future, missing or contextual information. In particular, image colorization [267, 128], patch location prediction [51, 52], image jigsaw puzzle [162], image inpainting [170] and geometric transformations [78, 57] have been shown to be helpful. Currently, contrastive learning [5, 95, 163, 214, 155] has achieved state-of-the-art performance on representation learning [88, 30, 35, 31]. Most contrastive methods are instance-wise, aiming to learn an embedding space where samples from the same instance are pulled closer and samples from different instances are pushed apart [245, 30]. Recently, contrastive learning based on prototypes has shown promising results in representation learning [134, 4, 22, 71].

Self-supervised Learning for Domain Adaptation. Self-supervision-based methods incorporate SSL losses into the original task network [77, 76]. Reconstruction was first utilized as self-supervised task in some early works [77, 76], in which source and target share

the same encoder to extract domain-invariant features. To capture both domain-specific and shared properties, Bousmalis *et al.* [17] explicitly extracts image representations into two spaces, one private for each domain and one shared across domains. In [21], solving jigsaw puzzle [162] was leveraged as a self-supervision task to solve domain adaptation and generalization. Sun *et al.* [210] further proposed to perform adaptation by jointly learning multiple self-supervision tasks. The feature encoder is shared by both source and target images, and the extracted features are then fed into different self-supervision task heads. Recently, based on instance discrimination [245], Kim *et al.* [119] proposed a cross-domain SSL approach for adaptation with few source labels. SSL has also been incorporated for adaptation in other fields, including point cloud recognition [1], medical imaging [108], action segmentation [28], robotics [110], facial tracking [257], *etc.*

2.4 Approach

In few-shot unsupervised domain adaptation, we are given a very limited number of labeled source images $\mathcal{D}_s = \left\{(\mathbf{x}_i^s, y_i^s)\right\}_{i=1}^{N_s}$, as well as unlabeled source images $\mathcal{D}_{su} = \left\{(\mathbf{x}_i^{su})\right\}_{i=1}^{N_{su}}$. In the target domain, we are only given unlabeled target images $\mathcal{D}_{tu} = \left\{\left(\mathbf{x}_i^{tu}\right)\right\}_{i=1}^{N_{tu}}$. The goal is to train a model on $\mathcal{D}_s, \mathcal{D}_{su}$, and $\mathcal{D}_{tu}$; and evaluate on $\mathcal{D}_{tu}$.

The base model consists of a feature encoder F, a ℓ_2 normalization layer, which outputs a normalized feature vector $\mathbf{f} \in \mathbb{R}^d$ and a cosine similarity-based classifier C.

In-domain Prototypical Contrastive Learning

We learn a shared feature encoder F that extracts discriminative features in both domains. Instance Discrimination [245] is employed in [119] to learn discriminative features. As an instance-wise contrastive learning method, it results in an embedding space where all instances are well separated. Despite promising results, instance discrimination has a fundamental weakness: the semantic structure of the data is not encoded by the learned representations. This is because two instances are treated as negative pairs as long as they are from different samples, regardless of their semantics. For a single domain, ProtoNCE [134] is proposed to learn semantic structure of the data by performing iterative clustering and representation learning. The goal is to drive features within the same cluster to become more aggregated and features in different clusters further apart.

However, naively applying ProtoNCE to $\mathcal{D}_s \cup \mathcal{D}_{su} \cup \mathcal{D}_{tu}$ in our domain adaptation setting would cause potential problems. Primarily due to the domain shift, images of different classes from different domains can be incorrectly aggregated into the same cluster, and images of the same class from different domains can be mapped into clusters that are far apart. To mitigate these problems, we propose to perform prototypical contrastive learning separately in $\mathcal{D}_s \cup \mathcal{D}_{su}$ and in $\mathcal{D}_{tu}$. This aims to prevent the incorrect clustering of images across domains and indiscriminative feature learning.

Specifically, two memory banks $\boldsymbol{V^s}$ and $\boldsymbol{V^t}$ are maintained for source and target respectively:

$$\boldsymbol{V^s} = [\mathbf{v}_1^s, \cdots, \mathbf{v}_{(N_s+N_{su})}^s], \ \boldsymbol{V^t} = [\mathbf{v}_1^t, \cdots, \mathbf{v}_{N_{tu}}^t], \tag{2.1}$$

where $\mathbf{v}_i$ is the stored feature vector of $\mathbf{x}_i$, initialized with $\mathbf{f}_i$ and updated with a momentum m after each batch:

$$\mathbf{v}_i \leftarrow m\mathbf{v}_i + (1-m)\mathbf{f}_i. \tag{2.2}$$

In order for in-domain prototypical contrastive learning, k-means clustering is performed on $\boldsymbol{V^s}$ and $\boldsymbol{V^t}$ to get source clusters $\boldsymbol{C}^s = \{C_1^{(s)}, C_2^{(s)}, \ldots, C_k^{(s)}\}$ and similarly $\boldsymbol{C}^t$ with normalized source prototypes $\{\mu_j^s\}_{j=1}^k$ and normalized target prototypes $\{\mu_j^t\}_{j=1}^k$. Specifically, $\mu_j^s = \frac{\mathbf{u}_j^s}{\|\mathbf{u}_j^s\|}$, where $\mathbf{u}_j^s = \frac{1}{|C_j^{(s)}|}\sum_{\mathbf{v}_i^s \in C_j^{(s)}} \mathbf{v}_i^s$. We explain only on the source domain for succinct notation, all operations are performed on target similarly.

During training, with the feature encoder F, we compute a feature vector $\mathbf{f}_i^s = F(\mathbf{x}_i^s)$. To perform in-domain prototypical contrastive learning, we compute the similarity distribution vector between $\mathbf{f}_i^s$ and $\{\mu_j^s\}_{j=1}^k$ as $P_i^s = [P_{i,1}^s, P_{i,2}^s, \ldots, P_{i,k}^s]$, with

$$P_{i,j}^s = \frac{\exp(\mu_j^s \cdot \mathbf{f}_i^s/\phi)}{\sum_{r=1}^k \exp(\mu_r^s \cdot \mathbf{f}_i^s/\phi)}, \tag{2.3}$$

where ϕ is a temperature value determining the level of concentration. Then the in-domain prototypical contrastive loss can be written as:

$$\mathcal{L}_{\text{PC}} = \sum_{i=1}^{N_s+N_{su}} \mathcal{L}_{CE}(P_i^s, c_s(i)) + \sum_{i=1}^{N_{tu}} \mathcal{L}_{CE}(P_i^t, c_t(i)) \tag{2.4}$$

where $c_s(\cdot)$ and $c_t(\cdot)$ return the cluster index of the instance.

Due to the randomness in clustering, we perform k-means on the samples M times with different number of clusters $\{k_m\}_{m=1}^M$. Moreover, in the FUDA setting, since the number of classes n_c is known, we set $k_m = n_c$ for most m. The overall loss for in-domain self-supervision is:

$$\mathcal{L}_{\text{InSelf}} = \frac{1}{M}\sum_{m=1}^{M} \mathcal{L}_{\text{PC}}^{(m)} \tag{2.5}$$

Cross-domain Instance-Prototype SSL

In order to explicitly enforce learning domain-aligned and more discriminative features in both source and target domains, we perform cross-domain instance-prototype self-supervised learning.

Many previous works focus on domain alignment via discrepancy minimization or adversarial learning. However, these methods provide inferior performance or have unstable training. Moreover, most of them focus on distribution matching, without considering semantic similarity matching across domains. Instance-instance matching [119] is proposed to

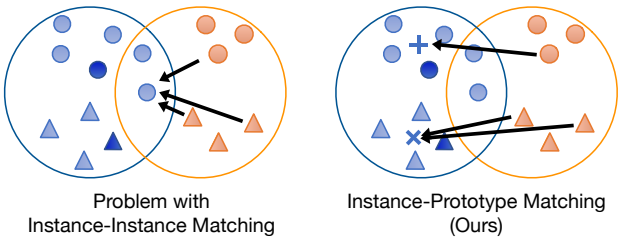


Figure 2.3: Comparison of cross-domain instance-instance (I-I) matching [119] (left) and our cross-domain instance-prototype (I-P) matching (right). Left: I-I incorrectly matches all orange samples to the same blue sample. Right: I-P robustly matches samples to the correct prototypes.

match an instance i to another instance j in the other domain with the most similar representation. However, due to the domain gap, instances can be easily mapped to instances of different classes in the other domain. In some cases, if an outlier in one domain is extremely close to the other domain, it will be matched to all the instances in the other domain, as illustrated in Figure 2.3.

Instead, our method discovers positive matching as well as negative matchings between instance and cluster prototypes in different domains. To find a matching for an instance i, we perform entropy minimization on the similarity distribution vector between its representation, e.g. $\mathbf{f}_i^s$ and the centroids of the other domain, e.g. $\{\mu_j^t\}_{j=1}^k$.

Specifically, given feature vector $\mathbf{f}_i^s$ in the source domain, and centroids $\{\mu_j^t\}_{j=1}^k$ in the target domain, we first compute the similarity distribution vector $P_i^{s\to t} = [P_{i,1}^{s\to t}, \ldots, P_{i,k}^{s\to t}]$, in which

$$P_{i,j}^{s\to t} = \frac{\exp(\mu_j^t \cdot \mathbf{f}_i^s/\tau)}{\sum_{r=1}^k \exp(\mu_r^t \cdot \mathbf{f}_i^s/\tau)}. \tag{2.6}$$

Then we minimize the entropy of $P_i^{s\to t}$, which is:

$$H(P_i^{s\to t}) = -\sum_{j=1}^{k} P_{i,j}^{s\to t} \log P_{i,j}^{s\to t}. \tag{2.7}$$

Similarly, we can compute $H(P_i^{t\to s})$, and the final loss for cross-domain instance-prototype SSL is:

$$\mathcal{L}_{\text{CrossSelf}} = \sum_{i=1}^{N_s+N_{su}} H(P_i^{s\to t}) + \sum_{i=1}^{N_{tu}} H(P_i^{t\to s}) \tag{2.8}$$

Adaptive Prototypical Classifier Learning

The goal of this section is to learn a better domain-aligned, discriminative feature encoder F and more importantly, a cosine classifier C that could achieve high accuracy on the target domain.

The cosine classifier C consists of weight vectors $\mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_{n_c}]$, where n_c denotes the total number of classes, and a temperature T. The output of C, $\frac{1}{T}\mathbf{W}^{\mathrm{T}}\mathbf{f}$ is fed into a softmax layer σ to obtain the final probabilistic output $\mathbf{p}(\mathbf{x}) = \sigma(\frac{1}{T}\mathbf{W}^{\mathrm{T}}\mathbf{f})$. With the availability of the labeled set $\mathcal{D}_s$, it is straightforward to train F and C with a standard cross-entropy loss for classification:

$$\mathcal{L}_{\text{cls}} = \mathbb{E}_{(\mathbf{x},y)\in\mathcal{D}_s}\mathcal{L}_{CE}(\mathbf{p}(\mathbf{x}), y) \tag{2.9}$$

However, since $\mathcal{D}_s$ is quite small under FUDA setting, only training with $\mathcal{L}_{\text{cls}}$ is hard to get a C with high performance on the target.

Adaptive Prototype-Classifier Update (APCU) Note that for C to classify samples correctly, the direction of a weight vector $\mathbf{w}_i$ needs to be representative of features of the corresponding class i. This indicates that the meaning of $\mathbf{w}_i$ coincide with the ideal cluster prototype of class i. We propose to use an estimate of the ideal cluster prototypes to update $\mathbf{W}$. Yet the computed $\{\mu_j^s\}$ and $\{\mu_j^t\}$ cannot be naively used for this purpose, not only because the correspondence between $\{\mathbf{w}_i\}$ and $\{\mu_j\}$ is unknown, but also the k-means result may contain very impure clusters, leading to non-representative prototypes.

We use the few-shot labeled data as well as samples with high-confident predictions to estimate the prototype for each class. Formally, we define $\mathcal{D}_s^{(i)} = \{\mathbf{x}|(\mathbf{x}, y) \in \mathcal{D}_s, y = i\}$ and denote by $\mathcal{D}_{su}^{(i)}$ and $\mathcal{D}_{tu}^{(i)}$ the set of samples with high-confident label i in source and target, respectively. With $\mathbf{p}(\mathbf{x}) = [\mathbf{p}(\mathbf{x})_1, \ldots, \mathbf{p}(\mathbf{x})_{n_c}]$, $\mathcal{D}_{su}^{(i)} = \{\mathbf{x}|\mathbf{x} \in \mathcal{D}_{su}, \mathbf{p}(\mathbf{x})_i > t\}$, where t is a confidence threshold; and similarly for $\mathcal{D}_{tu}^{(i)}$. Then the estimate of $\mathbf{w}_i$ from source and target domain can be computed as:

$$\hat{\mathbf{w}}_i^s = \frac{1}{|\mathcal{D}_{s^+}^{(i)}|}\sum_{\mathbf{x}\in\mathcal{D}_{s^+}^{(i)}} \boldsymbol{V}^{\boldsymbol{s}}(\mathbf{x}); \hat{\mathbf{w}}_i^t = \frac{1}{|\mathcal{D}_{tu}^{(i)}|}\sum_{\mathbf{x}\in\mathcal{D}_{tu}^{(i)}} \boldsymbol{V}^{\boldsymbol{t}}(\mathbf{x}) \tag{2.10}$$

where $\mathcal{D}_{s^+}^{(i)} = \mathcal{D}_s^{(i)} \cup \mathcal{D}_{su}^{(i)}$ and $\boldsymbol{V}(\mathbf{x})$ returns the representation in memory bank corresponding to $\mathbf{x}$.

With only few labeled samples in source, it is hard to learn a representative prototype shared across domains. Instead of directly employing a global prototype for a class i, we further propose to update $\mathbf{w}_i$ in an domain adaptive manner, with $\hat{\mathbf{w}}_i^s$ during early training stage and with $\hat{\mathbf{w}}_i^t$ at later stage. This is because that $\hat{\mathbf{w}}_i^s$ is more robust in early training stage due to the few labeled source samples, while $\hat{\mathbf{w}}_i^t$ would be more representative later for target domain to get better adaptation performance. Specifically, we use $|\mathcal{D}_{tu}^{(i)}|$ to determine whether $\hat{\mathbf{w}}_i^t$ is robust to use:

$$\mathbf{w}_i = \begin{cases} unit(\hat{\mathbf{w}}_i^s) & \text{if } |\mathcal{D}_{tu}^{(i)}| < t_w \\ unit(\hat{\mathbf{w}}_i^t) & \text{otherwise} \end{cases} \tag{2.11}$$

where $unit(\cdot)$ normalizes the input vector and t_w is a threshold hyper-parameter.

Table 2.1: Adaptation accuracy (%) comparison on 1-shot and 3-shots per class on the Office dataset.

Method	Office: Target Acc. on 1-shot / 3-shots						
	A→D	A→W	D→A	D→W	W→A	W→D	Avg
SO	27.5 / 49.2	28.7 / 46.3	40.9 / 55.3	65.2 / 85.5	41.1 / 53.8	62.0 / 86.1	44.2 / 62.7
MME [188]	21.5 / 51.0	12.2 / 54.6	23.1 / 60.2	60.9 / 89.7	14.0 / 52.3	62.4 / 91.4	32.3 / 66.5
CDAN [145]	11.2 / 43.7	6.2 / 50.1	9.1 / 65.1	54.8 / 91.6	10.4 / 57.0	41.6 / 89.8	22.2 / 66.2
SPL [236]	12.0 / 77.1	7.7 / 80.3	7.3 / 74.2	7.2 / 93.5	7.2 / 64.4	10.2 / 91.6	8.6 / 80.1
CAN [117]	25.3 / 48.6	26.4 / 45.3	23.9 / 41.2	69.4 / 78.2	21.2 / 39.3	67.3 / 82.3	38.9 / 55.8
MDDIA [113]	45.0 / 62.9	54.5 / 65.4	55.6 / 67.9	84.4 / 93.3	53.4 / 70.3	79.5 / 93.2	62.1 / 75.5
CDS [119]	33.3 / 57.0	35.2 / 58.6	52.0 / 67.6	59.0 / 86.0	46.5 / 65.7	57.4 / 81.3	47.2 / 69.3
DANN + ENT [69]	32.5 / 57.6	37.2 / 54.1	36.9 / 54.1	70.1 / 87.4	43.0 / 51.4	58.8 / 89.4	46.4 / 65.7
MME + ENT	37.6 / 69.5	42.5 / 68.3	48.6 / 66.7	73.5 / 89.8	47.2 / 63.2	62.4 / 95.4	52.0 / 74.1
CDAN + ENT	31.5 / 68.3	26.4 / 71.8	39.1 / 57.3	70.4 / 88.2	37.5 / 61.5	61.9 / 93.8	44.5 / 73.5
CDS + ENT	40.4 / 61.2	44.7 / 66.7	66.4 / 73.1	71.6 / 90.6	58.6 / 71.8	69.3 / 86.1	58.5 / 74.9
CDS + MME + ENT	39.4 / 61.6	43.6 / 66.3	66.0 / 74.5	75.7 / 92.1	53.1 / 73.0	70.9 / 90.6	58.5 / 76.3
CDS + CDAN + ENT	52.6 / 65.1	55.2 / 68.8	65.7 / 71.2	76.6 / 88.1	59.7 / 71.0	73.3 / 87.3	63.9 / 75.3
CDS / MME + ENT†	55.4 / 75.7	57.2 / 77.2	62.8 / 69.7	84.9 / 92.1	62.6 / 69.9	77.7 / 95.4	65.3 / 80.0
CDS / CDAN + ENT†	53.8 / 78.1	65.6 / 79.8	59.5 / 70.7	83.0 / 93.2	57.4 / 64.5	77.1 / 97.4	66.1 / 80.6
PCS (Ours)	**60.2** / **78.2**	**69.8** / **82.9**	**76.1** / **76.4**	**90.6** / **94.1**	**71.2** / **76.3**	**91.8** / 96.0	**76.6** / **84.0**
Improvement	**+4.8** / **+0.1**	**+4.2** / **+3.1**	**+9.7** / **+1.9**	**+5.7** / **+0.9**	**+8.6** / **+3.3**	**+14.1** / -1.4	**+10.5** / **+3.4**

† Two-stage training results reported in [119].

Mutual Information Maximization In order for the above unified prototype-classifier learning paradigm to work well, the network is desired to have enough confident predictions, e.g. $|\mathcal{D}^{(i)}| > t_w$, for all classes to get robust $\hat{\mathbf{w}}_i^s$ and $\hat{\mathbf{w}}_i^t$ for $i = 1, \ldots, n_c$. First, to promote the network to have diversified outputs over the dataset, we maximize the entropy of expected network prediction $\mathcal{H}(\mathbb{E}_{\mathbf{x}\in\mathcal{D}}[p(y|\mathbf{x};\theta)])$, where θ denotes learnable parameters in both F and C, and $\mathcal{D} = \mathcal{D}_s \cup \mathcal{D}_{su} \cup \mathcal{D}_{tu}$. Second, in order to get high-confident prediction for each sample, we leverage entropy minimization on the network output which has shown efficacy in label-scarce scenarios [86, 11]. These two desired behaviors turn out to be equivalent to maximizing the mutual information between input and output:

$$\mathcal{I}(y;\mathbf{x}) = \mathcal{H}(\mathbf{p}_0) - \mathbb{E}_{\mathbf{x}}[\mathcal{H}(p(y|\mathbf{x};\theta))], \tag{2.12}$$

where the prior distribution $\mathbf{p}_0$ is given by $\mathbb{E}_{\mathbf{x}}[p(y|\mathbf{x};\theta)]$, and the detailed derivation is presented in the supplementary material. We can get the objective as:

$$\mathcal{L}_{\text{MIM}} = -\mathcal{I}(y;\mathbf{x}) \tag{2.13}$$

PCS Learning for FUDA

The PCS learning framework performs in-domain prototypical contrastive learning, cross-domain instance-prototype self-supervised learning, and unified adaptive prototype-classifier

Table 2.2: Performance contribution of each part in PCS framework on Office.

Method	Office: Target Acc. on 1-shot / 3-shots						
	A→D	A→W	D→A	D→W	W→A	W→D	Avg
$\mathcal{L}_{\text{cls}}$	27.5 / 49.2	28.7 / 46.3	40.9 / 55.3	65.2 / 85.5	41.1 / 53.8	62.0 / 86.1	44.2 / 62.7
$+\mathcal{L}_{\text{InSelf}}$	39.0 / 55.6	38.6 / 55.1	47.2 / 68.5	71.7 / 89.4	50.9 / 68.4	65.1 / 90.6	52.1 / 71.3
$+\mathcal{L}_{\text{CrossSelf}}$	47.2 / 71.1	52.7 / 70.6	59.0 / 75.5	76.4 / 90.3	58.5 / 74.1	66.9 / 91.8	60.1 / 78.9
$+\mathcal{L}_{\text{MIM}}$	52.8 / 73.5	57.5 / 71.2	67.2 / 76.3	78.9 / 91.4	64.2 / 74.3	68.7 / 92.2	64.9 / 79.8
+APCU (PCS)	**60.2 / 78.2**	**69.8 / 82.9**	76.1 / **76.4**	**90.6 / 94.1**	**71.2 / 76.3**	**91.8 / 96.0**	**76.6 / 84.0**
PCS w/o MIM	59.0 / 75.9	58.6 / 76.5	**76.2 / 76.4**	87.8 / 93.2	68.7 / 74.7	89.8 / 95.0	73.5 / 82.0

learning. Together with APCU in Eq. 2.11, the overall learning objective is:

$$\begin{aligned}\mathcal{L}_{PCS} = \mathcal{L}_{\text{cls}} + \lambda_{\text{in}} \cdot \mathcal{L}_{\text{InSelf}} \\ + \lambda_{\text{cross}} \cdot \mathcal{L}_{\text{CrossSelf}} + \lambda_{\text{mim}} \cdot \mathcal{L}_{\text{MIM}}\end{aligned} \tag{2.14}$$

2.5 Experiments

Experimental Setting

Datasets. We evaluate our approach on four public datasets and choose labeled images in source domain based on previous work [119]. **Office** [186] is a real-world dataset for domain adaptation tasks. It contains 3 domains (Amazon, DSLR, Webcam) with 31 classes. Experiments are conducted with 1-shot and 3-shots source labels per class in this dataset. **Office-Home** [227] is a more difficult dataset than Office, which consists of 4 domains (Art, Clipart, Product, Real) in 65 classes. Following [119], we look into the settings with 3% and 6% labeled source images per class, which means each class has 2 to 4 labeled images on average. **VisDA-2017** [172] is a challenging simulation-to-real dataset containing over 280K images across 12 classes. We validate our model on settings with 0.1% and 1% labeled source images per class as suggested in [119]. **DomainNet** [171] is a large-scale domain adaptation benchmark. Since some domains and classes are noisy, we follow [188] and use a subset containing four domains (Clipart, Real, Painting, Sketch) with 126 classes. We show results on settings with 1-shot and 3-shots source labels on this dataset.

Implementation Details. We use ResNet-101 [94] (for DomainNet) and ResNet-50 (for other datasets) pre-trained on ImageNet [183] as our backbones. To enable a fair comparison with [119], we replaced the last FC layer with a 512-D randomly initialized linear layer. L2-normalizing are performed on the output features. We use k-means GPU implementation in faiss [115] for efficient clustering. We use SGD with momentum of 0.9, a learning rate of 0.01, a batch size of 64. More implementation details can be found in the supplementary material.

Table 2.3: Adaptation accuracy (%) comparison on 3% and 6% labeled samples per class on the Office-Home dataset.

Method	Office-Home: Target Acc. (%)												
	Ar →Cl	Ar →Pr	Ar →Rw	Cl →Ar	Cl →Pr	Cl →Rw	Pr →Ar	Pr →Cl	Pr →Rw	Rw →Ar	Rw →Cl	Rw →Pr	Avg
3% labeled source													
SO	24.4	38.3	43.1	26.4	34.7	33.7	27.5	26.5	42.6	41.2	29.0	52.3	35.0
MME [188]	4.5	15.4	25.0	28.7	34.1	37.0	25.6	25.4	44.9	39.3	29.0	52.0	30.1
CDAN [145]	5.0	8.4	11.8	20.6	26.1	27.5	26.6	27.0	40.3	38.7	25.5	44.9	25.2
MDDIA [113]	21.7	37.3	42.8	29.4	43.9	44.2	37.7	29.5	51.0	47.1	29.2	56.4	39.1
CAN [117]	17.1	30.5	33.2	22.5	34.5	36.0	18.5	19.4	41.3	28.7	18.6	43.2	28.6
CDS [119]	33.5	41.1	41.9	45.9	46.0	49.3	44.7	37.8	51.0	51.6	35.7	53.8	44.4
DANN + ENT [69]	19.5	30.2	38.1	18.1	21.8	24.2	31.6	23.5	48.1	40.7	28.1	50.2	31.2
MME + ENT	31.2	35.2	40.2	37.3	39.5	37.4	48.7	42.9	60.9	59.3	46.4	58.6	44.8
CDAN + ENT	20.6	31.4	41.2	20.6	24.9	30.6	33.5	26.5	56.7	46.9	29.5	48.4	34.2
CDS + ENT	39.2	46.1	47.8	49.9	50.7	54.1	48.0	43.5	59.3	58.6	44.3	59.3	50.1
CDS + MME + ENT	39.4	48.0	52.1	50.0	52.3	56.4	47.8	44.2	60.6	61.1	45.3	62.1	51.6
CDS + CDAN + ENT	43.8	55.5	60.2	51.5	56.4	59.6	51.3	46.4	64.5	62.2	52.4	70.2	56.2
CDS / MME + ENT†	41.7	49.4	57.8	51.8	52.3	55.9	54.3	46.2	69.0	65.6	52.2	68.2	55.4
CDS / CDAN + ENT†	37.7	49.2	56.5	49.8	51.9	55.9	50.0	42.3	68.1	63.1	48.7	67.5	53.4
PCS (Ours)	**42.1**	**61.5**	**63.9**	**52.3**	**61.5**	**61.4**	**58.0**	**47.6**	**73.9**	**66.0**	**52.5**	**75.6**	**59.7**
Improvement	-1.7	**+6.0**	**+6.1**	**+3.7**	**+5.1**	**+1.8**	**+3.7**	**+1.2**	**+4.9**	**+0.4**	**+0.1**	**+5.4**	**+3.5**
6% labeled source													
SO	28.7	45.7	51.2	31.9	39.8	44.1	37.6	30.8	54.6	49.9	36.0	61.8	42.7
MME [188]	27.6	43.2	49.5	41.1	46.6	49.5	43.7	30.5	61.3	54.9	37.3	66.8	46.0
CDAN [145]	26.2	33.7	44.5	34.8	42.9	44.7	42.9	36.0	59.3	54.9	40.1	63.6	43.6
MDDIA [113]	25.1	44.5	51.9	35.6	46.7	50.3	48.3	37.1	64.5	58.2	36.9	68.4	50.3
CAN [117]	20.4	34.7	44.7	29.0	40.4	38.6	33.3	21.1	53.4	36.8	19.1	58.0	35.8
CDS [119]	38.8	51.7	54.8	53.2	53.3	57.0	53.4	44.2	65.2	63.7	45.3	68.6	54.1
DANN + ENT [69]	22.4	32.9	43.5	23.2	30.9	33.3	33.2	26.9	54.6	46.8	32.7	55.1	36.3
MME + ENT	37.2	42.4	50.9	46.1	46.6	49.1	53.5	45.6	67.2	63.4	48.1	71.2	51.8
CDAN + ENT	23.1	35.5	49.2	26.1	39.2	43.8	44.7	33.8	61.7	55.1	34.7	67.9	42.9
CDS + ENT	42.9	55.5	59.5	55.2	55.1	59.1	54.3	46.9	68.1	65.7	50.6	71.5	57.0
CDS + MME + ENT	41.7	58.1	61.7	55.7	56.2	61.3	54.6	47.3	68.6	66.4	50.3	72.1	57.8
CDS + CDAN + ENT	45.4	60.4	65.5	54.9	59.2	63.8	55.4	49.0	71.6	66.6	54.1	75.4	60.1
CDS / MME + ENT†	44.1	51.6	63.3	53.9	55.2	62.0	56.5	46.6	70.9	67.7	54.7	74.7	58.4
CDS / CDAN + ENT†	39.0	51.3	63.1	51.0	55.0	63.6	57.8	45.9	72.8	65.8	50.4	73.5	57.4
PCS (Ours)	**46.1**	**65.7**	**69.2**	**57.1**	**64.7**	**66.2**	**61.4**	**47.9**	**75.2**	67.0	53.9	**76.6**	**62.6**
Improvement	**+0.7**	**+5.3**	**+3.7**	**+2.2**	**+5.5**	**+2.4**	**+3.6**	-1.1	**+2.4**	-0.7	-0.8	**+1.2**	**+2.5**

† Two-stage training results reported in [119].

Results on FUDA

Baselines. **SO** is a model only trained using the labeled source images. **CDAN** [145] and **MDDIA** [113] are both state-of-the-art methods in UDA with a domain classifier to perform domain alignment. **MME** [188] minimizes the conditional entropy of unlabeled target data with respect to the feature extractor and maximizes it with respect to the classifier. **CAN** [117] uses clustering information to contrast discrepancy of source and target domain. **CDS** [119] is a instance-based cross-domain self-supervised pre-training, which can be used for other domain adaptation methods and form *two-stage* methods, such as **CDS / CDAN** and **CDS / MME**. We re-implement CDS into an end-to-end version by adding losses in two stage together and tuning the weight for different losses. We also investigate the *one-stage* version of the methods above (**CDS + CDAN**, **CDS + MME**). Following [119], entropy minimization (**ENT**) on source is added to previous DA methods to obtain better baseline performance.

We compare the proposed PCS with state-of-the-art methods on FUDA (adaptation with

Table 2.4: Adaptation accuracy (%) comparison on 0.1% and 1% labeled samples per class on the VisDA-2017 dataset.

Method	VisDA: Target Acc. (%)	
	0.1% Labeled	1% Labeled
SO	47.9	51.4
MME [188]	55.6	69.4
CDAN [145]	58.0	61.5
MDDIA [113]	68.9	71.3
CAN [117]	51.3	57.2
CDS [119]	34.2	67.5
DANN + ENT [69]	44.5	50.2
MME + ENT	54.0	66.1
CDAN + ENT	57.7	58.1
CDS + ENT	49.8	75.3
CDS + ENT + MME	60.0	<u>78.3</u>
CDS / MME + ENT†	62.5	69.4
CDS / CDAN + ENT†	<u>69.0</u>	69.1
PCS (Ours)	**78.0**	**79.0**
Improvement	**+9.0**	**+0.7**

† Two-stage training results reported in [119].

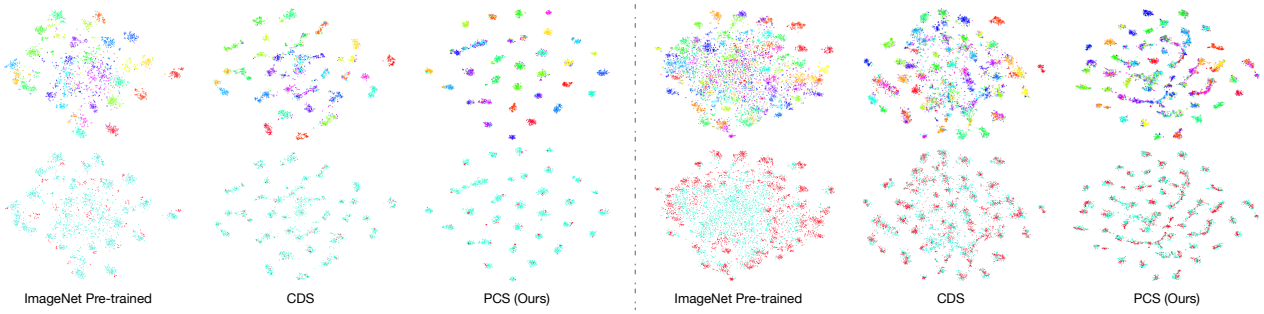


Figure 2.4: t-SNE visualization of ours and baselines on Office (left) and Office-Home (right). Top row: Coloring represents the class of each sample. Features with PCS are more discriminative than the ones with other methods. Bottom row: Cyan represents source features and Red represents target features. Feature from PCS are better-aligned between domains compared to other methods.

few source labels). Extensive experiments are conducted on Office, Office-Home, VisDA-2017 and DomainNet, with the results presented in Table 2.1, 2.3, 2.4, and 2.5, respectively. We

Table 2.5: Adaptation accuracy (%) comparison on 1-shot and 3-shots per class on the DomainNet dataset.

Method	DomainNet: Target Acc. (%)							
	R→C	R→P	R→S	P→C	P→R	C→S	S→P	Avg
1-shot labeled source								
SO	18.4	30.6	16.7	16.2	28.9	12.7	10.5	19.1
MME [188]	13.8	29.2	9.7	16.0	26.0	13.4	14.4	17.5
CDAN [145]	16.0	25.7	12.9	12.6	19.5	7.2	8.0	14.6
MDDIA [113]	18.0	30.6	15.9	15.4	27.4	9.3	10.2	18.1
CAN [117]	18.3	22.1	16.7	13.2	23.9	11.1	12.1	16.8
CDS [119]	16.7	24.4	11.1	14.1	15.9	13.4	19.0	16.4
CDS + ENT	21.7	30.1	18.2	17.4	20.5	18.6	22.7	21.5
CDS + MME + ENT	21.2	28.8	15.5	15.8	17.6	19.0	20.7	19.8
PCS (Ours)	**39.0**	**51.7**	**39.8**	**26.4**	**38.8**	**23.7**	**23.6**	**34.7**
Improvement	**+17.3**	**+21.1**	**+21.6**	**+9.0**	**+9.9**	**+4.7**	**+0.9**	**+13.2**
3-shots labeled source								
SO	30.2	44.2	25.7	24.6	49.8	24.2	23.2	31.7
MME [188]	22.8	46.5	14.5	25.1	50.0	20.1	24.9	29.1
CDAN [145]	30.0	40.1	21.7	21.4	40.8	17.1	19.7	27.3
MDDIA [113]	41.4	50.7	37.4	31.4	52.9	23.1	24.1	37.3
CAN [117]	28.1	33.5	25	24.7	46.9	23.3	20.1	28.8
CDS [119]	35.0	43.8	36.7	34.1	36.8	31.1	34.5	36.0
CDS + ENT	44.5	52.2	40.9	40.0	47.2	33.0	40.1	42.5
CDS + MME + ENT	43.8	54.9	41.1	38.9	45.9	32.8	38.7	42.3
PCS (Ours)	**45.2**	**59.1**	**41.9**	**41.0**	**66.6**	31.9	37.4	**46.1**
Improvement	**+0.7**	**+6.9**	**+0.8**	**+1.0**	**+13.7**	-0.9	-2.7	**+3.6**

can see that PCS outperforms the best state-of-the-arts in all the benchmarks, with large improvements: 10.5% and 3.4% on Office, 4.3% and 4.2% on Office-Home, 9.0% and 0.7% on VisDA, 13.2% and 3.6% on DomainNet. If we look at the result of each domain pair in each dataset (e.g. D $\rightarrow$ A in Office), PCS outperforms previous best in 47 out of 52 settings. Finally, as the number of labeled samples decreases, PCS shows larger performance improvements against the previous best methods, which demonstrates PCS is extremely beneficial in label-scarce adaptation scenarios.

Ablation Study and Analysis

Next, we investigate the effectiveness of each component in PCS on Office. Table 2.2 shows that adding each component contributes to the final results without any performance degradation. Comparing the last row in Table 2.2 and Table 2.1, we can see even without MIM, PCS still outperforms all previous methods.

We plot the learned features with t-SNE [152] on the DSLR-to-Amazon setting in Office,

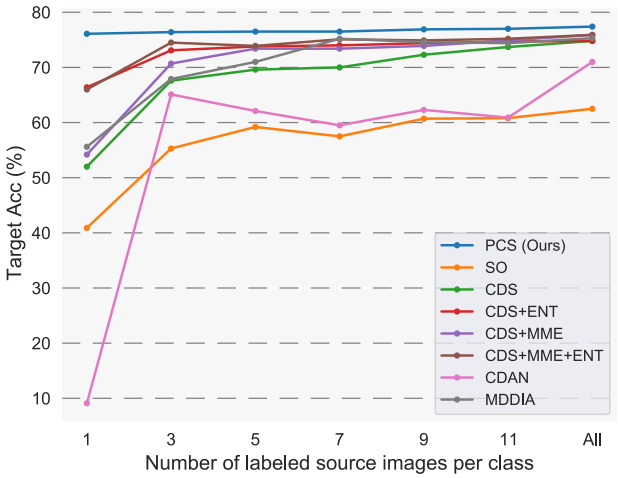


Figure 2.5: Sample efficiency comparison from DSLR to Amazon in Office dataset.

and Real-to-Clipart in Office-Home respectively in left and right of Figure 2.4. In the top row, the color represents the class of each sample; while in the bottom row, cyan represents source samples and red represents target samples. Compared to ImageNet pre-training and CDS, it qualitatively shows that PCS well clusters samples with the same class in the feature space; thus, PCS favors more discriminative features. Also, the features from PCS are more closely aggregated than ImageNet pre-training and CDS, which demonstrates that PCS learns a better semantic structure of the datasets.

Sample Efficiency

We compare our method with other state-of-the-art methods on Office dataset (DSLR as source and Amazon as target) with a varying number of source labels. From Figure 2.5, we can see that PCS outperforms all SOTA methods in all settings with different number of labeled samples. Moreover, our method is very label-efficient: a) For 1-shot image per class (31 labeled source images in total), PCS can achieve 76.1% accuracy. b) For the fully-labeled setting (498 labeled source images in total), PCS can achieve 77.4% accuracy. c) With **94%** less labeled source images, the performance degradation of our method is only **1.3%**. In short, *with less labeled source data, PCS outperforms other works by a larger margin.*

2.6 Conclusion

In this chapter, we investigated Few-shot Unsupervised Domain Adaptation where only few labeled samples are available in the source domain, and no labeled samples in the target domain. We proposed a novel Prototypical Cross-domain Self-supervised learning (PCS) framework that performs both in-domain and cross-domain prototypical self-supervised learning,

as well as adaptive prototpe-classifier learning. We perform extensive experiments on multiple benchmark datasets, which demonstrates the superiority of PCS over previous best methods. PCS sets a new state of the art for Few-shot Unsupervised Domain Adaptation.

Chapter 3

Generalization to Unseen Domains

3.1 Chapter Overview

This chapter targets the problem of domain generalization for semantic segmentation, where the test domains are not accessible during training. We propose a framework for this problem involving domain randomization with auxiliary domains and pyramid consistency at two levels: across styles and within an image.

3.2 Introduction

Simulation has spurred growing interests for training deep neural nets (DNNs) for computer vision tasks [248, 56, 124, 262]. This is partially due to the community's recent exploration to embodied vision [215, 288, 18], in which the perception has to be embodied and purposive for an agent to actively perceive and/or navigate through a physical environment [44, 56]. Moreover, training data generated by simulation is often low-cost and diverse, especially benefiting the tasks that otherwise need heavy human annotations (e.g. semantic segmentation [101, 268, 100]). Finally, in the case of autonomous driving, simulation can complement the insufficient coverage of real data by synthesizing rare events and scenes, such as construction sites, lane merges, and accidents. In summary, the promise of simulation is that one may conveniently acquire a large amount of labeled and diverse imagery from simulated environments. This scale is vital for training state-of-the-art deep convolutional neural networks (CNNs) with millions of parameters.

However, when we learn a semantic segmentation neural network from a synthetic dataset, its visual difference from real-world scenes often discounts its performance on real images. To mitigate the domain mismatch between simulation and the real world, existing work often resorts to **domain adaptation** [101, 99, 268], which aims to tailor the model for a particular target domain by jointly learning from the source synthetic data and the (often unlabeled) data of the target real domain. This setting is, unfortunately, very stringent. Take autonomous driving for instance. It is almost impossible for a car manufacturer to

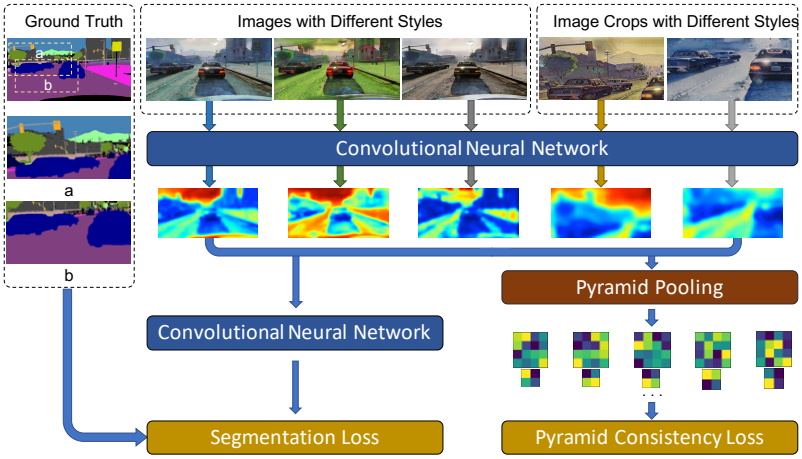


Figure 3.1: Domain randomization and pyramid consistency enforce the learned semantic segmentation network invariant to the change of domains. As a result, the semantic segmentation network can generalize to various domains, including those of real scenes.

know in advance under which domain (which city, what weather, day or night) the vehicle would be used.

In this chapter, we instead propose to harness the potential of simulation from a **domain generalization** manner [7, 132, 77, 215], without the need of accessing any target domain data in training and yet aiming to generalize well to multiple real-world target domains. We focus on the semantic segmentation of self-driving scenes, but the proposed method is readily applicable to similar tasks and scenarios. Our main idea is to randomize the labeled synthetic images to the styles of real images. We further enforce the semantic segmentation network to generate consistent predictions, in a pyramid form, over these domains. Our conjecture is that if the network is exposed to a sufficient number of domains in the training stage, it should *interpolate* well to new real-world target domains. In contrast, the domain adaptation work [101, 268, 99] can be seen as *extrapolating* from a single source domain to a single target domain.

Our approach comprises two key steps: domain randomization and consistency-enforced training, as illustrated in Figure 3.1. Unlike [219, 185], we do not require any control of the simulators for randomizing the source domain imagery. Instead, we leverage the recently advanced image-to-image translation [285] to transfer a source domain image to multiple styles, each dubbed an *auxiliary domain*. This has at least three advantages over manipulating the simulator. First, it enables us to select auxiliary domains from the real world. After all, our goal is to achieve good performance on real data. Second, we have a more concrete anticipation about the look of the randomized images as we view the auxiliary domains. Finally, the randomized images are naturally grouped according to the auxiliary domains. The last point facilitates us to devise effective techniques in the second step to train the networks in a domain-invariant way.

In the second step of our approach, we train a deep CNN for semantic segmentation with a pyramid consistency loss. If the network fits well to not only the synthetic source domain but also the auxiliary domains — synthetic images with the styles of real images, it may become invariant to domain changes to a certain degree and thus generalize well to real-world target domain(s). To ensure consistent performance across different training domains, we explicitly regularize the network's internal activations so that they do not deviate from each other too much for the stylized versions of the same source domain image. We find that it is vital to apply the regularization over average-pooled pyramids rather than the raw feature maps, probably because the pooled pyramid gives the network certain flexibility.

To the best of our knowledge, this is the first work to explore domain randomization for the semantic segmentation problem. Experiments show that the proposed approach gives rise to robust domain-invariant CNNs trained using synthetic images. It significantly outperforms the straightforward source-only baseline and the newly designed network [165], where the latter reduces the network's dependency on the training set by a hybrid of batch and instance normalizations. Our results are on par or even better than state-of-the-art domain adaptation results which are obtained by accessing the target data in training.

3.3 Related Work

We now discuss some related work on semantic segmentation, domain adaptation, domain generalization, domain randomization, and data augmentation.

Domain Adaptation for Semantic Segmentation. Until [101, 268] first studied the domain shift problem in semantic segmentation, most works in domain adaptation had focused on the task of image classification. After that, the problem subsequently became one of the tracks in the Visual Domain Adaptation Challenge (VisDA) 2017 [172] and started receiving increasing attention. Since then, adversarial training has been utilized in most of the following works [99, 36, 190, 270] for feature alignment. Most of these works were inspired by the unsupervised adversarial domain adaptation approach in [68] which shares similar idea with generative adversarial networks. One of their most important objectives is to learn domain-invariant representations by trying to deceive the domain classifier. Zhang *et al.* [268] perform segmentation adaptation by aligning label distributions both globally and across superpixels in an image. Recently, an unsupervised domain adaptation method has been proposed for semantic segmentation via class-balanced self-training [290]. Please refer to [269, Section 5] for a brief survey of other related works.

Domain Generalization In contrast to Domain Adaptation, where the network is tested on a known target domain, and the images in the target domain, although without labels, are accessible during the training process, Domain Generalization is tested on unseen domains [158, 67]. Current domain generalization researches mostly focus on the image classification problem. Image data is hard to manually divide into discrete domains, [80] devised a nonparametric formulation and optimization procedure to discover domains among both training and test data. [133] imposed Maximum Mean Discrepancy measure to align the

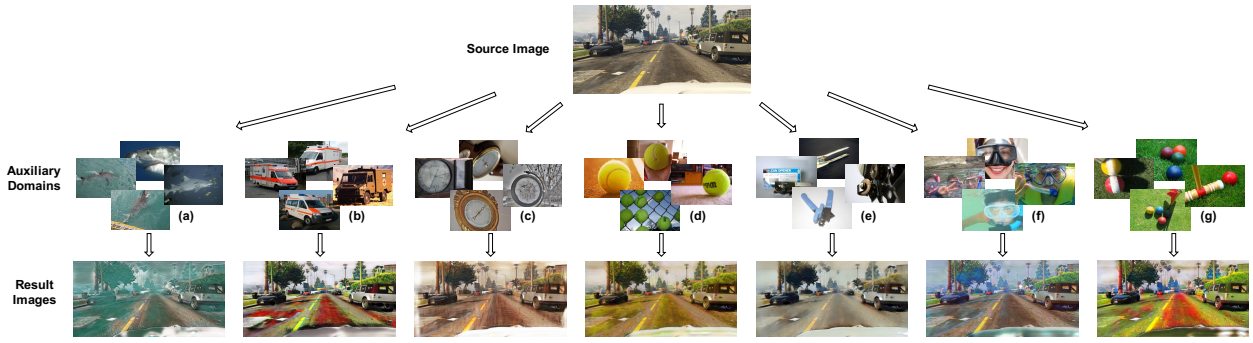


Figure 3.2: The domain randomization process. Top: an original synthetic image from the source domain; Mid: auxiliary image sets composed of ImageNet classes: (a) great white shark, (b) ambulance, (c) barometer, (d) tennis ball, (e) can opener, (f) snorkel, (g) tennis ball; Bottom: stylized images with same image content as the synthetic image and meanwhile corresponding styles of the ImageNet classes.

distributions among different domains and train the network with adversarial feature learning. [131] assigned a separate network duplication to each training domain during training and used the shared parameter for inference. [132] improved generalization performance by using a meta-learning approach on the split training sets.

Domain Randomization. Domain randomization (DR) is a complementary class of techniques for domain adaptation. Tobin *et al.* [215] introduced the concept of Domain Randomization. Their approach randomly varies the texture and color of the foreground object, the background image, the number of lights in the scene, the pose of the lights, the camera position, etc. The goal is to close the reality gap by generating synthetic data with sufficient variation that the network views real-world data as just another variation. Randomization in the visual domain has been used to directly transfer vision-based policies from simulation to the real world without requiring real images during training [185, 215]. DR has also been utilized to do object detection and 6D pose estimation [219, 173, 211]. All the above DR methods require modifying objects inside the simulation environment. We instead propose a different DR method which is orthogonal to all the aforementioned methods.

Data Augmentation. Data augmentation is the process of supplementing a dataset with similar data created from the information in that dataset, which is ubiquitous in deep learning. When dealing with images, it often includes the application of rotation, translation, blurring, and other modifications [40, 229, 191] to existing images that allow a network to better generalize [201]. In [130], a network is proposed to automatically generate augmented data by merging two or more samples from the same class. A Bayesian approach is proposed in [218] to generate data based on the distribution learned from the training set. In [48], simple transformations are used in the learned feature space to augment data. Counterexamples are considered to help data augmentation in [59]. Recently, AutoAugment has been

proposed to learn augmentation policies from data [43]. The type of domain randomization we proposed in this chapter can also be considered as a type of data augmentation.

3.4 Approach

The main idea of our approach is twofold, illustrated in Figure 3.1. The first part is **Domain Randomization with Stylization**: mapping the synthetic imagery to multiple auxiliary real domains (cf. Figure 3.2) in the training stage, such that, at the test stage, the target domain is not a surprise for the CNN model but merely another real domain. The second part is **Consistency-enforced Training**: enforcing pyramid consistency across domains and within an image to learn representations with better generalization ability.

Domain Randomization with Stylization

Keeping in mind that the target domain consists of real images, we randomly draw K real-life categories from ImageNet [45] for stylizing the synthetic images. Each category is called an auxiliary domain. We then use the image-to-image translation work [285] to map the synthetic images to each of the auxiliary domains. As a result, the training set is augmented to $K+1$ times the original size.

Figure 3.2 illustrates this procedure and some qualitative results. We can see that each auxiliary domain stylizes the synthetic images by different real-world elements. Meanwhile, the semantic content of the original image is retained at most parts of the images. Some edge-preserving methods [136] on style transfer may give rise to better results, and are left for future work.

A straightforward method is to train a CNN segmentation model using the augmented training set. Denote by $\mathcal{D}^k, k = 0, 1, \cdots, K$, the training domains, where $\mathcal{D}^0$ stands for the original source domain of synthetic images and $\mathcal{D}^k, k > 0$ the auxiliary domains. A synthetic image $I_n^0 \in \mathcal{D}^0$ has K stylized copies $I_n^k \in \mathcal{D}^k$ in the auxiliary domains, and yet they all share the same semantic segmentation map Y_n as the labels. The objective function for training a segmentation network $f(\cdot;\theta)$ is:

$$\min_{\theta} \quad \mathcal{L} := \frac{1}{Z} \sum_{n} \sum_{k=0}^{K} L\Big(Y_n, f(I_n^k;\theta)\Big), \tag{3.1}$$

where θ denotes the weights of the network, $L(\cdot,\cdot)$ is the mean of pixel-wise cross-entropy losses, and $Z = (K+1)\big|\mathcal{D}^0\big|$ is a normalization constant.

Our experiments (cf. Section 3.5) show that the network trained using this augmented training set $\mathcal{D}^0 \cup \mathcal{D}^1 \cdots \cup \mathcal{D}^K$ generalizes better to the unseen target domain than using the single source domain $\mathcal{D}^0$. Two factors may attribute to this result: 1) the training set is augmented in size and 2) the training set is augmented in style, especially in the styles closer to the real images. Despite being effective, this baseline method fails to track the multi-domain structure of the training set. We improve on it by the following.

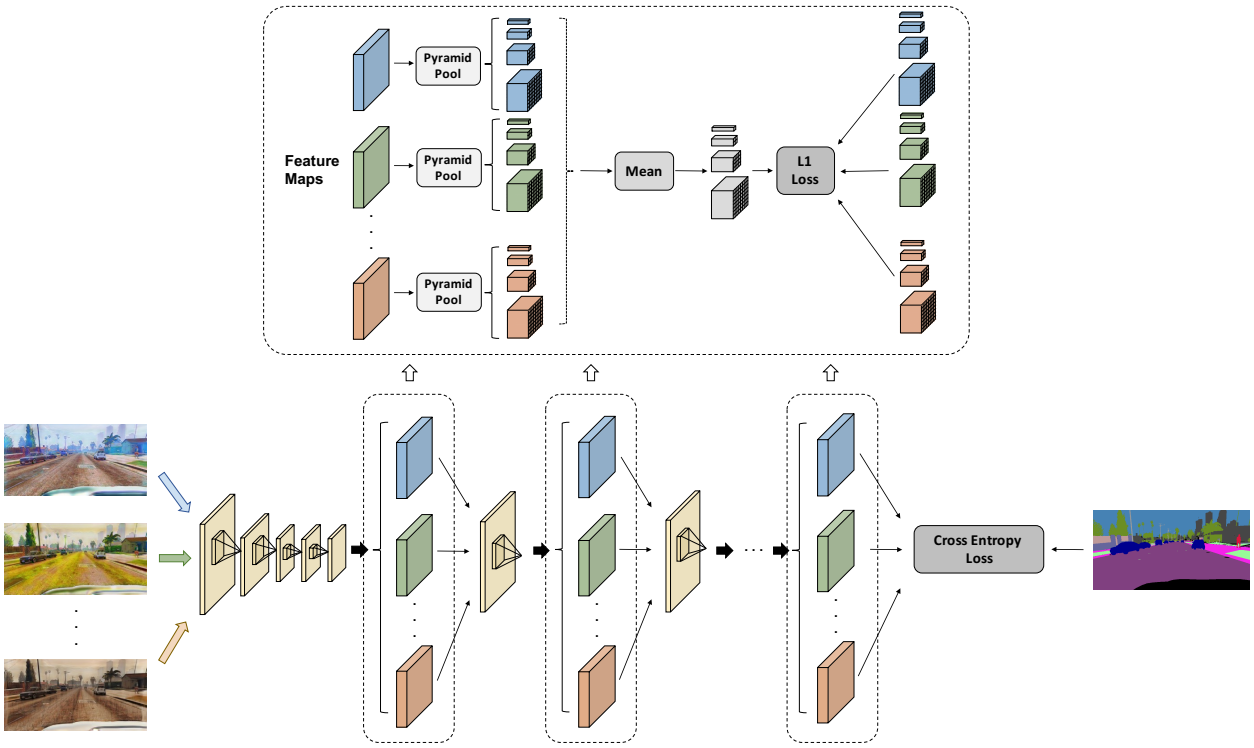


Figure 3.3: Pyramid Consistency across Domains. After feeding the images from different domains with the same content into the neural network, we impose the pyramid consistency loss on the activation maps at each of the last few layers (shown in blue, green and red).

Consistency-enforced Training

We aim to learn an image representation through the segmentation network that is domain-invariant for the semantic segmentation task. However, simply training the network with images from different domains (i.e., Eq. (3.1), the baseline method) has some problems: a) images from different domains may drive the network toward distinct representations, making the training process not converge well; b) even if the network fits well to the training domains, it could capture the idiosyncrasies of each individual and yet fail at interpolating between them or to the new target domain.

In order to tackle these caveats, we regularize the network by a consistency loss. The intuition is that if the network can generalize well, it should extract similar high-level semantic features and perform similar predictions for the images with the same content regardless of the styles. The consistency loss is simply imposed as the following:

$$\mathcal{R} := \sum_{n,k} \sum_{l \in \mathcal{P}} \lambda_l \, L_1 \Big(\overline{g_l(\mathcal{I}_n; \theta)}, \; g_l(I_n^k; \theta) \Big), \tag{3.2}$$

where l indexes an operator $g_l(\cdot;\theta)$ (e.g., average pooling) which maps a hidden layer's activations to a vector of smaller dimension, $\overline{g_l(\mathcal{I}_n;\theta)}$ denotes the target after each operation g_l (cf. Sections 3.4 and 3.4 for details), and L_1 is the ℓ_1 distance. We argue that, by doing so, the network can be better guided to find a generic and domain-invariant representation for the semantic segmentation task.

The design of the operators $g_l(\cdot;\theta), l \in \mathcal{P}$ is key to the overall performance. The obvious identity mapping — so the ℓ_1 distance is directly calculated over the hidden activations — does not work well in the experiments. One of the reasons is that it strictly requires the network to give about the same representations across different training domains, while some domains may be harder than the others to fit.

Pyramid consistency across domains

We find that the spatial pyramid pooling [96, 273, 129] serves as very effective operators $g_l(\cdot;\theta), l \in P$ in our context probably because it accommodates subtle differences of the network representations and meanwhile enables Eq. (3.2) to enforce the consistency at multiple scales. Pyramid pooling has been used in supervised visual understanding before, mostly as a part of the backbone networks. In this work, instead, we use the pooled features to define regularization losses for training the network. The pyramid consistency we consider is over the images of different styles but with the same semantic content.

Figure 3.3 illustrates our pyramid consistency scheme across the training domains. Consider a set of images $\mathcal{I}_n = \{I_n^k \mid k = 0, 1, \ldots, K\}$ of $K+1$ different styles with the same annotation Y_n and denote by $M_n^{l,k} \in \mathbb{R}^{C_l \times H_l \times W_l}$ the feature map of input I_n^k at layer l. Then, a spatial pyramid pooling operation is done on $M_n^{l,k}$. The spatial pyramid pooling operation is designed to fuse features under four different pyramid levels. First of all, a global average pooling is of the coarsest level that generates a single bin output. Each other pyramid levels separates the feature map into sub-regions evenly and performs average pooling inside each sub-region. In our design, we use 1×1, 2×2, 4×4 and 8×8 as the pyramid pooling scales, namely the spatial size of the outputs of the pyramid pooling. After the pooling, we squeeze and concatenate the output tensors into a tensor $P_n^{l,k} \in \mathbb{R}^{C_l \times \left(1+2^2+4^2+8^2\right)}$, which is much lower-dimensional than the original feature map $M_n^{l,k}$. For a pair of images $I_n^k, I_n^{k'} \in \mathcal{I}_n$, the network is expected to have similar understanding and thus similar high-level features in a deep layer l. Note that simply constraining $M_n^{l,k}$ and $M_n^{l,k'}$ to be the same is too strong and could easily lead to degraded performance. To save computation, we avoided pair-wise terms and instead use the mean of $P_n^{l,k}(k = 0, 1, ..., K)$ as the target value for the loss. Back to equation (3.2), we have $g_l(I_n^k;\theta) = P_n^{l,k}$, the target is the mean across domains $\overline{g_l(\mathcal{I}_n;\theta)} = \frac{1}{K+1}\sum_k P_n^{l,k}$, and the set $\mathcal{P} = \{l\}$ is the layers down deep of the network.

Pyramid consistency within an image

The pyramid consistency loss across the training domains can guide the network to learn style-invariant features so that it can generalize well to the unseen target domains with differ-

ent appearances. However, in many cases, style is not the only difference between domains. The view angles and parameters of cameras also lead to systematic domain mismatches in terms of the layout and scale of scenes. Take the *focal length* parameter for instance. With different focal lengths, the same objects may be of different scales as the fields of view vary.

In order to alleviate the issues above, we propose to further apply the pyramid consistency between random crops and full images. The idea is to artificially randomize the scale of the images and, therefore, guide the network to be robust to the domain gap incurred by the scene layouts and scales. Formally, following the notations in Section 3.4, each image I_n^k of size (H, W) is first randomly cropped at the same height-width ratio, with the top-left corner at (h_n^k, w_n^k) and with the height $\overline{h_n^k}$. Then the crop is scaled back to the full image size, denoted as C_n^k, and finally fed to the network. Denote by $M_n^{l,k}$ and $MC_n^{l,k} \in \mathbb{R}^{C_l \times H_l \times W_l}$ the feature maps of the image I_n^k and crop C_n^k at layer l, respectively. Denote by $\overline{M_n^{l,k}}$ the part of $M_n^{l,k}$ corresponding to the crop. When there is no significant padding through the layers, then $\overline{M_n^{l,k}}$ is of shape $C_l \times (\rho \cdot H_l) \times (\rho \cdot W_l)$, where $\rho = \overline{h_n^k}/h$.

We perform the spatial pyramid pooling on the cropped feature map $\overline{M_n^{l,k}}$ and the feature map $MC_n^{l,k}$ of the crop. The results are the same-size maps, $\overline{P_n^{l,k}}, PC_n^{l,k} \in \mathbb{R}^{C_l \times \left(1+2^2+4^2+8^2\right)}$. Back to Eq. (3.2), we have $g_l(I_n^k; \theta) = PC_n^{l,k}$ and the target vector is $\overline{g_l(\mathcal{I}_n; \theta)} = \overline{P_n^{l,k}}$.

3.5 Experiments and Results

In this section, we describe the experimental setup and present results on the semantic segmentation generalization by learning from synthetic data. Experimental analysis and comparison with other methods are also provided.

Experimental Settings

It should be emphasized that our experiment setting is different from domain adaptation. Since domain adaptation aims to achieve good performance on a particular target domain, it requires unlabeled target domain data during training and also (sometimes) uses some labeled target domain images for validation. In contrast, our model is trained without any target domain data and is tested on unseen domains.

Datasets. In our experiments, we use GTA [179] and SYNTHIA [181] as the source domains and a small subset of ImageNet [45] as well as datasets used in CycleGAN [285] as the auxiliary domains for "stylizing" the source domain images. We consider three target domains of real-world images, whose official validation sets are used as our test sets: Cityscapes [41], Berkeley Deep Drive Segmentation (BDDS) [260], and Mapillary [160].

GTA is a vehicle-egocentric image dataset collected in a computer game with pixel-wise semantic labels. It contains 24,966 images with the resolution 1914×1052. There are 19

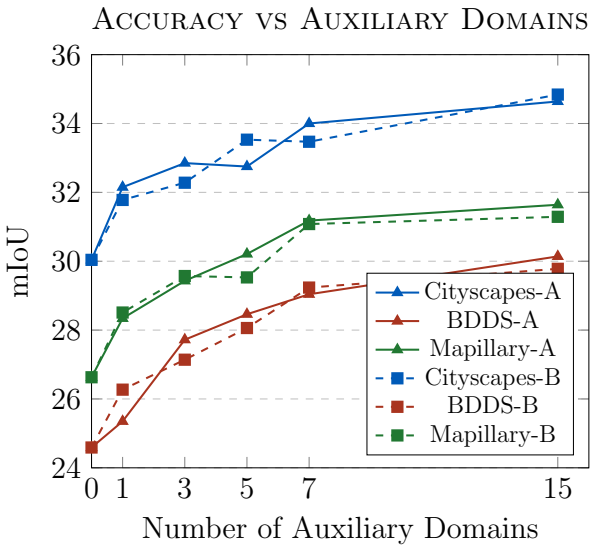


Figure 3.4: Accuracy of FCN8s-VGG16 with varying numbers of auxiliary domains. Two domain sets A and B are used. Models are trained on GTA and tested on Cityscapes, BDDS, and Mapillary.

classes which are compatible with other semantic segmentation datasets of outdoor scenes e.g. Cityscapes.

SYNTHIA is a large synthetic dataset with pixel-level semantic annotations. A subset, SYNTHIA-RAND-CITYSCAPES, is used in our experiments which contains 9,400 images with annotations compatible with Cityscapes.

Cityscapes contains vehicle-centric urban street images taken from some European cities. There are 5,000 images with pixel-wise annotations. The images have the resolution of 2048×1024 and are labeled into 19 classes.

BDDS contains thousands of real-world dashcam video frames with accurate pixel-wise annotations. It has a compatible label space with Cityscapes and the image resolution is 1280×720. The training, validation, and test sets contain 7,000, 1,000 and 2,000 images, respectively.

Mapillary contains street-level images collected from all around the world. The annotations contain 66 object classes, but only the 19 classes that overlap with Cityscapes and GTA are used in our experiments. It has a training set with 18,000 images and a validation set with 2,000 images.

Validation. To select a model for a particular real-world dataset $\mathcal{D}_R$ (e.g. Cityscapes), we randomly pick up 500 images from the training set of another real-world dataset $\mathcal{D}_{R'}$ (e.g. BDDS) as the validation set. This cross-validation is to imitate the following real-life scenarios. When we train a neural network from a randomized source domain without

knowing to which target domain it will be applied, we can probably collect a validation set which is as representative as possible of the potential target domains. Still take the car manufacturers for instance. A manufacturer may collect images of Los Angeles and NYC for the model selection while the cars will also be used in San Francisco and many other cities.

Evaluation. We evaluate the performance of a model on a test set using the standard PASCAL VOC intersection-over-union, i.e. IoU. The mean IoU (mIoU) is the mean of all IoU values over all categories. To measure the generalizability of a model M, we propose a new metric,

$$G_{perf}(M) = \mathbb{E}_{\mathcal{B}\in\mathcal{P}}\, mIoU(M, \mathcal{B}) \approx \frac{1}{L}\sum_{l} mIoU(M, \mathcal{B}_l)$$

where $\mathcal{B}$ is an unseen domain drawn from a distribution of all possible real-world domains $\mathcal{P}$, and L is the number of unseen test domains, which is 3 in our experiment setting.

Implementation Details In our experiments, we choose to use FCN [144] as our semantic segmentation network. To make it easier to compare with most of other methods, we use VGG-16 [202], ResNet-50, and ResNet-101 [94] as FCN backbones. The weights of the feature extraction layers in the networks are initialized from models trained on ImageNet [45]. We add the pyramid consistency loss across domains on the last 5 layers, with $\lambda = 0.2, 0.4, 0.6, 0.8, 1$, respectively. The pyramid consistency within an image is only added on the last layer. The network is implemented in PyTorch and trained with Adam optimizer [121] using a batch size of 32 for the baseline models and 8 for our models. Our machines are equipped with 8 NVIDIA Tesla P40 GPUs and 8 NVIDIA Tesla P100 GPUs.

Evaluation of Domain Randomization

In total, we use two sets of 15 auxiliary domains: A) 10 from ImageNet [45] and 5 from CycleGAN [285], and B) 15 from ImageNet with each domain corresponding to one semantic class in Cityscapes. Please see supplementary materials for additional auxiliary domains, including color augmentation as an auxiliary domain.

To evaluate our domain randomization method, we conduct experiments generalizing from GTA to Cityscapes, BDDS, and Mapillary with FCN8s-VGG16. We augment the training set with images from different numbers of auxiliary domains in both setting A and B, and show the result in Figure 3.4. As we can see from the plot, the accuracy increases with the number of auxiliary domains. The accuracy eventually saturates with the number of auxiliary domains. This is probably because 1) the 15 auxiliary domains are somehow sufficient to cover the appearance domain gap, and 2) as the number of images of the same content goes up, it is harder for the network to converge for the sake of the data scale and data variation.

Table 3.1: Performance contribution of each design.

Method	DR	PCD	PCI	mIoU		
				Cityscapes	BDDS	Mapillary
FCN				30.04	24.59	26.63
+DR	✓			34.64	30.14	31.64
+PCD	✓	✓		35.47	31.21	32.06
+PCI	✓		✓	35.12	30.87	32.12
All	✓	✓	✓	**36.11**	**31.56**	**32.25**

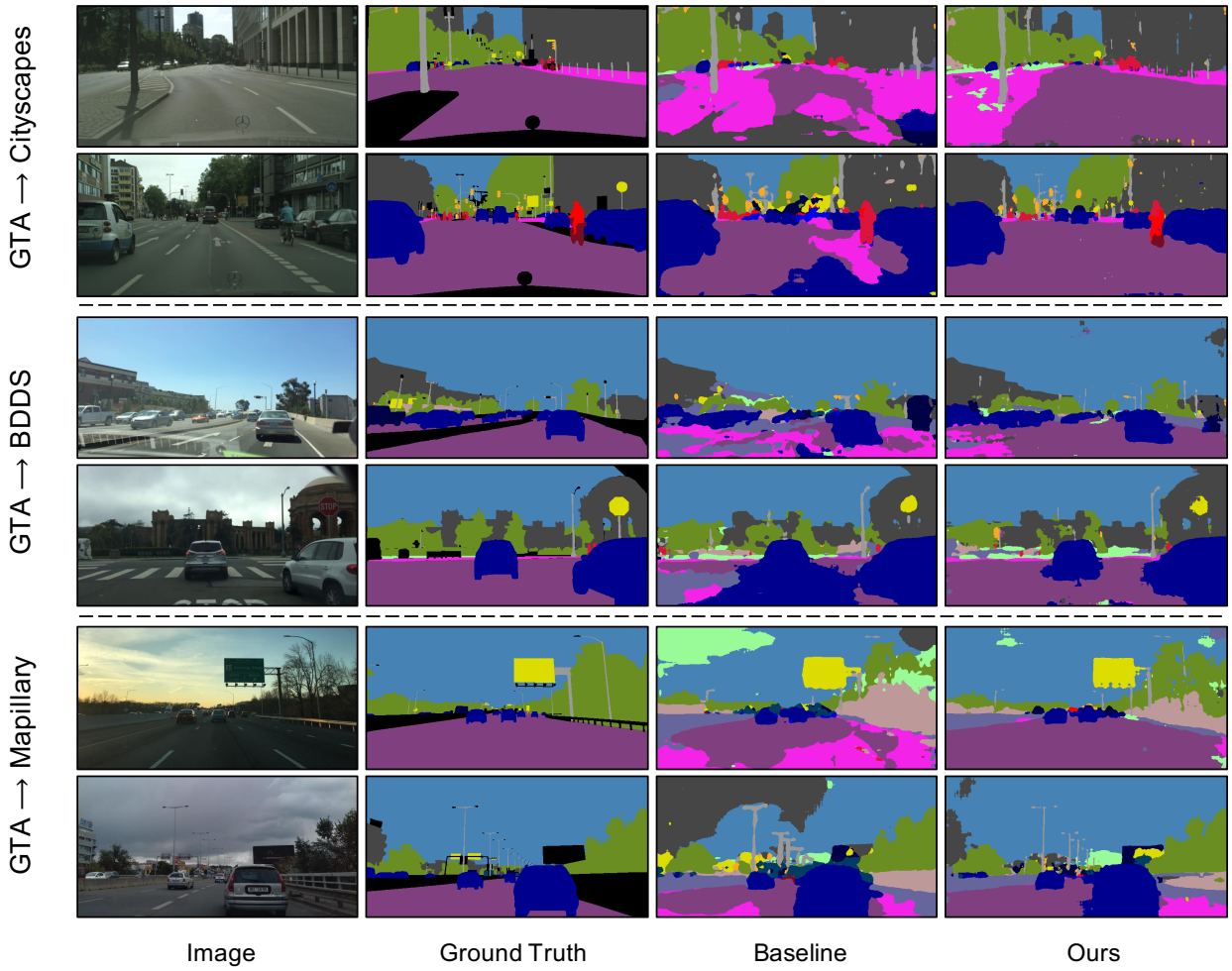


Figure 3.5: Qualitative semantic segmentation results of the generalization from GTA to Cityscapes, BDDS, and Mapillary.

Table 3.2: Domain generalization performance from (G)TA and (S)YNTHIA to (C)ityscapes, (B)DDS, and (M)apillary.

	VGG-16		ResNet-50		ResNet-101	
	NonAdapt	Ours	NonAdapt	Ours	NonAdapt	Ours
G→ C	30.04	36.11	32.45	37.42	33.56	42.53
G→ B	24.59	31.56	26.73	32.14	27.76	38.72
G→ M	26.63	32.25	25.66	34.12	28.33	38.05
G_{perf}	27.09	33.31	28.28	34.56	29.88	39.77
S → C	26.80	35.52	28.36	35.65	29.67	37.58
S → B	24.38	29.45	25.16	31.53	25.64	34.34
S → M	24.39	32.27	27.24	32.74	28.73	34.12
G_{perf}	25.34	32.41	26.92	33.31	28.01	35.35

Table 3.3: Comparison with other domain generalization methods.

Methods	Base Net	mIoU	mIoU↑
NonAdapt	ResNet-50	22.17	7.47
IBN-Net [165]		29.64	
NonAdapt	ResNet-50	32.45	4.97
Ours		**37.42**	

Ablation Study

Next, we study how each design in our approach influences the overall performance. The experiments are still adapting from GTA to the 3 tests with FCN8s-VGG16. Table 3.1 details the mIoU improvement on Cityscapes, BDDS and Mapillary by considering one more factor each time: Domain Randomization (DR), Pyramid Consistency across Domains (PCD) and within an Image (PCI). DR is a generic way to alleviate domain shift. In our case, it helps boost the performance from 30.04 to 34.64, from 24.59 to 30.14 and from 26.63 to 31.64, respectively for Cityscapes, BDDS and Mapillary. PCD and PCI further enhance the performance gains. By integrating all methods, our full approach finally reaches 36.11, 31.56 and 32.25 on Cityscapes, BDDS and Mapillary, respectively. Figure 3.5 showcases some examples of the semantic segmentation results on the 3 test sets.

Table 3.4: Adaptation from GTA to Cityscapes with FCN-8s.

Network	Method	Train w/ Tgt	Val on Tgt	mIoU	mIoU↑
VGG-19	NonAdapt Curriculum [268]	✓	✓	22.3 28.9	6.6
	NonAdapt CGAN [102]	✓	✓	NA 44.5	NA
VGG-16	NonAdapt FCN wld [101]	✓	✓	21.1 27.1	6.0
	NonAdapt CYCADA [99]	✓	✓	17.9 35.4	17.5
	NonAdapt LSD [190]	✓	✓	29.6 37.1	7.5
	NonAdapt ROAD [36]	✓	✓	21.9 35.9	14.0
	NonAdapt MCD [187]	✓	✓	24.9 28.8	3.9
	NonAdapt I2I [159]	✓	✓	NA 31.8	NA
	NonAdapt CBST-SP [290]	✓	✓	24.3 36.1	11.8
	NonAdapt DCAN [246]	✓	✓	27.8 36.2	8.4
	NonAdapt PTP [286]	✓	✓	30.0 38.1	8.1
	NonAdapt AdaptSegNet [220]	✓	✓	NA 35.0	NA
	NonAdapt BDL [137]	✓	✓	NA **41.3**	NA
	Non Adapt CLAN [151]	✓	✓	17.9 36.6	**18.7**
	NonAdapt DAM [107]	✓	✓	18.8 32.6	13.8
	NonAdapt **Ours**	✗	✓	30.0 38.6	8.6
	NonAdapt **Ours**	✗	✗	29.8 36.1	6.3

Generalization from GTA and SYNTHIA

Then, we conduct extensive experiments to evaluate the generalization ability of our proposed methods. Specifically, we tested *2 source domains*, GTA and SYNTHIA; *3 models with different backbone networks*, VGG-16, ResNet-50 and ResNet-101; *3 test sets*, Cityscapes, BDDS and Mapillary; and *2 sets of auxiliary domains* (cf. Section 3.5). The experiments with ResNet-50 are conducted with auxiliary domain set B, while the rest of the experiments

Table 3.5: Adaptation from SYNTHIA to Cityscapes with FCN-8s.

Network	Method	Train w/ Tgt	Val on Tgt	mIoU	mIoU↑
VGG-19	Non Adapt	✓	✓	22.0	7.0
	Curriculum [268]			29.0	
	Non Adapt	✓	✓	NA	NA
	CGAN [102]			41.2	
VGG-16	Non Adapt	✓	✓	17.4	2.8
	FCN Wld [101]			20.2	
	Non Adapt	✓	✓	25.4	10.8
	ROAD [36]			36.2	
	Non Adapt	✓	✓	26.8	9.3
	LSD [190]			36.1	
	Non Adapt	✓	✓	22.6	**12.8**
	CBST [290]			35.4	
	Non Adapt	✓	✓	27.8	8.4
	DCAN [246]			35.4	
	Non Adapt	✓	✓	NA	NA
	DAM [107]			30.7	
	Non Adapt	✓	✓	NA	9.3
	PTP [286]			34.2	
	Non Adapt	✓	✓	24.9	NA
	BDL [137]			**39.0**	
	Non Adapt	✗	✓	27.3	9.1
	Ours			36.4	
	Non Adapt	✗	✗	26.8	8.7
	Ours			35.5	

are with set A. The validation set and test set in each experiment are from different domains, e.g. using Cityscapes to select the model which will be evaluated on BDDS/Mapillary. The G_{perf} value of each model is computed and the results are shown in Table 3.2. We can see that the proposed techniques can greatly boost the generalizability by 5%∼12% of different models regardless of dataset combinations.

Then we compare our method with the only known existing state-of-the-art domain generalization method for semantic segmentation IBN-Net [165] under the generalization setting from GTA to Cityscapes. From the comparison shown in Table 3.3, we can see that our domain generalization method has better final performance. IBN-Net improves domain generalization by fine-tuning the ResNet building blocks. Our method would be complementary with theirs.

Adaptation from GTA and SYNTHIA

All experiments in the sections above are conducted in the domain generalization setting, where the validation set and the test set are from different domains. Now we conduct more experiments using the domain adaptation setting and compare our results with previous state-of-the-art works. Since most of the previous works conducted adaptation to Cityscapes with VGG backbone networks, we present the adaptation mIoU comparison on GTA $\rightarrow$ Cityscapes and SYNTHIA $\rightarrow$ Cityscapes in Table 3.4 and Table 3.5, leaving class-wise comparison details in the supplementary material. We can see that our method could outperform the state-of-the-art methods in both settings. Further, we should notice that the domain generalization performance of our method (last row) outperforms the adaptation performance of most other techniques. In addition, since our method is target domain-agnostic, no data is needed from the target domain, resulting in more extensive applicability.

3.6 Conclusion

In this chapter, we present a domain generalization approach for generalizing semantic segmentation networks from simulation to the real world without accessing any target domain data. We propose to randomize the synthetic images with auxiliary datasets and enforce pyramid consistency across domains and within an image. Finally, we experimentally validate our method on a variety of experimental settings, and show superior performance over state-of-the-art methods in both domain generalization and domain adaptation, which clearly demonstrates the effectiveness of our proposed method.

Chapter 4

Multi-source Domain Adaptation for Semantic Segmentation

4.1 Chapter Overview

This chapter targets the problem of multi-source domain adaptation for semantic segmentation, where there are multiple fully-labeled source domains available. We propose a generative adversarial domain aggregation network for this problem.

4.2 Introduction

Semantic segmentation assigns a semantic label (e.g. car, cyclist, pedestrian, road) to each pixel in an image. This computer vision kernel plays a crucial role in many applications, ranging from autonomous driving [72] and robotic control [103] to medical imaging [39] and fashion recommendation [109]. With the advent of deep learning, especially convolutional neural networks (CNNs), several end-to-end approaches have been proposed for semantic segmentation [144, 142, 282, 138, 259, 6, 273, 27, 235, 93]. Although these methods have achieved promising results, they suffer from some limitations. On the one hand, training these methods requires large-scale labeled data with pixel-level annotations, which is prohibitively expensive and time-consuming to obtain. For example, it takes about 90 minutes to label each image in the Cityscapes dataset [41]. On the other hand, they cannot well generalize their learned knowledge to new domains, because of the presence of *domain shift* or *dataset bias* [217, 242].

To sidestep the cost of data collection and annotation, unlimited amounts of synthetic labeled data can be created from simulators like CARLA and GTA-V [179, 181, 262], thanks to the progress in graphics and simulation infrastructure. To mitigate the gap between different domains, domain adaptation (DA) or knowledge transfer techniques have been proposed [169] with both theoretical analysis [10, 85, 149, 221] and algorithm design [164, 79, 111, 8, 77, 147, 100]. Besides the traditional task loss on the labeled source domain,

deep unsupervised domain adaptation (UDA) methods are generally trained with another loss to deal with domain shift, such as a discrepancy loss [147, 206, 205, 289], adversarial loss [83, 17, 139, 285, 17, 277, 184, 189, 104, 100, 276], reconstruction loss [77, 76, 16], *etc.* Current simulation-to-real DA methods for semantic segmentation [101, 268, 172, 36, 190, 270, 100, 64, 286, 246, 263] all focus on the single-source setting and do not consider a more practical scenario where the labeled data are collected from multiple sources with different distributions. Simply combining different sources into one source and directly employing single-source DA may not perform well, since images from different source domains may interfere with each other during the learning process [180].

Early efforts on multi-source DA (MDA) used shallow models [209, 63, 207, 61, 24, 62, 255, 192, 253, 208]. Recently, some multi-source deep UDA methods have been proposed which only focus on image classification [252, 272, 171]. Directly extending these MDA methods from classification to segmentation may not perform well due to the following reasons. (1) Segmentation is a structured prediction task, the decision function of which is more involved than classification because it has to resolve the predictions in an exponentially large label space [268, 220]. (2) Current MDA methods mainly focus on feature-level alignment, which only aligns high-level information. This may be enough for coarse-grained classification tasks, but it is obviously insufficient for fine-grained semantic segmentation, which performs pixel-wise prediction. (3) These MDA methods only align each source and target pair. Although different sources are matched towards the target, there may exist significant mis-alignment across different sources.

To address the above challenges, in this chapter we propose a novel framework, termed Multi-source Adversarial Domain Aggregation Network (MADAN), which consists of Dynamic Adversarial Image Generation, Adversarial Domain Aggregation, and Feature-aligned Semantic Segmentation. First, for each source, we generate an adapted domain using a Generative Adversarial Network (GAN) [83] with cycle-consistency loss [285], which enforces pixel-level alignment between source images and target images. To preserve the semantics before and after image translation, we propose a novel semantic consistency loss by minimizing the KL divergence between the source predictions of a pretrained segmentation model and the adapted predictions of a *dynamic segmentation model*. Second, instead of training a classifier for each source domain [252, 171], we propose *sub-domain aggregation discriminator* to directly make different adapted domains indistinguishable, and *cross-domain cycle discriminator* to discriminate between the images from each source and the images transferred from other sources. In this way, different adapted domains can be better aggregated into a more unified domain. Finally, the segmentation model is trained based on the aggregated domain, while enforcing feature-level alignment between the aggregated domain and the target domain.

In summary, our contributions are three-fold. (1) We propose to perform domain adaptation for semantic segmentation from multiple sources. To the best of our knowledge, this is the first work on multi-source structured domain adaptation. (2) We design a novel framework termed MADAN to do MDA for semantic segmentation. Besides feature-level alignment, pixel-level alignment is further considered by generating an adapted domain for

each source cycle-consistently with a novel dynamic semantic consistency loss. Sub-domain aggregation discriminator and cross-domain cycle discriminator are proposed to better align different adapted domains. (3) We conduct extensive experiments from synthetic GTA [179] and SYNTHIA [181] to real Cityscapes [41] and BDDS [260] datasets, and the results demonstrate the effectiveness of our proposed MADAN model.

4.3 Problem Setup

We consider the unsupervised MDA scenario, in which there are multiple labeled source domains $S_1, S_2, \cdots, S_M$, where M is number of sources, and one unlabeled target domain T. In the ith source domain S_i, suppose $X_i = \{\mathbf{x}_i^j\}_{j=1}^{N_i}$ and $Y_i = \{\mathbf{y}_i^j\}_{j=1}^{N_i}$ are the observed data and corresponding labels drawn from the source distribution $p_i(\mathbf{x}, \mathbf{y})$, where N_i is the number of samples in S_i. In the target domain T, let $X_T = \{\mathbf{x}_T^j\}_{j=1}^{N_T}$ denote the target data drawn from the target distribution $p_T(\mathbf{x}, \mathbf{y})$ without label observation, where N_T is the number of target samples. Unless otherwise specified, we have two assumptions: (1) homogeneity, i.e. $\mathbf{x}_i^j \in \mathbb{R}^d, \mathbf{x}_T^j \in \mathbb{R}^d$, indicating that the data from different domains are observed in the same image space but with different distributions; (2) closed set, i.e. $\mathbf{y}_i^j \in \mathcal{Y}, \mathbf{y}_T^j \in \mathcal{Y}$, where $\mathcal{Y}$ is the label set, which means that all the domains share the same space of classes. Based on covariate shift and concept drift [169], we aim to learn an adaptation model that can correctly predict the labels of a sample from the target domain trained on $\{(X_i, Y_i)\}_{i=1}^M$ and $\{X_T\}$.

4.4 Multi-source Adversarial Domain Aggregation Network

In this section, we introduce the proposed Multi-source Adversarial Domain Aggregation Network (MADAN) for semantic segmentation adaptation. The framework is illustrated in Figure 4.1, which consists of three components: Dynamic Adversarial Image Generation (DAIG), Adversarial Domain Aggregation (ADA), and Feature-aligned Semantic Segmentation (FSS). DAIG aims to generate adapted images from source domains to the target domain from the perspective of visual appearance while preserving the semantic information with a dynamic segmentation model. In order to reduce the distances among the adapted domains and thus generate a more aggregated unified domain, ADA is proposed, including Cross-domain Cycle Discriminator (CCD) and Sub-domain Aggregation Discriminator (SAD). Finally, FSS learns the domain-invariant representations at the feature-level in an adversarial manner. Table 4.1 compares MADAN with several state-of-the-art DA methods.

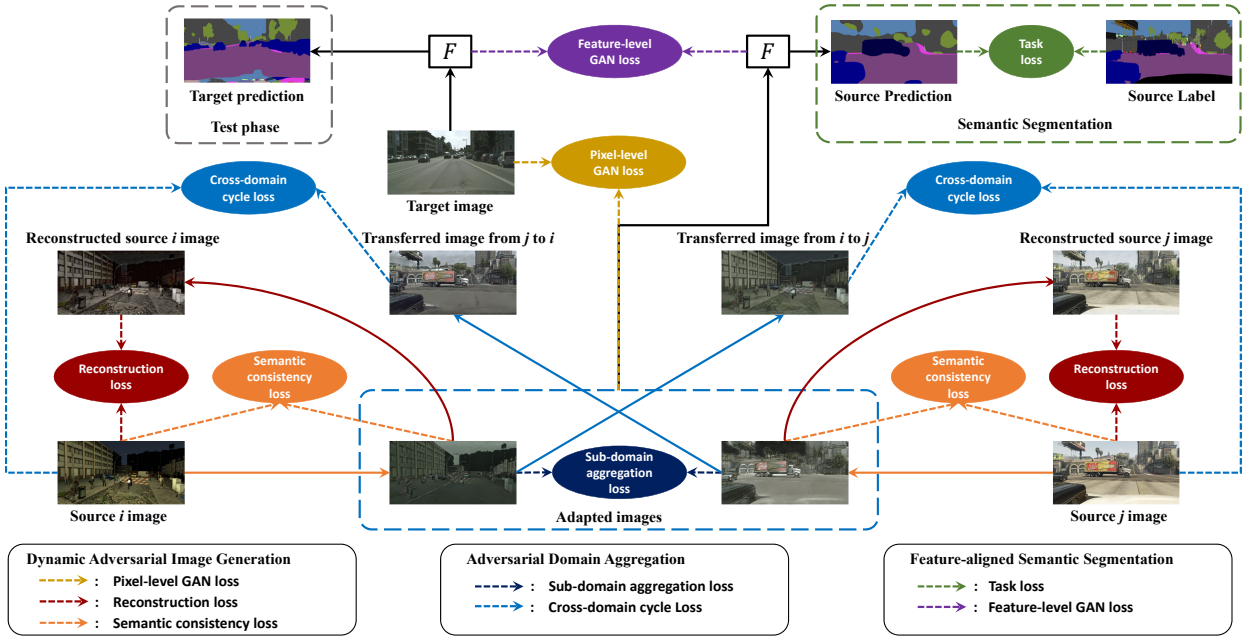


Figure 4.1: The framework of the proposed Multi-source Adversarial Domain Aggregation Network (MADAN). The colored solid arrows represent generators, while the black solid arrows indicate the segmentation network F. The dashed arrows correspond to different losses.

Dynamic Adversarial Image Generation

The goal of DAIG is to make images from different source domains visually similar to the target images, as if they are drawn from the same target domain distribution. To this end, for each source domain S_i, we introduce a generator $G_{S_i \to T}$ mapping to the target T in order to generate adapted images that fool D_T, which is a pixel-level adversarial discriminator. D_T is trained simultaneously with each $G_{S_i \to T}$ to classify real target images X_T from adapted images $G_{S_i \to T}(X_i)$. The corresponding GAN loss function is:

$$\mathcal{L}_{GAN}^{S_i \to T}(G_{S_i \to T}, D_T, X_i, X_T) = \mathbb{E}_{\mathbf{x}_i \sim X_i} \log D_T(G_{S_i \to T}(\mathbf{x}_i)) + \mathbb{E}_{\mathbf{x}_T \sim X_T} \log[1 - D_T(\mathbf{x}_T)]. \quad (4.1)$$

Since the mapping $G_{S_i \to T}$ is highly under-constrained [83], we employ an inverse mapping $G_{T \to S_i}$ as well as a cycle-consistency loss [285] to enforce $G_{T \to S_i}(G_{S_i \to T}(\mathbf{x}_i)) \approx \mathbf{x}$ and vice versa. Similarly, we introduce D_i to classify X_i from $G_{T \to S_i}(X_T)$, with the following GAN loss:

$$\mathcal{L}_{GAN}^{T \to S_i}(G_{T \to S_i}, D_i, X_T, X_i) = \mathbb{E}_{\mathbf{x}_i \sim X_i} \log[1 - D_i(\mathbf{x}_i)] + \mathbb{E}_{\mathbf{x}_t \sim X_T} \log D_i(G_{T \to S_i}(\mathbf{x}_t)). \quad (4.2)$$

The cycle-consistency loss [285] ensures that the learned mappings $G_{S_i \to T}$ and $G_{T \to S_i}$ are

Table 4.1: Comparison of the proposed MADAN model with several state-of-the-art domain adaptation methods. The full names of each property from the second to the last columns are pixel-level alignment, feature-level alignment, semantic consistency, cycle consistency, multiple sources, domain aggregation, one task network, and fine-grained prediction, respectively.

	pixel	feat	sem	cycle	multi	aggr	one	fine
ADDA [221]	✗	✓	–	–	✗	–	✓	✓
CycleGAN [285]	✓	✗	✗	✓	✗	–	✓	✗
PiexlDA [17]	✓	✗	✗	✗	✗	–	✓	✓
SBADA [184]	✓	✗	✓	✓	✗	–	✓	✗
GTA-GAN [189]	✓	✓	✗	✗	✗	–	✓	✗
DupGAN [104]	✓	✓	✓	✗	✗	–	✓	✗
CyCADA [100]	✓	✓	✓	✓	✗	–	✓	✓
DCTN [252]	✗	✓	–	–	✓	✗	✗	✗
MDAN [272]	✗	✓	–	–	✓	✗	✓	✗
MMN [171]	✗	✓	–	–	✓	✗	✗	✗
MADAN (ours)	✓	✓	✓	✓	✓	✓	✓	✓

cycle-consistent, thereby preventing them from contradicting each other, is defined as:

$$\begin{aligned}\mathcal{L}_{cyc}^{S_i \leftrightarrow T}(G_{S_i \to T}, G_{T \to S_i}, X_i, X_T) =& \mathbb{E}_{\mathbf{x}_i \sim X_i} \parallel G_{T \to S_i}(G_{S_i \to T}(\mathbf{x}_i)) - \mathbf{x}_i \parallel_1 + \\ & \mathbb{E}_{\mathbf{x}_T \sim X_T} \parallel G_{S_i \to T}(G_{T \to S_i}(\mathbf{x}_t)) - \mathbf{x}_t \parallel_1 .\end{aligned} \tag{4.3}$$

The adapted images are expected to contain the same semantic information as original source images, but the semantic consistency is only partially constrained by the cycle consistency loss. The semantic consistency loss in CyCADA [100] was proposed to better preserve semantic information. $\mathbf{x}_i$ and $G_{S_i \to T}(\mathbf{x}_i)$ are both fed into a segmentation model F_i pretrained on (X_i, Y_i). However, since $\mathbf{x}_i$ and $G_{S_i \to T}(\mathbf{x}_i)$ are from different domains, employing the same segmentation model, i.e. F_i, to obtain the segmentation results and then computing the semantic consistency loss may be detrimental to image generation. Ideally, the adapted images $G_{S_i \to T}(\mathbf{x}_i)$ should be fed into a network F_T trained on the target domain, which is infeasible since target domain labels are not available in UDA. Instead of employing F_i on $G_{S_i \to T}(\mathbf{x}_i)$, we propose to dynamically update the network F_A, which takes $G_{S_i \to T}(\mathbf{x}_i)$ as input, so that its optimal input domain (the domain that the network performs best on) gradually changes from that of F_i to F_T. We employ the task segmentation model F trained on the adapted domain as F_A, i.e. $F_A = F$, which has two advantages: (1) $G_{S_i \to T}(\mathbf{x}_i)$ becomes the optimal input domain of F_A, and as F is trained to have better performance on the target domain, the semantic loss after F_A would promote $G_{S_i \to T}$ to generate images that are closer to target domain at the pixel-level; (2) since F_A and F can share the parameters, no additional training or memory space is introduced, which is quite efficient. The proposed

dynamic semantic consistency (DSC) loss is:

$$\mathcal{L}_{sem}^{S_i}(G_{S_i\to T}, X_i, F_i, F_A) = \mathbb{E}_{\mathbf{x}_i\sim X_i} KL(F_A(G_{S_i\to T}(\mathbf{x}_i))||F_i(\mathbf{x}_i)), \tag{4.4}$$

where $KL(\cdot||\cdot)$ is the KL divergence between two distributions.

Adversarial Domain Aggregation

We can train different segmentation models for each adapted domain and combine different predictions with specific weights for target images [252, 171], or we can simply combine all adapted domains together and train one model [272]. In the first strategy, it is challenging to determine how to select the weights for different adapted domains. Moreover, each target image needs to be fed into all segmentation models at reference time, and this is rather inefficient. For the second strategy, since the alignment space is high-dimensional, although the adapted domains are relatively aligned with the target, they may be significantly misaligned with each other. In order to mitigate this issue, we propose adversarial domain aggregation to make different adapted domains more closely aggregated with two kinds of discriminators. One is the sub-domain aggregation discriminator (SAD), which is designed to directly make the different adapted domains indistinguishable. For S_i, a discriminator D_A^i is introduced with the following loss function:

$$\begin{aligned}\mathcal{L}_{SAD}^{S_i}(G_{S_1\to T}, \dots G_{S_i\to T}, \dots, G_{S_M\to T}, D_A^i) = \mathbb{E}_{\mathbf{x}_i\sim X_i} \log D_A^i(G_{S_i\to T}(\mathbf{x}_i)) + \\ \frac{1}{M-1}\sum\nolimits_{j\neq i} \mathbb{E}_{\mathbf{x}_j\sim X_j} \log[1 - D_A^i(G_{S_j\to T}(\mathbf{x}_j))].\end{aligned} \tag{4.5}$$

The other is the cross-domain cycle discriminator (CCD). For each source domain S_i, we transfer the images from the adapted domains $G_{S_j\to T}(X_j),\ j = 1, \cdots, M, j \neq i$ back to S_i using $G_{T\to S_i}$ and employ the discriminator D_i to classify X_i from $G_{T\to S_i}(G_{S_j\to T}(X_j))$, which corresponds to the following loss function:

$$\begin{aligned}\mathcal{L}_{CCD}^{S_i}(G_{T\to S_1}, \dots G_{T\to S_{i-1}}, G_{T\to S_{i+1}}, \dots, G_{T\to S_M}, G_{S_i\to T}, D_i) = \mathbb{E}_{\mathbf{x}_i\sim X_i} \log D_i(\mathbf{x}_i) + \\ \frac{1}{M-1}\sum\nolimits_{j\neq i} \mathbb{E}_{\mathbf{x}_j\sim X_j} \log[1 - D_i(G_{T\to S_i}((G_{S_j\to T}(\mathbf{x}_j))))].\end{aligned} \tag{4.6}$$

Please note that using a more sophisticated combination of different discriminators' losses to better aggregate the domains with larger distances might improve the performance. We leave this as future work and would explore this direction by dynamic weighting of the loss terms and incorporating some prior domain knowledge of the sources.

Feature-aligned Semantic Segmentation

After adversarial domain aggregation, the adapted images of different domains $X_i'(i = 1, \cdots, M)$ are more closely aggregated and aligned. Meanwhile, the semantic consistency

loss in dynamic adversarial image generation ensures that the semantic information, i.e. the segmentation labels, is preserved before and after image translation. Suppose the images of the unified aggregated domain are $X' = \bigcup_{i=1}^{M} X'_i$ and corresponding labels are $Y = \bigcup_{i=1}^{M} Y_i$. We can then train a task segmentation model F based on X' and Y with the following cross-entropy loss:

$$\mathcal{L}_{task}(F, X', Y) = -\mathbb{E}_{(\mathbf{x}',\mathbf{y})\sim(X',Y)} \sum_{l=1}^{L} \sum_{h=1}^{H} \sum_{w=1}^{W} \mathbb{1}_{[l=\mathbf{y}_{h,w}]} \log(\sigma(F_{l,h,w}(\mathbf{x}'))), \tag{4.7}$$

where L is the number of classes, H, W are the height and width of the adapted images, σ is the softmax function, $\mathbb{1}$ is an indicator function, and $F_{l,h,w}(\mathbf{x}')$ is the value of $F(\mathbf{x}')$ at index (l, h, w).

Further, we impose a feature-level alignment between X' and X_T, which can improve the segmentation performance during inference of X_T on the segmentation model F. We introduce a discriminator D_F to achieve this goal. The feature-level GAN loss is defined as:

$$\mathcal{L}_{feat}(F_f, D_{F_f}, X', X_T) = \mathbb{E}_{\mathbf{x}'\sim X'} \log D_{F_f}(F_f(\mathbf{x}')) + \mathbb{E}_{\mathbf{x}_T\sim X_T} \log[1 - D_{F_f}(F_f(\mathbf{x}_T))], \tag{4.8}$$

where $F_f(\cdot)$ is the output of the last convolution layer (i.e. a feature map) of the encoder in F.

MADAN Learning

The proposed MADAN learning framework utilizes adaptation techniques including pixel-level alignment, cycle-consistency, semantic consistency, domain aggregation, and feature-level alignment. Combining all these components, the overall objective loss function of MADAN is:

$$\begin{aligned}
&\mathcal{L}_{MADAN}(G_{S_1\to T}\cdots G_{S_M\to T}, G_{T\to S_1}\cdots G_{T\to S_M}, D_1\cdots D_M, D_A^1\cdots D_A^M, D_{F_f}, F) \\
&\quad= \sum\nolimits_i \Big[\mathcal{L}_{GAN}^{S_i\to T}(G_{S_i\to T}, D_T, X_i, X_T) + \mathcal{L}_{GAN}^{T\to S_i}(G_{T\to S_i}, D_i, X_T, X_i) \\
&\quad+ \mathcal{L}_{cyc}^{S_i\leftrightarrow T}(G_{S_i\to T}, G_{T\to S_i}, X_i, X_T) + \mathcal{L}_{sem}^{S_i}(G_{S_i\to T}, X_i, F_i, F) \\
&\quad+ \mathcal{L}_{SAD}^{S_i}(G_{S_1\to T}, \ldots G_{S_i\to T}, \ldots, G_{S_M\to T}, D_A^i) \\
&\quad+ \mathcal{L}_{CCD}^{S_i}(G_{T\to S_1}, \ldots G_{T\to S_{i-1}}, G_{T\to S_{i+1}}, \ldots, G_{T\to S_M}, G_{S_i\to T}, D_i)\Big] \\
&\quad+ \mathcal{L}_{task}(F, X', Y) + \mathcal{L}_{feat}(F_f, D_{F_f}, X', X_T).
\end{aligned} \tag{4.9}$$

The training process corresponds to solving for a target model F according to the optimization:

$$F^* = \arg\min_F \min_D \max_G \mathcal{L}_{MADAN}(G, D, F), \tag{4.10}$$

where G and D represent all the generators and discriminators in Eq. (4.9), respectively.

Table 4.2: Comparison with the state-of-the-art DA methods for semantic segmentation from GTA and SYNTHIA to Cityscapes. The best class-wise IoU and mIoU trained on the source domains are emphasized in bold (similar below).

Standards	Method	road	sidewalk	building	wall	fence	pole	t-light	t-sign	vegetation	sky	person	rider	car	bus	m-bike	bicycle	mIoU
Source-only	GTA	54.1	19.6	47.4	3.3	5.2	3.3	0.5	3.0	69.2	43.0	31.3	0.1	59.3	8.3	0.2	0.0	21.7
	SYNTHIA	3.9	14.5	45.0	0.7	0.0	14.6	0.7	2.6	68.2	68.4	31.5	4.6	31.5	7.4	0.3	1.4	18.5
	GTA+SYNTHIA	44.0	19.0	60.1	11.1	13.7	10.1	5.0	4.7	74.7	65.3	40.8	2.3	43.0	15.9	1.3	1.4	25.8
GTA-only DA	FCN Wld [101]	70.4	32.4	62.1	14.9	5.4	10.9	14.2	2.7	79.2	64.6	44.1	4.2	70.4	7.3	3.5	0.0	27.1
	CDA [268]	74.8	22.0	71.7	6.0	11.9	8.4	16.3	11.1	75.7	66.5	38.0	9.3	55.2	18.9	16.8	14.6	28.9
	ROAD [36]	85.4	31.2	78.6	**27.9**	**22.2**	21.9	23.7	11.4	80.7	68.9	48.5	14.1	78.0	23.8	8.3	0.0	39.0
	AdaptSeg [220]	**87.3**	29.8	78.6	21.1	18.2	22.5	21.5	11.0	79.7	71.3	46.8	6.5	**80.1**	26.9	10.6	0.3	38.3
	CyCADA [100]	85.2	37.2	76.5	21.8	15.0	23.8	22.9	21.5	80.5	60.7	50.5	9.0	76.9	28.2	4.5	0.0	38.7
	DCAN [246]	82.3	26.7	77.4	23.7	20.5	20.4	**30.3**	15.9	**80.9**	69.5	**52.6**	11.1	79.6	21.2	17.0	6.7	39.8
SYNTHIA-only DA	FCN Wld [101]	11.5	19.6	30.8	4.4	0.0	20.3	0.1	11.7	42.3	68.7	51.2	3.8	54.0	3.2	0.2	0.6	20.2
	CDA [268]	65.2	26.1	74.9	0.1	0.5	10.7	3.7	3.0	76.1	70.6	47.1	8.2	43.2	20.7	0.7	13.1	29.0
	ROAD [36]	77.7	30.0	77.5	9.6	0.3	**25.8**	10.3	15.6	77.6	**79.8**	44.5	16.6	67.8	14.5	7.0	23.8	36.2
	CyCADA [100]	66.2	29.6	65.3	0.5	0.2	15.1	4.5	6.9	67.1	68.2	42.8	14.1	51.2	12.6	2.4	20.7	29.2
	DCAN [246]	79.9	30.4	70.8	1.6	0.6	22.3	6.7	**23.0**	76.9	73.9	41.9	16.7	61.7	11.5	10.3	**38.6**	35.4
Source-combined DA	CyCADA [100]	82.8	35.8	78.2	17.5	15.1	10.8	6.1	19.4	78.6	77.2	44.5	15.3	74.9	17.0	10.3	12.9	37.3
Multi-source DA	MDAN [272]	64.2	19.7	63.8	13.1	19.4	5.5	5.2	6.8	71.6	61.1	42.0	12.0	62.7	2.9	12.3	8.1	29.4
	MADAN (Ours)	86.2	**37.7**	**79.1**	20.1	17.8	15.5	14.5	21.4	78.5	73.4	49.7	**16.8**	77.8	**28.3**	**17.7**	27.5	**41.4**
Oracle-Train on Tgt	FCN [144]	96.4	74.5	87.1	35.3	37.8	36.4	46.9	60.1	89.0	89.8	65.6	35.9	76.9	64.1	40.5	65.1	62.6

Table 4.3: Comparison with the state-of-the-art DA methods for semantic segmentation from GTA and SYNTHIA to BDDS. The best class-wise IoU and mIoU are emphasized in bold.

Standards	Method	road	sidewalk	building	wall	fence	pole	t-light	t-sign	vegetation	sky	person	rider	car	bus	m-bike	bicycle	mIoU
Source-only	GTA	50.2	18.0	55.1	3.1	7.8	7.0	0.0	3.5	61.0	50.4	19.2	0.0	58.1	3.2	**19.8**	0.0	22.3
	SYNTHIA	7.0	6.0	50.5	0.0	0.0	15.1	0.2	2.4	60.3	**85.6**	16.5	0.5	36.7	3.3	0.0	3.5	17.1
	GTA+SYNTHIA	54.5	19.6	64.0	3.2	3.6	5.2	0.0	0.0	61.3	82.2	13.9	0.0	55.5	16.7	13.4	0.0	24.6
GTA-only DA	CyCADA [100]	**77.9**	26.8	68.8	13.0	**19.7**	13.5	**18.2**	**22.3**	64.2	84.2	39.0	**22.6**	**72.0**	11.5	15.9	2.0	35.7
SYNTHIA-only DA	CyCADA [100]	55	13.8	45.2	0.1	0.0	13.2	0.5	10.6	63.3	67.4	22.0	6.9	52.5	10.5	10.4	13.3	24.0
Source-combined DA	CyCADA [100]	61.5	27.6	**72.1**	6.5	2.8	15.7	10.8	18.1	78.3	73.8	44.9	16.3	41.5	21.1	21.8	**25.9**	33.7
Multi-source DA	MDAN [272]	35.9	15.8	56.9	5.8	16.3	9.5	8.6	6.2	59.1	80.1	24.5	9.9	53.8	11.8	2.9	1.6	25.0
	MADAN (Ours)	60.2	**29.5**	66.6	**16.9**	10.0	**16.6**	10.9	16.4	**78.8**	75.1	**47.5**	17.3	48.0	**24.0**	13.2	17.3	**36.3**
Oracle-Train on Tgt	FCN [144]	91.7	54.7	79.5	25.9	42.0	23.6	30.9	34.6	81.2	91.6	49.6	23.5	85.4	64.2	28.4	41.1	53.0

4.5 Experiments

In this section, we first introduce the experimental settings and then compare the segmentation results of the proposed MADAN and several state-of-the-art approaches both quantitatively and qualitatively, followed by some ablation studies.

Experimental Settings

Datasets. In our adaptation experiments, we use synthetic GTA [179] and SYNTHIA [181] datasets as the source domains and real Cityscapes [41] and BDDS [260] datasets as the target domains.

Baselines. We compare MADAN with the following methods. **(1) Source-only**, i.e.

train on the source domains and test on the target domain directly. We can view this as a lower bound of DA. **(2) Single-source DA**, perform multi-source DA via single-source DA, including FCNs Wld [101], CDA [268], ROAD [36], AdaptSeg [220], CyCADA [100], and DCAN [246]. **(3) Multi-source DA**, extend some single-source DA method to multi-source settings, including MDAN [272]. For comparison, we also report the results of an oracle setting, where the segmentation model is both trained and tested on the target domain. For the source-only and single-source DA standards, we employ two strategies: (1) single-source, i.e. performing adaptation on each single source; (2) source-combined, i.e. all source domains are combined into a traditional single source. For MDAN, we extend the original classification network for our segmentation task.

Evaluation Metric. Following [101, 268, 100, 263], we employ mean intersection-over-union (mIoU) to evaluate the segmentation results. In the experiments, we take the 16 intersection classes of GTA and SYNTHIA, compatible with Cityscapes and BDDS, for all mIoU evaluations.

Implementation Details. Although MADAN could be trained in an end-to-end manner, due to constrained hardware resources, we train it in three stages. First, we train two CycleGANs (9 residual blocks for generator and 4 convolution layers for discriminator) [285] without semantic consistency loss, and then train an FCN F on the adapted images with corresponding labels from the source domains. Second, after updating F_A with F trained above, we generate adapted images using CycleGAN with the proposed DSC loss in Eq. (4.4) and aggregate different adapted domains using SAD and CCD. Finally, we train an FCN on the newly adapted images in the aggregated domain with feature-level alignment. The above stages are trained iteratively.

We choose to use FCN [144] as our semantic segmentation network, and, as the VGG family of networks is commonly used in reporting DA results, we use VGG-16 [202] as the FCN backbone. The weights of the feature extraction layers in the networks are initialized from models trained on ImageNet [45]. The network is implemented in PyTorch and trained with Adam optimizer [121] using a batch size of 8 with initial learning rate 1e-4. All the images are resized to 600×1080, and are then cropped to 400×400 during the training of the pixel-level adaptation for 20 epochs. SAD and CCD are frozen in the first 5 and 10 epochs, respectively.

Comparison with State-of-the-art

The performance comparisons between the proposed MADAN model and the other baselines, including source-only, single-source DA, and multi-source DA, as measured by class-wise IoU and mIoU are shown in Table 4.2 and Table 4.3. From the results, we have the following observations:

(1) The source-only method that directly transfers the segmentation models trained on the source domains to the target domain obtains the worst performance in most adaptation settings. This is obvious, because the joint probability distributions of observed images and labels are significantly different among the sources and the target, due to the presence

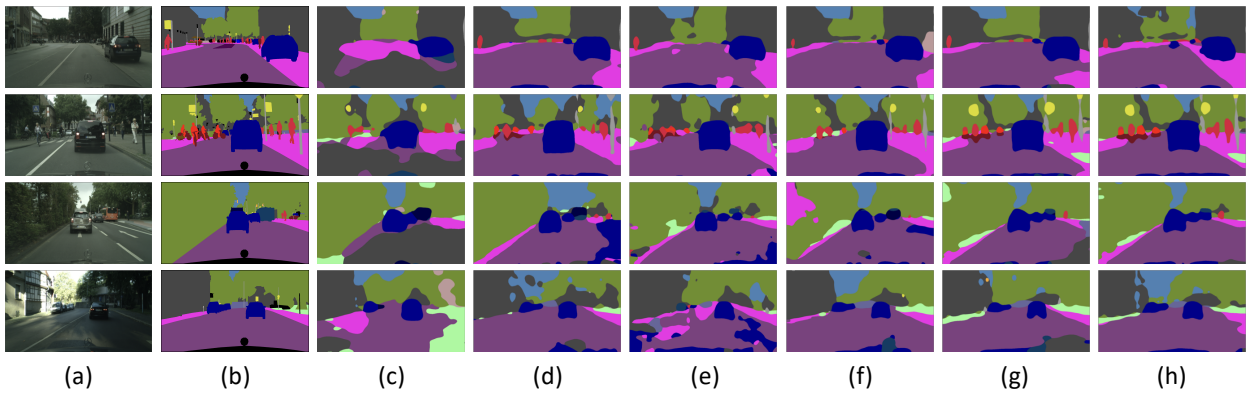


Figure 4.2: Qualitative semantic segmentation result from GTA and SYNTHIA to Cityscapes. From left to right are: (a) original image, (b) ground truth annotation, (c) source only from GTA, (d) CycleGANs on GTA and SYNTHIA, (e) +CCD+DSC, (f) +SAD+DSC, (g) +CCD+SAD+DSC, and (h) +CCD+SAD+DSC+Feat (MADAN).

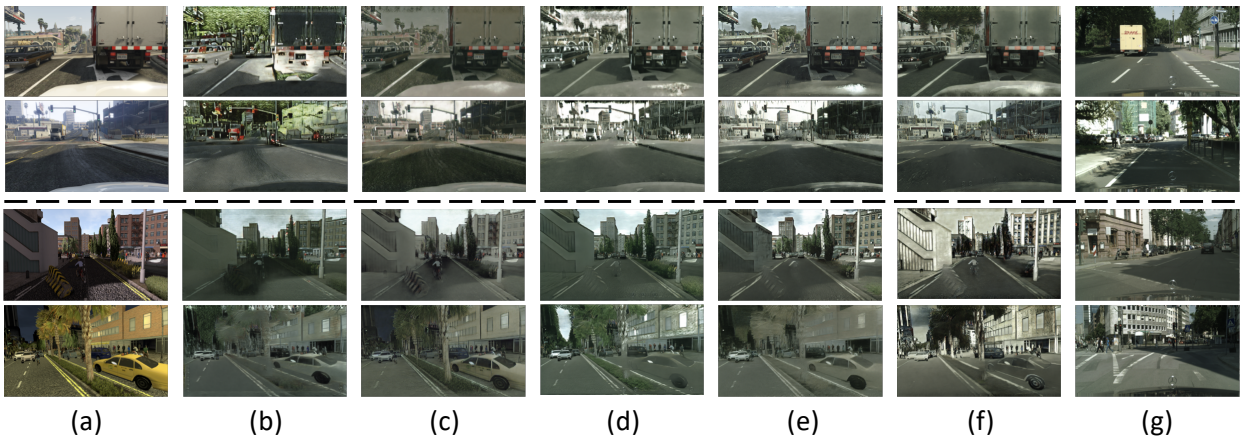


Figure 4.3: Visualization of image translation. From left to right are: (a) original source image, (b) CycleGAN, (c) CycleGAN+DSC, (d) CycleGAN+CCD+DSC, (e) CycleGAN+SAD+DSC, (f) CycleGAN+CCD+SAD+DSC, and (g) target Cityscapes image. The top two rows and bottom rows are GTA $\rightarrow$ Cityscapes and SYNTHIA $\rightarrow$ Cityscapes, respectively.

Table 4.4: Comparison between the proposed dynamic semantic consistency (DSC) loss in MADAN and the original SC loss in [100] on Cityscapes. The better mIoU for each pair is emphasized in bold.

Source	Method	road	sidewalk	building	wall	fence	pole	t-light	t-sign	vegettion	sky	person	rider	car	bus	m-bike	bicycle	mIoU
GTA	CycleGAN+SC	85.6	30.7	74.7	14.4	13.0	17.6	13.7	5.8	74.6	69.9	38.2	3.5	72.3	5.0	3.6	0.0	32.7
	CycleGAN+DSC	76.6	26.0	76.3	17.3	18.8	13.6	13.2	17.9	78.8	63.9	47.4	14.8	72.2	24.1	19.8	10.8	**38.1**
	CyCADA w/ SC	85.2	37.2	76.5	21.8	15.0	23.8	21.5	22.9	80.5	60.7	50.5	9.0	76.9	28.2	9.8	0.0	38.7
	CyCADA w/ DSC	84.1	27.3	78.3	21.6	18.0	13.8	14.1	16.7	78.1	66.9	47.8	15.4	78.7	23.4	22.3	14.4	**40.0**
SYNTHIA	CycleGAN+SC	64.0	29.4	61.7	0.3	0.1	15.3	3.4	5.0	63.4	68.4	39.4	11.5	46.6	10.4	2.0	16.4	27.3
	CycleGAN + DSC	68.4	29.0	65.2	0.6	0.0	15.0	0.1	4.0	75.1	70.6	45.0	11.0	54.9	18.2	3.9	26.7	**30.5**
	CyCADA w/ SC	66.2	29.6	65.3	0.5	0.2	15.1	4.5	6.9	67.1	68.2	42.8	14.1	51.2	12.6	2.4	20.7	29.2
	CyCADA w/ DSC	69.8	27.2	68.5	5.8	0.0	11.6	0.0	2.8	75.7	58.3	44.3	10.5	68.1	22.1	11.8	32.7	**31.8**

Table 4.5: Comparison between the proposed dynamic semantic consistency (DSC) loss in MADAN and the original SC loss in [100] on BDDS. The better mIoU for each pair is emphasized in bold.

Source	Method	road	sidewalk	building	wall	fence	pole	t-light	t-sign	vegettion	sky	person	rider	car	bus	m-bike	bicycle	mIoU
GTA	CycleGAN+SC	62.1	20.9	59.2	6.0	23.5	12.8	9.2	22.4	65.9	78.4	34.7	11.4	64.4	14.2	10.9	1.9	31.1
	CycleGAN+DSC	74.4	23.7	65.0	8.6	17.2	10.7	14.2	19.7	59.0	82.8	36.3	19.6	69.7	4.3	17.6	4.2	**32.9**
	CyCADA w/ SC	68.8	23.7	67.0	7.5	16.2	9.4	11.3	22.2	60.5	82.1	36.1	20.6	63.2	15.2	16.6	3.4	32.0
	CyCADA w/ DSC	70.5	32.4	68.2	10.5	17.3	18.4	16.6	21.8	65.6	82.2	38.1	16.1	73.3	20.8	12.6	3.7	**35.5**
SYNTHIA	CycleGAN+SC	50.6	13.6	50.5	0.2	0.0	7.9	0.0	0.0	63.8	58.3	21.6	7.8	50.2	1.8	2.2	19.9	21.8
	CycleGAN + DSC	57.3	13.4	56.1	2.7	14.1	9.8	7.7	17.1	65.5	53.1	11.4	1.4	51.4	13.9	3.9	8.7	**22.5**
	CyCADA w/ SC	49.5	11.1	46.6	0.7	0.0	10.0	0.4	7.0	61.0	74.6	17.5	7.2	50.9	5.8	13.1	4.3	23.4
	CyCADA w/ DSC	55	13.8	45.2	0.1	0.0	13.2	0.5	10.6	63.3	67.4	22.0	6.9	52.5	10.5	10.4	13.3	**24.0**

of domain shift. Without domain adaptation, the direct transfer cannot well handle this domain gap. Simply combining different source domains performs better than each single source, which indicates the superiority of multiple sources over single source despite the domain shift among different sources.

(2) Comparing source-only with single-source DA respectively on GTA and SYNTHIA, it is clear that all adaptation methods perform better, which demonstrates the effectiveness of domain adaptation in semantic segmentation. Comparing the results of CyCADA in single-source and source-combined settings, we can conclude that simply combining different source domains and performing single-source DA may result in performance degradation.

(3) MADAN achieves the highest mIoU score among all adaptation methods, and benefits from the joint consideration of pixel-level and feature-level alignments, cycle-consistency, dynamic semantic-consistency, domain aggregation, and multiple sources. MADAN also significantly outperforms source-combined DA, in which domain shift also exists among different sources. By bridging this gap, multi-source DA can boost the adaptation performance. On the one hand, compared to single-source DA [101, 268, 36, 220, 100, 246], MADAN utilizes

more useful information from multiple sources. On the other hand, other multi-source DA methods [252, 272, 171] only consider feature-level alignment, which may be enough for course-grained tasks, e.g. image classification, but is obviously insufficient for fine-grained tasks, e.g. semantic segmentation, a pixel-wise prediction task. In addition, we consider pixel-level alignment with a dynamic semantic consistency loss and further aggregate different adapted domains.

(4) The oracle method that is trained on the target domain performs significantly better than the others. However, to train this model, the ground truth segmentation labels from the target domain are required, which are actually unavailable in UDA settings. We can deem this performance as a upper bound of UDA. Obviously, a large performance gap still exists between all adaptation algorithms and the oracle method, requiring further efforts on DA.

Visualization. The qualitative semantic segmentation results are shown in Figure 4.2. We can clearly see that after adaptation by the proposed method, the visual segmentation results are improved notably. We also visualize the results of pixel-level alignment from GTA and SYNTHIA to Cityscapes in Figure 4.3. We can see that with our final proposed pixel-level alignment method (f), the styles of the images are close to Cityscapes while the semantic information is well preserved.

Ablation Study

First, we compare the proposed dynamic semantic consistency (DSC) loss in MADAN with the original semantic consistency (SC) loss in CyCADA [100]. As shown in Table 4.4 and Table 4.5, we can see that for all simulation to real adaptations, DSC achieves better results. After demonstrating its value, we employ the DSC loss in subsequent experiments.

Second, we incrementally investigate the effectiveness of different components in MADAN on both Cityscapes and BDDS. The results are shown in Table 4.6 and Table 4.7. We can observe that: (1) both domain aggregation methods, i.e. SAD and CCD, can obtain better performance by making different adapted domains more closely aggregated, while SAD outperforms CCD; (2) adding the DSC loss could further improve the mIoU score, again demonstrating the effectiveness of DSC; (3) feature-level alignments also contribute to the adaptation task; (4) the modules are orthogonal to each other to some extent, since adding each one of them does not introduce performance degradation.

Discussions

Computation cost. Since the proposed framework deals with a harder problem, i.e. multi-source domain adaptation, more modules are used to align different sources, which results in a larger model. In our experiments, MADAN is trained on 4 NVIDIA Tesla P40 GPUs for 40 hours using two source domains which is about twice the training time as on a single source. However, MADAN does not introduce any additional computation during inference, which is the biggest concern in real industrial applications, e.g. autonomous driving.

Table 4.6: Ablation study on different components in MADAN on Cityscapes. Baseline denotes using piexl-level alignment with cycle-consistency, +SAD denotes using the sub-domain aggregation discriminator, +CCD denotes using the cross-domain cycle discriminator, +DSC denotes using the dynamic semantic consistency loss, and +Feat denotes using feature-level alignment.

Method	road	sidewalk	building	wall	fence	pole	t-light	t-sign	vegettion	sky	person	rider	car	bus	m-bike	bicycle	mIoU
Baseline	74.9	27.6	67.5	9.1	10.0	12.8	1.4	13.6	63.0	47.1	41.7	13.5	60.8	22.4	6.0	8.1	30.0
+SAD	79.7	33.2	75.9	11.8	3.6	15.9	8.6	15.0	74.7	**78.9**	44.2	17.1	68.2	24.9	16.7	14.0	36.4
+CCD	82.1	36.3	69.8	9.5	4.9	11.8	12.5	15.3	61.3	54.1	49.7	10.0	70.7	9.7	19.7	12.4	33.1
+SAD+CCD	82.7	35.3	76.5	15.4	**19.4**	14.1	7.2	13.9	75.3	74.2	**50.9**	**19.0**	66.5	26.6	16.3	6.7	37.5
+SAD+DSC	83.1	36.6	78.0	**23.3**	12.6	11.8	3.5	11.3	75.5	74.8	42.2	17.9	72.2	27.2	13.8	10.0	37.1
+CCD+DSC	**86.8**	36.9	78.6	16.2	8.1	**17.7**	8.9	13.7	75.0	74.8	42.2	18.2	74.6	22.5	**22.9**	12.7	38.1
+SAD+CCD+DSC	84.2	35.1	78.7	17.1	18.7	15.4	**15.7**	**24.1**	77.9	72.0	49.2	17.1	75.2	24.1	18.9	19.2	40.2
+SAD+CCD+DSC+Feat	86.2	**37.7**	**79.1**	20.1	17.8	15.5	14.5	21.4	**78.5**	73.4	49.7	16.8	**77.8**	**28.3**	17.7	**27.5**	**41.4**

Table 4.7: Ablation study on different components in MADAN on BDDS.

Method	road	sidewalk	building	wall	fence	pole	t-light	t-sign	vegettion	sky	person	rider	car	bus	m-bike	bicycle	mIoU
Baseline	31.3	17.4	55.4	2.6	12.9	12.4	6.5	18.0	63.2	79.9	21.2	5.6	44.1	14.2	6.1	11.7	24.6
+SAD	58.9	18.7	61.8	6.4	10.7	17.1	20.3	17.0	67.3	83.7	21.1	6.7	66.6	22.7	4.5	14.9	31.2
+CCD	52.7	13.6	63.0	6.6	11.2	17.8	**21.5**	**18.9**	67.4	**84.0**	9.2	2.2	63.0	21.6	2.0	14.0	29.3
+SAD+CCD	61.6	20.2	61.7	7.2	12.1	**18.5**	19.8	16.7	64.2	83.2	25.9	7.3	66.8	22.2	5.3	14.9	31.8
+SAD+DSC	60.2	29.5	66.6	16.9	10.0	16.6	10.9	16.4	**78.8**	75.1	**47.5**	17.3	48.0	**24.0**	13.2	**17.3**	34.3
+CCD+DSC	61.5	27.6	72.1	6.5	12.8	15.7	10.8	18.1	78.3	73.8	44.9	16.3	41.5	21.1	21.8	15.9	33.7
+SAD+CCD+DSC	64.6	**38.0**	75.8	17.8	13.0	9.8	5.9	4.6	74.8	76.9	41.8	**24.0**	**69.0**	20.4	23.7	11.3	35.3
+SAD+CCD+DSC+Feat	**69.1**	36.3	**77.9**	**21.5**	**17.4**	13.8	4.1	16.2	76.5	76.2	42.2	16.4	56.3	22.4	**24.5**	13.5	**36.3**

On the poorly performing classes. There are two main reasons for the poor performance on certain classes (e.g. fence and pole): 1) lack of images containing these classes and 2) structural differences of objects between simulation images and real images (e.g. the trees in simulation images are much taller than those in real images). Generating more images for different classes and improving the diversity of objects in the simulation environment are two promising directions for us to explore in future work that may help with these problems.

4.6 Conclusion

In this chapter, we studied multi-source domain adaptation for semantic segmentation from synthetic data to real data. A novel framework, termed Multi-source Adversarial Domain Aggregation Network (MADAN), is designed with three components. For each source domain, we generated adapted images with a novel dynamic semantic consistency loss. Further we proposed a sub-domain aggregation discriminator and cross-domain cycle discriminator to

better aggregate different adapted domains. Together with other techniques such as pixel- and feature-level alignments as well as cycle-consistency, MADAN achieves 15.6%, 1.6%, 4.1%, and 12.0% mIoU improvements compared with best source-only, best single-source DA, source-combined DA, and other multi-source DA, respectively on Cityscapes from GTA and SYNTHIA, and 11.7%, 0.6%, 2.6%, 11.3% on BDDS. For further studies, we plan to investigate multi-modal DA, such as using both image and LiDAR data, to better boost the adaptation performance. Improving the computational efficiency of MADAN, with techniques such as neural architecture search, is another direction worth investigating.

Chapter 5

Multi-source Domain Adaptation for Textual Sentiment Analysis

5.1 Chapter Overview

This chapter targets the problem of multi-source domain adaptation for textual sentiment analysis, where there are multiple fully-labeled source domains available. We propose a curriculum cycle-consistent generative adversarial network for this problem.

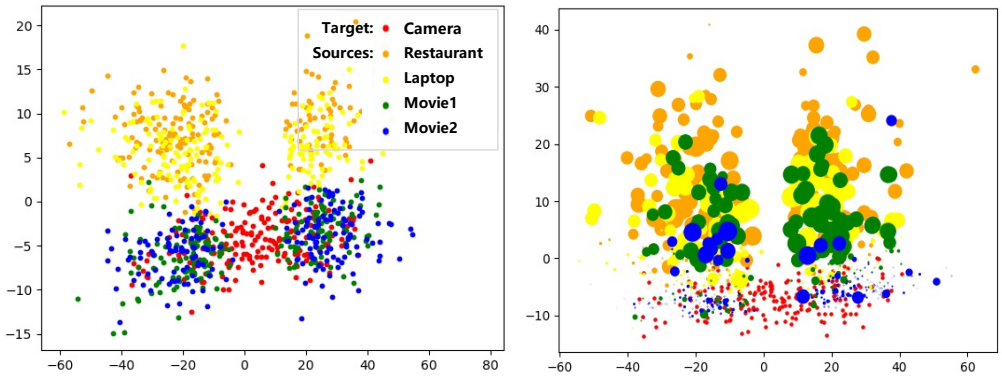
5.2 Introduction

The wide popularity of social networks and mobile devices enables human beings to reflect and share their opinions of the products and services they purchase online using text, images, and videos [281, 46, 81, 150, 82]. For example, when we plan to buy something, it is of high probability that we take a look at the comments on what others feel about this product. If the negative comments dominate the feedback, we might change our minds to a different brand. Sentiment analysis of user-generated large-scale multimedia data can not only help the customers to select what they want, but also prompt enterprises to improve the quality of their products and services [281, 37]. Among different multimedia modalities, text, the one focused on in this chapter, is the most direct and popular one [46].

Recent studies [271, 123, 74, 140, 254, 238, 237, 37, 12] have shown that deep neural networks (DNNs) achieve the state-of-the-art performance on textual sentiment analysis. However, training a DNN to maximize its capacity usually requires large-scale labeled data, which is expensive and time-consuming to obtain. One alternate solution is to train a DNN on a labeled source domain and transfer the DNN to the target domain. However, due to the presence of "domain shift" [217], i.e. the distribution differences between the source and target domains, direct transfer may result in significant performance degredation [223, 100, 263, 256]. Domain adaptation (DA) [169, 209, 126, 280, 274] that aims to minimize

	Accuracy on target
Single-best DANN	69.2
Source-combined DANN	67.0
C-CycleGAN	**73.8**

(a) Quantitative comparison



(b) Before adaptation **(c) After adaptation**

Figure 5.1: An example of *domain shift* in the multi-source scenario on the Reviews-5 dataset [261], where Camera (red points) is set as the target domain and the rest as source domains. (a) Naively combining multiple sources into one source and directly performing single-source domain adaptation (DANN [69]) does not guarantee better performance compared to just using the best individual source domain (69.2 vs. 67.0). The proposed C-CycleGAN framework achieves significant performance improvements over the source-trained model baselines (73.8 vs. 69.2). (b) and (c) visualize the representation space before and after adaptation. We can see clear domain shift across the sources and the target. After our domain adaptation, the source samples that are closer to the target domain (smaller points) are better aligned to the target domain (larger points indicate smaller sample weights).

the impact of domain shift provides an alternate solution by learning a model on the source domain with high transferability to the target domain.

Current DA methods for textual sentiment analysis mainly focus on the single-source unsupervised setting [141, 247], i.e. in which there is only one labeled source domain and one unlabeled target domain. While these unsupervised domain adaptation (UDA) methods perform well when the domain gap between the source and target domains is relatively small, they may fail when the domain gap is large or when there are multiple labeled source domains [90, 280], which is a more practical scenario. For example, if we have a target Kitchen domain, which may include reviews on cookbooks, bowls, and electric kettles, and three

source domains, books, cookware, and electronics, it is difficult to perfectly align each source and the target. Naive combination of different sources into one source and direct application of single-source UDA algorithms may lead to suboptimal results, because domain shift also exists across different sources, as shown in Figure 5.1. Sufficiently exploiting complementary information from different sources can allow for learning a better representation for the target domain, which calls for effective multi-source domain adaptation (MDA) techniques [209, 280].

Recently, some deep MDA approaches have been proposed for textual sentiment classification, most of which are based on adversarial learning, containing a pair of feature extractors and domain classifier (e.g. MDAN [272], MoE [90]). These methods mainly focus on extracting domain-invariant features of different domains, aligning each source and the target separately, or assigning weights to the source samples statically. Although they can obtain domain-invariant features among different domains, there are still some limitations. First, some discriminative features in the target domain that are related to sentiment might be missing. Since the shared feature extractor mainly aims to extract domain-invariant features by projecting both source samples and target samples to a lower-dimensional space, it may not include all sentiment-related features in the target domain. Second, some existing MDA methods separately align each source and the target and then combine the prediction results with known domain labels, which neglects the correlations of different source domains and different sub-domains even in each source. These methods would naturally fail when the domain labels of labeled source samples are not available. Finally, existing sampling-based methods mainly focus on selecting source samples that are closer to the target by training source selection models to calculate the weight of each sample (e.g. MDDA [278], CMSS [256]), which cannot reflect the varying optimal weighting during different training stages.

In this chapter, we propose a novel instance-level multi-source domain adaptation framework, named curriculum cycle-consistent generative adversarial network (C-CycleGAN), to address the above issues for textual sentiment classification. First, in order to encode all text instances in both source and target domains into a latent continuous representation space with minimal information loss, we introduce text reconstruction to better preserve information. Second, for the encoded source representations, we generate an intermediate domain to align the mixed source and target domains using a generative adversarial network (GAN) with cycle-consistency. To explore the importance of different source samples in a batch, we assign weights to them at instance-level with novel dynamic model-based and model-free weighting mechanisms. Finally, based on the adapted representations and corresponding source sentiment labels, we train a transferable task classifier. The sentiment loss of the classifier is also backpropagated to the source-to-target generator to preserve the sentiment information before and after generation. Extensive experiments are conducted on three benchmark datasets: Reviews-5 [261], Amazon benchmark [29], and Multilingual Amazon Reviews Corpus [34]. The results show that the proposed C-CycleGAN significantly outperforms the state-of-the-art DA methods for textual sentiment classification.

In summary, the contributions of this chapter are threefold:

(1) We propose a novel MDA method, named curriculum cycle-consistent generative adversarial network (C-CycleGAN), to minimize the domain shift between multiple source domains and the target domain. To the best of knowledge, we are the first to generate an intermediate representation domain with cycle-consistency and sentiment consistency for textual sentiment adaptation.

(2) We design novel instance-level model-based and model-free weighting mechanisms, which can update the sample weights dynamically. In this way, our framework does not require domain labels of samples, which allows it to exploit complementary information of all labeled source samples from different domains.

(3) We conduct extensive experiments on three benchmark datasets. As compared to the best baseline, the propsoed C-CycleGAN achieves 1.6%, 1.2%, and 13.4% improvements in average classification accuracy on Reviews-5, Amazon benchmark, and Multilingual Amazon Reviews Corpus, respectively.

5.3 Related Work

Textual Sentiment Analysis. Textual sentiment analysis, or opinion mining, aims to assess people's opinions, emotions, and attitudes from text towards entities such as products, services, or organizations [266]. The wide popularity of social networks such as product reviews, forum discussions, and WeChat, contributes to the rapid development of this task [266, 37]. Traditional sentiment analysis methods mainly focused on designing hand-crafted features [166, 156], which are fed into standard classifiers, such as SVM. Recent efforts on sentiment analysis are mainly based on DNNs [266, 238], which have shown great success in many natural language processing tasks. Some typical deep models that have been applied to sentiment analysis include Recursive Auto Encoder [204, 55, 174], Recursive Neural Tensor Network [203], Recurrent Neural Network (RNN) [213], Long short-term memory (LSTM) [97], Tree-LSTMs [212], RNN Encoder–Decoder [38], and BERT [47]. The above supervised learning methods usually require a large volume of labeled data for training [141, 37]. However, high-quality sentiment labels are often labor- and time-consuming to obtain. In this chapter, we employ a Bi-LSTM [97] as encoder and a multi-layer perceptron as classifier for the sentiment classification adaptation task.

Single-source UDA. Recent single-source UDA (SUDA) methods mainly employ deep learning architectures with two conjoined streams [289, 277]. One is trained on the labeled source data with a traditional task loss, such as cross-entropy loss for classification. The other aims to align the source and target domains to deal with the domain shift problem with different alignment losses, such as discrepancy loss, adversarial loss, self-supervision loss, *etc.* Discrepancy-based methods employ some distance measurements to explicitly minimize the discrepancy between the source and target domains on specific activation layers, such as maximum mean discrepancies [147, 234, 247], correlation alignment [206, 205, 289], and contrastive domain discrepancy [117]. Adversarial discriminative models usually employ a domain discriminator to adversarially align the extracted features between the source and

target domains by making them indistinguishable [69, 221, 26, 197, 220, 107, 127, 244]. Besides the domain discriminator, adversarial generative models also include a generative component to generate fake source or target data typically based on GAN [83] and its variants, such as CoGAN [139], SimGAN [199], and CycleGAN [285, 276, 100]. Self-supervision based methods incorporate auxiliary self-supervised learning tasks into the original task network to bring the source and target domains closer. The commonly used self-supervision tasks include reconstruction [77, 76, 33], image rotation prediction [210, 251], jigsaw prediction [21], and masking [228]. Although these methods achieve promising results for SUDA tasks, they suffer from significant performance decay when directly applied to MDA task.

Multi-source Domain Adaptation. Based on some theoretical analysis [10, 98], multi-source domain adaptation (MDA) aims to better deal with the scenario where training data are collected from multiple sources [209, 279]. The early shallow MDA methods mainly include two categories [209]: feature representation approaches [207, 61, 24, 62] and combination of pre-learned classifiers [253, 208]. Some special MDA cases are considered in recent shallow methods, such as incomplete MDA [50] and target shift [175].

Recently, some representative deep learning based MDA methods are proposed, such as multisource domain adversarial network (MDAN) [272], deep cocktail network (DCTN) [252], Mixture of Experts (MoE) [90], moment matching network (MMN) [171], multi-source adversarial domain aggregation network (MADAN) [279], multi-source distilling domain adaptation (MDDA) [278], and curriculum manager for source selection (CMSS) [256]. MDAN, DCTN, MoE, MMN, MADAN, and MDDA all require domain labels of source samples. MDDA and CMSS select source samples that are closer to the target domain with a static weighting mechanism, while the others do not consider the importance of different source samples. The MDA methods for textual sentiment classification, e.g. MDAN and MoE, only focus on extracting domain-invariant features, which may lose discriminative features of the target domain that are related to sentiment. Different from these methods, for the source samples, we generate an intermediate domain that is closer to the target domain with cycle-consistency and sentiment consistency. Further, we propose novel dynamic instance-level weighting mechanisms to assign weights to the source samples without the requirement of domain labels.

5.4 Proposed Approach

In this section, we formally define the MDA problem, give an overview of the proposed Curriculum CycleGAN (C-CycleGAN) framework, present each component of C-CycleGAN in detail, and finally introduce the joint learning process.

Problem Definition

We consider the multi-source unsupervised domain adaptation setup for textual sentiment classification, under the *covariate shift* assumption [169]. Assuming access to k source do-

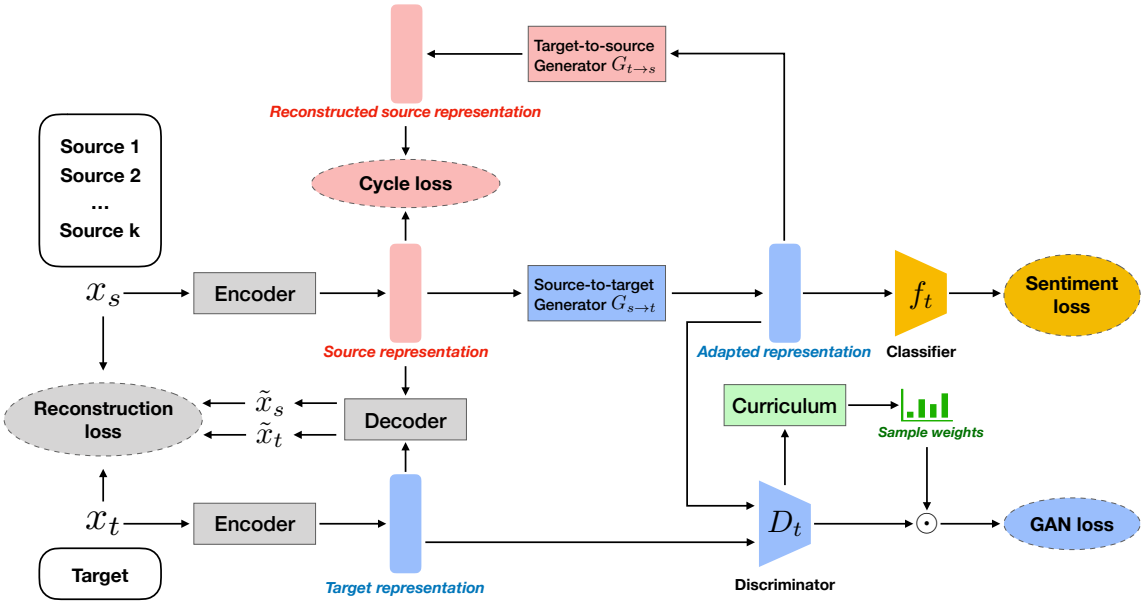


Figure 5.2: Illustration of the proposed C-CycleGAN framework. A text encoder is first pre-trained with a reconstruction loss to encode all text instances from the source and target domains into a latent continuous representation space (gray). Then the model is jointly trained using the cycle-consistency loss (pink), the curriculum GAN loss (**blue**), and the sentiment classification loss (yellow). We depict here the model-free curriculum (green) for sample weighting.

mains with labeled training data, denoted by $\{\mathcal{S}_i\}_{i=1}^{k}$, where each domain $\mathcal{S}_i$ contains a set of examples drawn from a joint distribution $p^{(S_i)}(\mathrm{x}, \mathrm{y})$ on the input space $\mathcal{X}$ and the output space $\mathcal{Y}$, we seek to learn a sentiment classifier $f : \mathcal{X} \to \mathcal{Y}$ that is transferable to a target domain $\mathcal{T}$, where only unlabeled data is available.

Overview

Our model bridges the domain gap by generating an intermediate domain using CycleGAN [285] trained with a learned curriculum (C-CycleGAN). As shown in Figure 5.2, the proposed framework has three primary components:

Pre-trained Text Encoder: Encode texts from source and target domains into a semantic-preserving latent continuous representation space $\mathcal{Z}$. This module is pre-trained using a *seq2seq*-based text autoencoder in an unsupervised fashion.

Intermediate Domain Generator: Generate an intermediate domain to align the multiple sources and the target. At the core of this component is a *curriculum cycle-consistent generative adversarial network*, which employs a domain adversarial loss for distributional alignment, and use cycle-consistency to prevent *mode collapse*. To deal with the varied

relevance of the mixed-source instances to the target domain at a specific training stage, we learn a curriculum to dynamically assign weights to source samples based on their proximity to the target domain distribution.

Task Classifier: Train the sentiment classifier based on the adapted representations in the intermediate domain and corresponding sentiment labels in the source domains.

Pre-trained Text Encoder

We use seq2seq-based text reconstruction to pre-train our text encoder, in order to obtain a semantic-preserving latent representation space. Let x denote a sequence of tokens $w_1, ..., w_L$, where L is the sequence length. The reconstruction process can be summarized as the following *encoding-decoding* paradigm:

$$\mathbf{z} = \texttt{Enc}(x; \boldsymbol{\theta}); \quad \tilde{x} = \texttt{Dec}(\mathbf{z}, x; \boldsymbol{\psi}) \tag{5.1}$$

where $\mathbf{z}$ is the text representation. We use a bidirectional LSTM (Bi-LSTM) [97] as the encoder, and obtain the representation $\mathbf{z}$ of an input sequence by concatenating the last states of forward LSTM and backward LSTM. A unidirectional LSTM then reconstructs x autoregressively conditioned on $\mathbf{z}$. At each time step of generation, we randomly sample from the ground-truth token and the generated token as input for the next token prediction. The overall reconstruction loss over both source and target domain data can thus be written as:

$$\mathcal{L}_{\text{rec}} = \mathbb{E}_{x \sim \mathcal{X}_S \cup \mathcal{X}_T} \Big[-\frac{1}{L} \sum_{t=1}^{L} \log P(\tilde{x}_t | x_{<t}, \tilde{x}_{<t}, \mathbf{z}) \Big] \tag{5.2}$$

After pre-training, the encoder will be fixed and the encoded representations will be directly used for the generation of the latent intermediate domain (Section 5.4).

Alternatively, we can directly use publicly available text encoders like BERT [47], which are designed to be general-purpose and pre-trained in a self-supervised fashion on a mixture of data sources. In this study, we experiment with BERT, and take the hidden state of the "[CLS]" token as the text representation.[1]

Intermediate Domain Generator

GAN with Cycle-consistency. This module generates an intermediate representation domain from the pre-trained representation space $\mathcal{Z}$ to bridge the gap across source and target, as shown in Figure 5.3. For that purpose, we introduce a source-to-target generator $G_{s \to t}$, and train it to generate target representations that aim to fool an adversarial discriminator D_t. This gives the following GAN loss:

$$\mathcal{L}_{\text{gan}}^{s \to t} = \mathbb{E}_{\mathbf{z} \sim \mathcal{Z}_S} \log[D_t\big(G_{s \to t}(\mathbf{z})\big)] + \mathbb{E}_{\mathbf{z} \sim \mathcal{Z}_T} \log[1 - D_t(\mathbf{z})] \tag{5.3}$$

[1]Note that the "[CLS]" representation is typically used as a text representation at the fine-tuning stage with supervision from end tasks, whereas we adopt it here as an unsupervised text representation.

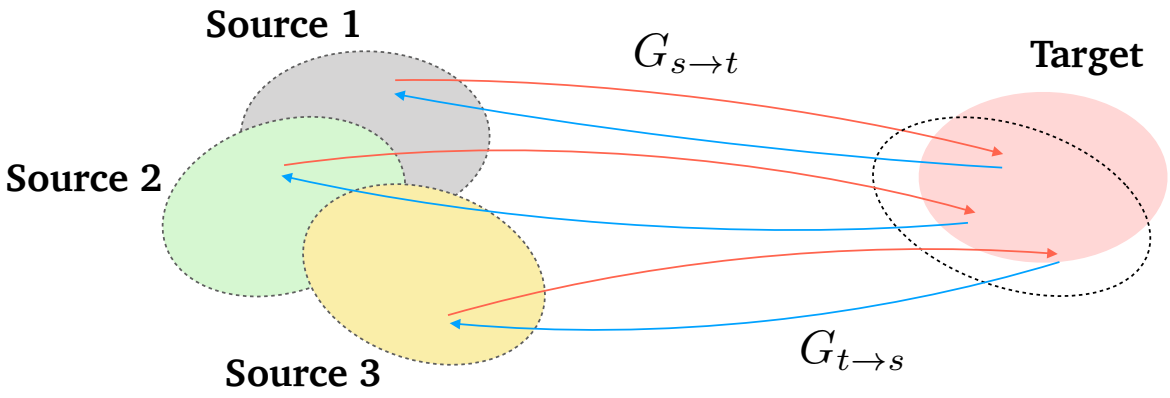


Figure 5.3: Intermediate domain generation with a CycleGAN.

In order to avoid *mode collapse* in the generated intermediate domain and encourage the internal structural information of the original example to be preserved, we follow [101] and optimize a *cycle-consistency* loss which is obtained by recontructing the representation of the original example from the intermediate domain representation. To implement this loss, we introduce a reverse generator from target to source $G_{t\to s}$, which can be trained using a reverse GAN loss $\mathcal{L}_{\text{adv}}^{t\to s}$ (this requires an additional discriminator at source side D_s). Then, the cycle-consistency loss can be written as:

$$\begin{aligned}\mathcal{L}_{cyc} = &\mathbb{E}_{\mathbf{z}\sim\mathcal{Z}_S} \parallel G_{t\to s}\big(G_{s\to t}(\mathbf{z})\big) - \mathbf{z} \parallel_1 \\ &+ \mathbb{E}_{\mathbf{z}\sim\mathcal{Z}_T} \parallel G_{s\to t}\big(G_{t\to s}(\mathbf{z})\big) - \mathbf{z} \parallel_1\end{aligned} \tag{5.4}$$

The above loss function treats all source examples in a training batch equally, while ignoring their varied relevance/importance to the target domain distribution due to the multi-source nature. To cope with this challenge, we explore two instance-level weight assignment mechanisms which operate on each batch: the *model-based curriculum* and the *model-free curriculum*.

Model-based Curriculum. We follow [256] and use an extra source selection network for calculating the weight distribution over examples in a batch. This network takes the generated representation ($G_{s\to t}(\mathbf{z})$) as input, and outputs a weight distribution with a Softmax layer. Denoting B as a batch of encoded examples sampled from the $\mathcal{Z}$: $\{\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_{|B|}\}$, the sample weights can be computed as:

$$\mathbf{w} = \texttt{softmax}\Big(h_t\big(G_{s\to t}(B)\big)\Big) \tag{5.5}$$

where h_t is the source selection network at target side. We then obtain the curriculum GAN loss ($\mathcal{L}_{cgan}$) as:

$$\begin{aligned}\mathcal{L}_{\text{cgan}}^{s\to t} = &\mathbb{E}_{B\sim\mathcal{Z}_S} \frac{1}{|B|} \sum_{\mathbf{z}\in B} w_z \log[D_t\big(G_{s\to t}(B)\big)] \\ &+ \mathbb{E}_{\mathbf{z}\sim\mathcal{Z}_T} \log[1 - D_t(\mathbf{z})]\end{aligned} \tag{5.6}$$

In the ideal case, if the input batch of the source selection network is extremely close to the *target* distribution, we would expect a uniform weighting. Therefore, we introduce additional inductive bias for training h_t by minimizing the KL-divergence between the output distribution and a uniform distribution ($\mathcal{U}$) when the input batch is sampled from the real target space:

$$\mathcal{L}_{uni}^{t} = \mathbb{E}_{\mathbf{z}\sim\mathcal{Z}_T}\mathrm{KL}[h_t(\mathbf{z}) \parallel \mathcal{U}] \tag{5.7}$$

The formulation of $\mathcal{L}_{cgan}^{t\to s}$ and $\mathcal{L}_{uni}^{s}$ can be adapted in a similar way, using a separate source selection network h_s.

Model-free Curriculum. Instead of relying on an extra source selection network, we can also compute sample weights directly from outputs of the domain discriminators (D_t), which indeed reflects the proximity of each example to the target domain. This gives us the following model-free weight assignment mechanism:

$$\mathbf{w} = \texttt{softmax}\Big(\log\big[D_t\big(G_{s\to t}(B)\big)\big]\Big) \tag{5.8}$$

In this way, examples with a higher probability of being classified as *target* will be more emphasized in the GAN loss.

Task Classifier

Assuming the source-to-target generation ($G_{s\to t}$) does not change the sentiment label, we can train a transferable sentiment classifier over the generated intermediate domain: $f_t : G_{s\to t}(\mathcal{Z}) \to \mathcal{Y}$ using labels from the source domains:

$$\mathcal{L}_{task} = -\mathbb{E}_{(\mathbf{z},y)\sim(\mathcal{Z}_S,\mathcal{Y}_S)}\Big[-\log P\Big(y|f_t\big(G_{s\to t}(\mathbf{z})\big)\Big)\Big] \tag{5.9}$$

After training, the classifier f_t can be directly used in the target domain. To promote sentiment consistency between the generated intermediate representations and their original examples, we further backpropagate the task loss to the source-to-target generator.

Joint Learning

Our final objective is a weighted combination of different losses in the C-CycleGAN framework.[2] For the model-based curriculum:

$$\mathcal{L}_{\text{c-cyclegan}} = \mathcal{L}_{\text{cgan}}^{s\to t} + \mathcal{L}_{\text{cgan}}^{t\to s} + \mathcal{L}_{\text{cyc}} + \mathcal{L}_{\text{uni}}^{t} + \mathcal{L}_{\text{uni}}^{s} + \mathcal{L}_{\text{task}} \tag{5.10}$$

For the model-free curriculum:

$$\mathcal{L}_{\text{c-cyclegan}} = \mathcal{L}_{\text{cgan}}^{s\to t} + \mathcal{L}_{\text{cgan}}^{t\to s} + \mathcal{L}_{\text{cyc}} + \mathcal{L}_{\text{task}} \tag{5.11}$$

This objective can be optimized by solving the following min-max game:

$$f_t^* = \arg\min_{f_t} \min_{\substack{G_{s\to t}\\ G_{t\to s}\\ h_t,h_s}} \max_{D_s,D_t} \mathcal{L}_{\text{c-cyclegan}} \tag{5.12}$$

[2]We omit the weights of each loss term for notation ease.

Table 5.1: Comparison with the state-of-the-art DA methods on Reviews-5 dataset. All numbers are percentages. The best class-wise and average classification accuracies trained on the source domains are emphasized in bold (similar below).

Standards	Models	Camera	Laptop	Restaurant	Movie1	Movie2	Avg
Source-only	Single-best	68.8	62.5	64.0	76.9	75.8	69.6
	Source-combined	69.6	71.5	68.5	77.0	76.7	72.7
Single-best DA	DANN [69]	69.2	72.6	68.5	78.3	80.7	73.9
	ADDA [221]	69.4	73.2	69.6	79.1	81.5	74.6
	DAAN [258]	69.4	73.8	71.6	79.5	82.8	75.4
Source-combined DA	DANN [69]	67.0	73.3	68.2	77.4	80.8	73.3
	ADDA [221]	69.6	74.1	69.5	80.5	82.6	75.3
	DAAN [258]	69.4	74.6	72.4	80.2	83.2	76.0
Multi-source DA	Autoencoder+MDAN [272]	65.0	59.0	64.5	60.8	52.1	60.3
	MDAN (TextCNN) [272]	68.0	72.0	71.0	77.4	78.7	73.4
	CMSS [256]	71.8	75.4	73.3	81.2	85.6	77.5
	C-CycleGAN (Ours)	**73.8**	**76.0**	**76.0**	**82.0**	**87.5**	**79.1**
Oracle	TextCNN	76.8	77.5	77.5	84.4	90.6	81.4

Table 5.2: Comparison with the state-of-the-art DA methods on Reviews-5 dataset using BERT embedding.

Standards	Models	Camera	Laptop	Restaurant	Movie1	Movie2	Avg
Source-only	Single-best	72.3	74.5	75.4	79.4	83.1	76.9
	Source-combined	73.6	74.8	76.8	80.1	85.7	78.2
Multi-source DA	**C-CycleGAN (Ours)**	**76.9**	**78.4**	**79.7**	**83.1**	**88.3**	**81.3**
Oracle	BERT	78.3	79.5	81.2	85.1	90.8	83.0

5.5 Experiments

In this section, we introduce the experimental settings and present results as well as analysis. Our source code will be released.

Experimental Settings

Datasets.

We evaluate our approach using two combined datasets of cross-domain sentiment analysis: Reviews-5 [261] and Amazon benchmark [29]. Each dataset contains multiple domains. For each dataset, we create multiple MDA settings by taking each domain as *target*, and the rest as *sources*. In addition, we further consider a cross-lingual transfer setting using the

Multilingual Amazon Reviews Corpus [34], to validate the generalizability of our approach to a broader family of transfer learning.

The **Reviews-5 dataset** [261] includes five domains of customer reviews. *Movie1* [167] and *Movie2* [203] are movie reviews; *Camera* [105] contains reviews of digital products such as MP3 players and cameras; *Laptop* and *Restaurant* are laptop and restaurant reviews respectively taken from SemEval 2015 Task 12 [261]. The training set sizes are 3,270, 1,707, 1,372, 9,162, and 8,113 for Movie1, Movie2, Camera, Laptop and Restaurant, respectively. The test size is 200 for all domains.

The **Amazon benchmark dataset** [29] contains four domains of product reviews on Amazon: *Books*, *DVD*, *Kitchen*, and *Electronics*, with training set sizes of 6,465, 5,586, 7,681, and 7,945 respectively. The test size is 200 for all domains. This dataset has been preprocessed by the authors into TF-IDF representations, using the 5,000 most frequent unigram and bigram features. Therefore, word order information is unavailable.

The **Multilingual Amazon Reviews Corpus** [34] is a collection of Amazon reviews from four languages: *German*, *English*, *French*, and *Japanese*. For each language, there are three domains including Books, DVD, and Music. The training set size and test set size for each domain of each language are 52,000 and 2,000.

Evaluation Metrics.

Following [272, 90], we use classification accuracy as metric to evaluate the sentiment classification results. Larger values represent better performances.

Baselines.

We consider the following baselines:

1. **Source-only**, directly training on the source domains and testing on the target domain, which includes two settings: single-best, the best test accuracy on target among all source domains; source-combined, the target accuracy of the model trained on the combined source domain.

2. **Single-source domain adaptation methods**, including DANN [69], ADDA [221], and DAAN [258], trained with both single-best and source-combined settings.

3. **Multi-source domain adaptation models**, including state-of-the-art approaches MDAN [272], MoE [90], and CMSS [256].

We also report the results of an oracle setting, where the model is both trained and tested on the target domain.

Table 5.3: Comparison with the state-of-the-art DA methods on Amazon Benchmark dataset.

Standards	Models	Books	DVD	Kitchen	Electronics	Avg
Source-only	Single-best	75.4	81.3	86.5	86.5	82.4
	Source-combined	76.5	81.6	86.7	85.3	82.5
Single-best DA	DANN [69]	76.5	77.2	83.6	84.3	80.4
	ADDA [221]	74.4	78.2	82.6	82.1	79.3
	DAAN [258]	77.2	76.8	83.5	86.5	81.0
Source-combined DA	DANN [69]	77.9	78.9	84.9	86.4	82.0
	ADDA [221]	76.6	77.1	82.5	82.5	79.7
	DAAN [258]	78.4	77.6	85.4	87.2	82.2
Multi-source DA	MDAN [272]	78.0	85	85.3	86.3	82.5
	MoE [90]	78.9	81.3	87.4	87.9	83.9
	CMSS [256]	78.1	80.2	87.2	87.2	83.2
	C-CycleGAN (Ours)	**80.3**	**82.2**	**88.9**	**89.1**	**85.1**
Oracle	TextCNN	76.7	81.3	87.1	85.2	82.6

Implementation Details.

For the pre-training of text encoder, we use a 2-layer Bidirectional LSTM as *encoder* and a 1-layer LSTM as *decoder*. The initial learning rate is 0.00001 with a decay rate of 0.5 every 200 steps. The dimension of word embeddings and hidden states are both set to 256. For experiments with BERT, we use the 12-layer "bert-base-uncased" version due to memory constraints. The weights for $\mathcal{L}_{\text{cgan}}$, $\mathcal{L}_{\text{cyc}}$, $\mathcal{L}_{\text{uni}}$, and $\mathcal{L}_{\text{task}}$ are 0.1, 1, 1 and 1, respectively. During decoding, we choose as input between the true previous token and the generated token with a probability of 0.5 of selecting either one. For the Amazon benchmark dataset, we use the original TF-IDF feature vectors as the representation, without further encoding or pre-training. We leverage a 4-layer multi-layer perceptron (MLP) to implement the generator and discriminator of CycleGAN, as well as the sentiment classifier. The initial learning rate is 0.0001 with a decay rate of 0.5 every 100 steps. We use Adam [120] as the optimizer with beta1 of 0.5, beta2 of 0.999, batch size of 64, and weight decay of 0.0001. In the multilingual transfer experiments, we obtain cross-lingual word embeddings by projecting the pre-trained monolingual word embeddings [14] of the 4 languages into English (pivot language) using an unsupervised method [3].

Results on Reviews-5 Dataset

We first evaluate our approach on the dataset of plain textual input: Reviews-5. We perform experiments with each domain as the target and the rest as sources. Table 5.1 shows the

Table 5.4: Comparison with the state-of-the-art DA methods on Multilingual Amazon Reviews Corpus dataset.

Standards	Models	German				English			
		Books	DVD	Music	Avg	Books	DVD	Music	Avg
Source-only	Single-best	63.6	64.7	64.9	64.4	65.3	62.5	63.3	63.7
	Source-combined	61.5	64.6	63.6	63.2	63.7	65.0	60.1	63.0
Multi-source DA	**C-CycleGAN (Ours)**	**78.3**	**78.4**	**79.1**	**78.6**	**78.0**	**77.8**	**79.0**	**78.3**
Oracle	TextCNN	83.2	89.0	88.2	86.8	85.2	85.5	81.1	83.9
Standards	Models	French				Japanese			
		Books	DVD	Music	Avg	Books	DVD	Music	Avg
Source-only	Single-best	65.3	64.3	64.2	64.6	63.5	63.5	64.8	64.0
	Source-combined	63.6	63.0	63.4	63.3	63.7	62.7	64.0	63.4
Multi-source DA	**C-CycleGAN (Ours)**	**78.6**	**77.6**	**76.9**	**77.7**	**75.2**	**74.9**	**76.8**	**76.2**
Oracle	TextCNN	88.3	77.6	84.1	83.3	60.4	61.8	69.4	69.4

Table 5.5: Ablation study on different components of the proposed C-CycleGAN framework on the Reviews-5 dataset.

Models	Camera	Laptop	Restaurant	Movie1	Movie2	Avg
CycleGAN [285]	68.7	75.4	71.6	82.5	86.7	77.0
MDAN [272] + CycleGAN [285]	70.8	75.2	71.2	79.9	86.2	76.7
CycleGAN [285]+CMSS [256]	71.5	75.4	70.8	81.1	86.1	77.0
C-CycleGAN (model-based)	72.8	75.7	73.5	81.7	87.3	78.2
C-CycleGAN (model-free)	**73.8**	**76.0**	**76.0**	**82.0**	**87.5**	**79.1**

performance of different DA methods and Table 5.2 shows the extended results using BERT embedding [47]. We have the following observations[3]:

(1) Without considering domain shift, both source-only settings, i.e. single-best and source-combined, obtain poor accuracy: 69.6% and 72.7%, around 10% worse than the oracle (81.4%). This motivates the research on domain adaptation.

(2) When directly applying to the MDA task, the single-source DA methods outperform the source-only setting. Since customers' reviews vary a lot across domains, features related to sentiment also vary a lot. Therefore these DA methods that can make the domain gap smaller achieve better results than source-only setting.

(3) Comparing the performances of source-combined and single-best DA settings, we can find that sometimes naively performing single-source domain adaptation approaches on

[3]The first 5 points are based on Table 5.1, and the last point is based on Table 5.2.

a combined dataset of different sources could produce worse result (i.e. 73.3% of DANN) than on a single source domain (i.e. 73.9% of DANN). This naturally motivates research on multi-source domain adaptation.

(4) Most of the state-of-the-art multi-source domain adaptation methods perform better than single-source domain adaptation methods by considering domain-invariant features and fusing information across all domains. However, MDAN [272], which has been demonstrated to be effective on Amazon benchmark dataset, performs worse (60.3% and 73.4%) than single-best DA settings (e.g. 74.6% and 75.4%). This indicates that some of the previous multi-source domain adaptation methods may be only effective on a certain kind of data representation (e.g. bag-of-words or TF-IDF).

(5) C-CycleGAN performs the best (79.1%) among all adaptation settings. Compared to the best results inside the Source-only, Single-best DA, Source-dombined DA and other Multi-source DA methods, C-CycleGAN achieves 6.4%, 3.7%, 3.1% and 1.6% performance boost, respectively. These results demonstrate that the proposed C-CycleGAN model can achieve significant better performance compared to state-of-the-art methods. The performance improvements benefit from the advantages of C-CycleGAN. First, an intermediate representation domain is generated with cycle-consistency and sentiment consistency which is closer to the target domain and preserves the annotation information of the source samples. Second, the proposed weighting mechanisms can dynamically assign weights to different source samples, which takes into account the source samples' similarity to the target and enhances the adaptation performance. Finally, the text reconstruction in the pre-trained text encoder minimizes the information loss during the feature encoding process.

(6) BERT embedding performs much better than Bi-LSTM for all the methods, which demonstrates the superiority of BERT in learning pre-trained embeddings. The proposed C-CycleGAN achieves 3.1% performance gains as compared to the best source-only setting.

Results on Amazon Benchmark Dataset

Table 5.3 shows the results on the Amazon benchmark dataset, which takes TF-IDF as text representations. We can observe that:

(1) Comparing the performance of source-only (82.5%) and Oracle (82.6%), we can see that the domain gap between sources and target is less than 1%, much smaller than the domain gap of Reviews-5 (¿10%). This indicates that the data representation type of the datasets is closely associated with how large the domain gap is.

(2) Several multi-source adaptation methods (e.g. MoE [90]) perform even better than Oracle. This is because that the domain gap is relatively small and multi-source adaptation leverages more information from multiple domains than Oracle, which only has access to the samples from the target. This further indicates the importance of diverse data from different source domains.

(3) The proposed C-CycleGAN has the best performance (85.1%) among all approaches with 1.2% and 2.5% better classification accuracy than MoE and Oracle respectively. Compared to other methods (e.g. MDAN) whose performance fluctuates significantly across

Table 5.6: Ablation study on the influence of cycle-consistency in C-CycleGAN on the Reviews-5 dataset.

Models	Camera	Laptop	Restaurant	Movie1	Movie2	Avg
C-CycleGAN w/o cycle-consistency	72.2	73.1	72.0	80.3	85.1	76.5
C-CycleGAN w cycle-consistency	**73.8**	**76.0**	**76.0**	**82.0**	**87.5**	**79.1**

datasets (Reviews-5 and Amazon Benchmark datasets), the proposed C-CycleGAN can provide consistent superior performance across datasets.

Multilingual Transfer Experiments

We also perform experiments on the Multilingual Amazon Reviews Corpus. For each category domain (Books, DVD, Music) of each language, we perform adaptation to it with datesets of the same category domain from other languages as sources. Table 5.4 shows the performance of different adaptation methods. We can observe that:

(1) The proposed C-CycleGAN achieves the best performance of all DA methods across all languages and on all category domains.

(2) In most cases, Oracle gives the best performance; however, in several settings, C-CycleGAN can achieve similar or even better results than the oracle (e.g. 77.6% and 77.6% for DVD in French; 76.8% and 69.4% for Music in Japanese). This further demonstrate that our framework has a wide range of applicability, not only across different types of data representation, but also across different languages.

Ablation Study

We conduct a series of ablation studies on the Reviews-5 dataset to demonstrate the improvements of C-CycleGAN over existing state-of-the-art approaches. The results are described in Table 5.5, where all CycleGANs are performed in a source-combined manner.

First, we investigate whether it is necessary to align the representations before applying CycleGAN. "MDAN + CycleGAN" in Table 5.5 represents first aligning the encoded representations using MDAN and then applying CycleGAN. Comparing the first two rows in Table 5.5, we can see applying MDAN before CycleGAN achieves worse performance, which indicates that it is unnecessary to perform additional alignment before CycleGAN. This is probably because extracting the domain-invariant features between the source and target domains might lose some discriminative features in the target domain that are related to sentiment.

Second, we investigate the effectiveness of the proposed model-based and model-free weighting methods. From the last three rows, we can see that compared to CMSS [256], the proposed model-based and model-free weighting schemes improve accuracy by 1.2% and

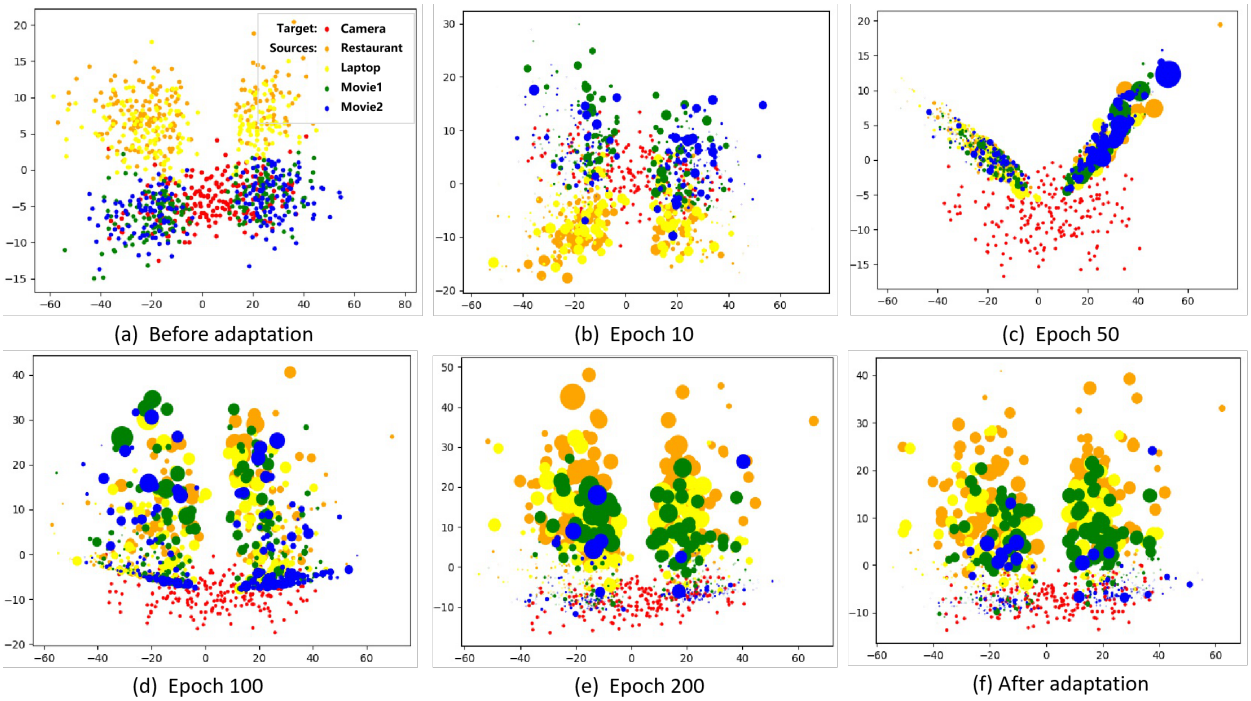


Figure 5.4: Visualization of feature spaces in different training stages of C-CycleGAN on the Reviews-5 dataset. Target samples are in red, while source samples are in other colors. Point size denotes the similarity of each source sample to the target domain obtained from output of the domain discriminator. For better visualization, smaller points represent samples closer to the target domain.

2.1% respectively. Because CMSS takes the original source samples as input to compute the weights, it cannot reflect the dynamic changing of source samples' weights. The proposed model-based weighting mechanism is based on the generated intermediate domain, which itself dynamically changes. The model-based method requires an additional network to compute the similarity to the target domain, which not only increase the computation cost, but also takes longer to learn the discriminative patterns between sources and target, before which CycleGAN may learn the wrong patterns.

Finally, we evaluate the influence of cycle-consistency in the proposed C-CycleGAN model. As in [285], we find that standard adversarial procedures without cycle-consistency often lead to the mode collapse problem, where all input representations are mapped to the same output representation and the optimization fails to make progress. The comparison between with and without cycle-consistency in C-CycleGAN on the Reviews-5 dataset is shown in Table 5.6. The result comparison (79.1 vs. 76.5) clearly demonstrates the effectiveness and necessity of cycle-consistency.

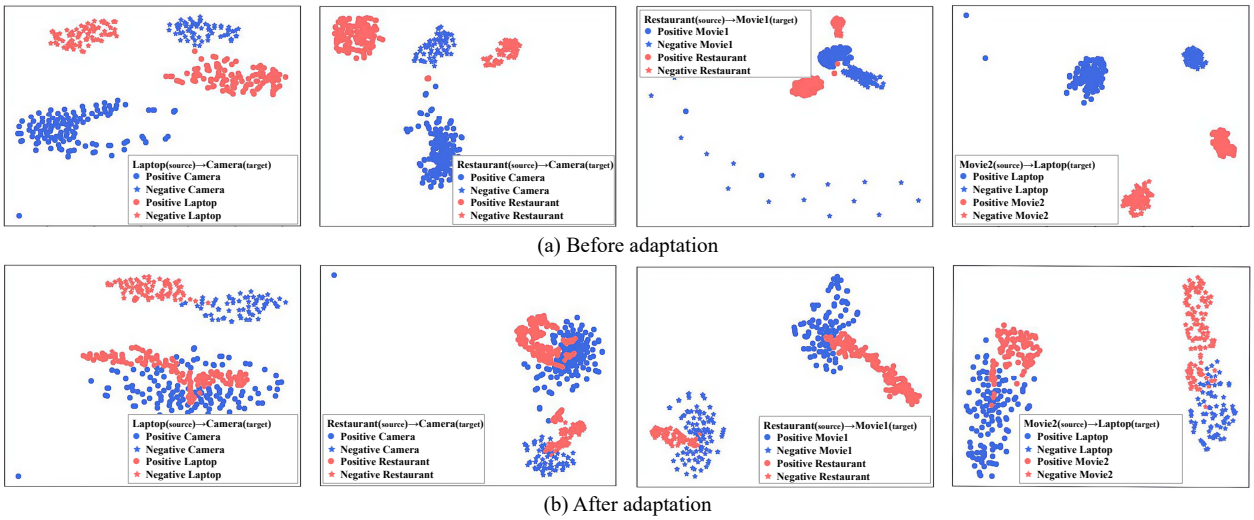


Figure 5.5: t-SNE visualization of the features before and after adaptation on the Reviews-5 dataset. Red represents source features and Blue represents target features.

Visualization

In this section, we visualize the features of source and target samples during different training stages of C-CycleGAN.

By using PCA to reduce the dimensionality of samples, we project samples from five domains in Reviews-5 [261] onto a 2-dimensional plane in different stages of training. The visualization results are shown in Figure 5.4. We can conclude that during the training process, all source domains get closer to the target domain. At the same time, we can see that the samples far from the target domain can be well differentiated by the discriminator, and are assigned with smaller weights (larger points).

Figure 5.4 (a) and (f) visualize the representation space before and after adaptation correspondingly. We can see that the samples in Movie1 and Movie2 are the closest since they are all about reviews in movies. Movie1 is also closer with Camera and Laptop after adaptation, which is desirable because these domains involve common reviews on image quality or upgrade of electronics. For example, the Camera domain may have reviews like "Picture is clear and easy to carry. Love SONY."; while in Movie1: "Transitions smoothly and the image quality is clean", and in Laptop: "The 4K display is so sharp, the slim book is so light in a bag". We can hardly distinguish which domains these reviews belong to without prior information.

We further plot the learned features with t-SNE [152] on four adaptation settings, with the results shown in Figure 5.5. The top row represents the feature embeddings before adaptation, while the bottom row represents the feature embeddings after adaptation by C-CycleGAN. Red represents source features and Blue represents target features. As we can see, before adaptation, the source samples can be obviously classified but such classifier

cannot work well on the target samples; with the proposed C-CycleGAN, source and target features of the same class become more aggregated after adaptation. These observations further demonstrate the effectiveness of C-CycleGAN.

5.6 Conclusion

In this chapter, we proposed a novel multi-source domain adaptation framework, named curriculum cycle-consistent generative adversarial network (C-CycleGAN), for textual sentiment classification. C-CycleGAN contains three main component: pre-trained text encoder for encoding text instances into a latent continuous representation space with minimal information loss; intermediate domain generator with curriculum instance-level adaptation considering the importance of different source samples; and task classifier to perform the final sentiment classification. The generated intermediate domain bridges the domain gap between the source and target domains, while preserving the sentiment semantics. The proposed dynamic model-based and model-free weighting mechanisms can assign higher weights to the source samples that are closer to the target domain. Further, C-CycleGAN does not require prior domain labels of source samples, which makes it more practical in real-world scenarios. Extensive experiments on multiple benchmark datasets demonstrate that C-CycleGAN significantly outperforms existing state-of-the-art DA methods. In future studies, we plan to construct a large-scale textual dataset with more fine-grained sentiment categories and extend our framework to corresponding MDA tasks. We will explore multi-modal domain adaptation by jointly modeling multiple modalities, such as image and text.

Chapter 6

Multi-source Few-shot Domain Adaptation

6.1 Chapter Overview

This chapter targets a new problem of multi-source few-shot domain adaptation, where there are multiple source domains but each with only limited labels. We propose a framework extending the method in Chapter 2 for this new problem.

6.2 Introduction

Deep neural networks have achieved remarkable performance for a variety of computer vision tasks [94, 144, 176, 122]. Despite high accuracy, these models have consistently fallen short in generalizing to new domains due to the presence of *domain shift* [217, 221, 54]. Unsupervised Domain Adaptation (UDA) is a challenging, yet frustratingly practical, setting which aims to transfer predictive models from a single fully-labeled source domain to an unlabeled target domain. UDA methods typically operate by using a task loss on the labeled source samples, as well as additional losses to account for domain shift, such as a discrepancy loss [147, 206, 205, 17, 289], adversarial loss [221, 83, 139, 100, 184, 189], and reconstruction loss [77, 76, 16].

Rather than using only a single labeled source domain, Multi-source Domain Adaptation (MDA) [154, 62, 252] generalizes this setting by transferring the task knowledge from multiple fully labeled source domains to an unlabeled target domain. Each source domain is correlated to the target in different ways and adaptation involves not only incorporating the combined prior knowledge from multiple sources, but simultaneously preventing the possibility of negative transfer [2]. Many MDA methods [280, 252, 171, 193, 226] outperform UDA methods and achieve high accuracy on the target domain by leveraging the abundant explicit supervision in the source domains, together with the unlabeled target samples for domain alignment.

In many real-world applications, however, getting large-scale annotations even in the source domain is often challenging due to the difficulty and high cost of annotation. For example, during the COVID-19 pandemic [225, 20], Harmon *et al.* [92] explored transferring a medical predictive model trained with data from different countries to a target country. However, during the early stages of the pandemic, there were few domain experts that could provide such annotations and even obtaining fully labeled source data was impractical. As another example, each retinal image in the Diabetic Retinopathy dataset [89] is annotated by a panel of 7+ U.S. board-certified ophthalmologists, with a total group of 54 doctors used for annotation [89, 265]. Thus it is practically too stringent to assume the availability of abundant labels across domains.

In this chapter, we explore a new Multi-source Few-shot Domain Adaptation (MFDA) setting that mitigates the need for large-scale labeled source datasets. In MFDA, only a small number of samples in each source domain are annotated while the rest source and target samples remain unlabeled. Many MDA methods seek to learn domain-invariant features by performing some form of distribution alignment [252, 171, 280, 272, 193], and learn discriminative features by performing supervised task loss on all source domains. In MFDA, however, with a limited number of labels in each source, it is much harder to learn discriminative features for both source and target. Some recent works [119, 265] perform few-shot adaptation with a single source, and in this work, we build upon these contributions and investigate the multiple source scenario.

In the newly proposed MFDA setting, we show that many existing domain adaptation methods do not learn discriminative features for both source and target domains. Therefore, we propose a Multi-Source Few-shot Adaptation Network (MSFAN), which consists of three major components: (i) multi-domain, self-supervised learning (SSL) with feature prototypes, (ii) cross-domain consistency learning that leverages a support-set of labeled and pseudo-labeled samples, and (iii) multi-domain prototypical classifier learning. In the first component, multi-domain prototypical SSL is performed within each domain and each source-target domain pair. The in-domain prototypical SSL aims to learn a well-clustered representation in each domain, while the cross-domain SSL aligns each source domain with the target domain. For the second component, MSFAN builds a support set consisting of the few labeled samples and high-confident unlabeled samples. Based on the support sets, a cross-domain prediction consistency is then enforced to further promote domain-invariant feature learning. Finally, we leverage both the prototypes and the labeled samples from all source domains in order to learn a good classifier for each domain. Mutual information constraints are further enforced on both source and target across all classifiers to learn better domain-invariant and class-discriminative features.

In summary, our contributions are three-fold. (1) We propose a new domain adaptation task MFDA, adapting to a fully unlabeled target domain from multiple sources with few labels, which is a both practical and challenging generalization of conventional multi-source domain adaptation. (2) To address this challenge, we propose a novel Multi-Source Few-shot Adaptation Network (MSFAN), which learns discriminative and domain-invariant features from multiple domains with only few labels. (3) We conduct extensive MFDA experiments

and demonstrate that the proposed method outperforms state-of-the-art MDA methods by large margins across multiple benchmark datasets, with 20.2%, 9.4%, and 16.2% improvement on Office, Office-Home, and DomainNet, respectively.

6.3 Multi-Source Few-shot Domain Adaptation

We consider the Multi-source Few-shot Domain Adaptation (MFDA) problem, in which there is one unlabeled target domain and M partially labeled source domains. In the i-th source domain, there is small labeled set $\mathcal{S}_i = \{(\mathbf{x}_i^j, y_i^j)\}_{j=1}^{N_i}$, and a large unlabeled set $\mathcal{S}_i^u = \{\mathbf{x}_i^{j,u}\}_{j=1}^{N_i^u}$, both drawn from the source distribution $p_i(\mathbf{x}, y)$ with partial label observation. N_i and N_i^u respectively denote the number of labeled and unlabeled samples in domain i, with $N_i \ll N_i^u$. In the target domain, let $\mathcal{T} = \{\mathbf{x}_T^j\}_{j=1}^{N_T}$ denote the target data drawn from the target distribution $p_T(\mathbf{x}, y)$ without label observation, where N_T is the number of target samples. We aim to learn an adaptation model that can correctly predict labels of target samples by training on $\{\mathcal{S}_i\}_{i=1}^M$, $\{\mathcal{S}_i^u\}_{i=1}^M$ and $\mathcal{T}$.

Figure 6.1 provides an overview of the Multi-Source Few-show Domain Adaptaion (MSFAN) framework proposed in this chapter. It consists of a base model and three major components: Multi-domain Prototypical Self-supervised Learning, Cross-domain Consistency via Support Sets, and Multi-domain Prototypical Classifier Learning. Similar to many previous works [252, 171, 232, 226], the base model of the MSFAN framework consists of a shared feature extractor F and M classifiers $\{C_i\}_{i=1}^M$, one for each source domain. However, instead of a standard linear classifier, each C_i is a cosine similarity-based classifier [32, 188]. In addition, there is an ℓ_2 normalization layer between F and C_i, which output a feature vector $\mathbf{f} \in \mathbb{R}^d$.

Multi-domain Prototypical Self-supervised Learning

Learning discriminative target features with limited labels per source domain and no labels in the target is a difficult task as the only categorical grounding comes from the few labeled source examples. To address this task, we propose to learn the latent feature-space clustering of each source and target domain, and align clusters with the same category across different domains in a self-supervised manner. Specifically, we use a ProtoNCE [135] loss to learn the semantic feature of a single domain as it has been shown to semantically align data across a single source and target domain [265]. We further extend it into the multi-source adaptation scenario to learn better discriminative and domain-invariant features across all domains.

In-domain Prototypical Contrastive Learning. To learn a well-clustered semantic structure in the feature space, it is problematic to apply ProtoNCE on a mixed dataset with different distributions, because images of different categories from different domains may be incorrectly aggregated into the same cluster. As a result, due to the domain shift between sources and target, we cannot directly apply ProtoNCE to $\bigcup_{i=1}^M (\mathcal{S}_i \cup \mathcal{S}_i^u) \cup \mathcal{T}$ as in [135], and

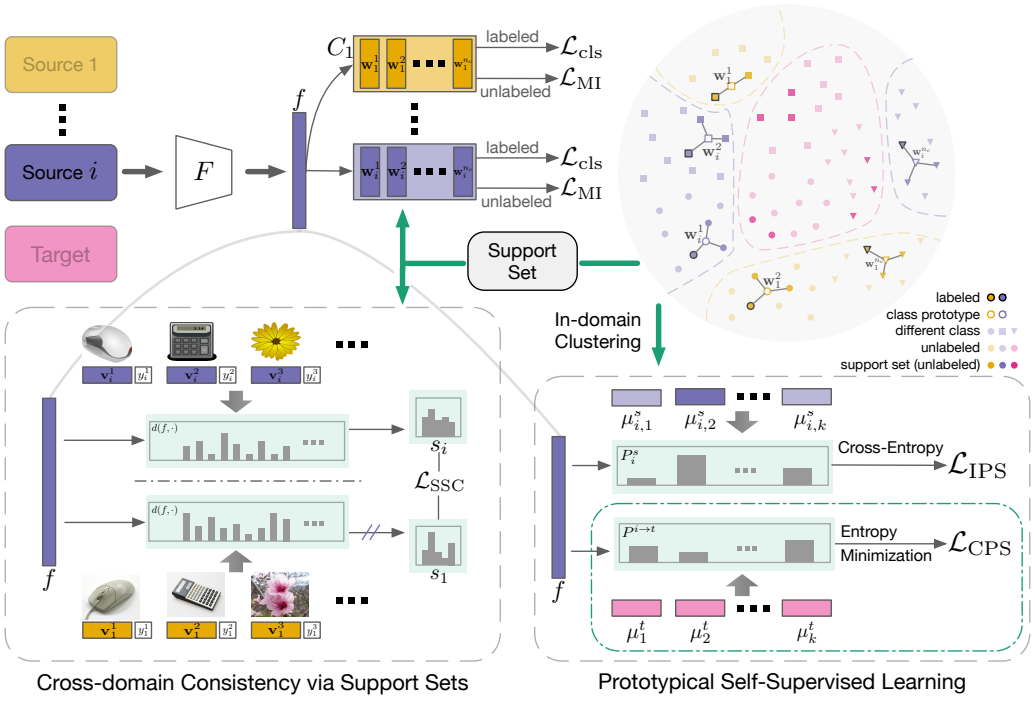


Figure 6.1: An overview of the proposed MSFAN framework, which consistes of Multi-domain Prototypical Self-supervised Learning (bottom-right), Cross-domain Consistency via Support Sets (bottom-left), and Prototypical Classifier Learning (top).

due to the domain shift among different sources, we cannot apply ProtoNCE to the source $\bigcup_{i=1}^{M}(\mathcal{S}_i \cup \mathcal{S}_i^u)$ and target $\mathcal{T}$ separately as [265]. Instead, we perform prototypical contrastive learning in each source $\mathcal{S}_i \cup \mathcal{S}_i^u)$ and target $\mathcal{T}$.

We maintain a memory bank $\boldsymbol{V_i^s}$ for each source domain i, and a memory bank $\boldsymbol{V^t}$ for the target:

$$\boldsymbol{V_i^s} = [\mathbf{v}_{i,1}^s, \mathbf{v}_{i,2}^s, \cdots, \mathbf{v}_{i,(N_i+N_i^u)}^s], \ \boldsymbol{V^t} = [\mathbf{v}_1^t, \mathbf{v}_2^t, \cdots, \mathbf{v}_{N_T}^t], \tag{6.1}$$

where $\mathbf{v}_j$ is the feature vector of $\mathbf{x}_j$ which is initialized with $\mathbf{f}_j$ and updated with a momentum η:

$$\mathbf{v}_j \leftarrow \eta \mathbf{v}_j + (1-\eta)\mathbf{f}_j. \tag{6.2}$$

After a set number of iterations, k-means clustering is performed separately on each $\boldsymbol{V_i^s}$ to obtain clusters $\boldsymbol{C_i^s} = \{C_{i,1}^s, C_{i,2}^s, \ldots, C_{i,k}^s\}$, and on the target to obtain target clusters $\boldsymbol{C^t}$. Then the normalized prototypes in source i are computed as $\{\mu_{i,c}^s\}_{c=1}^k$, where $\mu_{i,c}^s = \frac{\mathbf{u}_{i,c}^s}{\|\mathbf{u}_{i,c}^s\|}$ with $\mathbf{u}_{i,c}^s = \frac{1}{|C_{i,c}^s|}\sum_{\mathbf{v}_{i,j}^s \in C_{i,c}^s} \mathbf{v}_{i,j}^s$. Similarly, we compute the target prototypes $\{\mu_c^t\}_{c=1}^k$.

To simplify the explanation, we present a set of operations for source i and note that the same operations are applied to all sources and the target. During the training process, with the feature extractor F and ℓ_2 normalization layer, we compute a normalized feature vector

$\mathbf{f}^s_{i,j} = \ell_2 \circ F(\mathbf{x}^s_{i,j})$, where $\circ$ represents function composition. Then a similarity distribution vector between $\mathbf{f}^s_{i,j}$ and $\{\mu^s_{i,c}\}^k_{c=1}$ is computed as $P^s_{i,j} = [P^s_{i,j,1}, P^s_{i,j,2}, \ldots, P^s_{i,j,k}]$, with

$$P^s_{i,j,c} = \frac{\exp((\mu^s_{i,c} \cdot \mathbf{f}^s_{i,j} - \hat{m})/\phi)}{\sum^k_{r=1} \exp((\mu^s_{i,r} \cdot \mathbf{f}^s_{i,j} - \hat{m})/\phi)}, \ \hat{m} = m \cdot \mathbb{1}_{C^s_{i,c}}(\mathbf{x}^s_{i,j}), \tag{6.3}$$

where m is a margin value inspired by AMS [230]; $\mathbb{1}_{C^s_{i,c}}(\mathbf{x})$ is an indicator function returning 1 only when $\mathbf{x} \in C^s_{i,c}$; and ϕ is a temperature value determining the concentration level of clusters. The in-domain prototypical contrastive loss is then:

$$\mathcal{L}_{\text{PC}} = \sum^M_{i=1} \sum^{N_i+N^u_i}_{j=1} \mathcal{L}_{CE}(P^s_{i,j}, c^s_i(j)) + \sum^{N_T}_{j=1} \mathcal{L}_{CE}(P^t_j, c^t(j)), \tag{6.4}$$

where $c^s_i(\cdot)$ and $c^t(\cdot)$ take as input an instance index in a domain and return the cluster index in $\boldsymbol{C}^s_i$ and $\boldsymbol{C}^t$ respectively.

Considering the non-deterministic nature of k-means algorithm, we perform clustering R times with different number of clusters $\{k_r\}^R_{r=1}$. In MFDA, with prior knowledge of the underling semantic structure (i.e. the number of classes n_c is known), we set $k_r = n_c$ for most r. Finally, the overall in-domain prototypical self-supervision loss is:

$$\mathcal{L}_{\text{IPS}} = \frac{1}{R} \sum^R_{r=1} \mathcal{L}^{(r)}_{\text{PC}} \tag{6.5}$$

Cross-multi-domain Prototypical Self-supervised Learning. With the in-domain prototypical learning, the shared network backbone is able to extract discriminative features. To further ensure learning domain-aligned features between the M source domains, and the target domain, we perform cross-multi-domain Prototypical Self-supervised Learning.

Recently, self-supervised learning methods [119, 265] have been proposed to perform domain alignment between two domains. One trivial extension is combining all source domains first, and then perform domain alignment between $\bigcup^M_{i=1} \mathcal{S}_i \cup \mathcal{S}^u_i$ and $\mathcal{T}$. However, due to the potential domain shift among different sources, aligning the combined source and target would not yield a unified feature distribution. Another trivial extension is aligning each pair of the $M+1$ domains. However, aligning each domain with multiple different domains results in a brittle optimization problem. Moreover, the number of loss terms increases quadratically with the number of source domains.

In order to address these problems, we propose to perform prototypical domain alignment between each source ($\mathcal{S}_i \cup \mathcal{S}^u_i$) and target ($\mathcal{T}$) pair. For each instance in one source domain, we perform entropy minimization on the distribution similarity vector between its representation and all prototypes in the target domain.

Specifically, given a feature vector $\mathbf{f}^s_{i,j}$ in source domain i, and the prototypes $\{\mu^t_c\}^k_{c=1}$ in the target domain, we first compute the similarity distribution vector $P^{i\to t}_j = [P^{i\to t}_{j,1}, \ldots, P^{i\to t}_{j,k}]$,

with

$$P_{j,c}^{i\to t} = \frac{\exp(\mu_c^t \cdot \mathbf{f}_{i,j}^s/\tau)}{\sum_{r=1}^k \exp(\mu_r^t \cdot \mathbf{f}_{i,j}^s/\tau)}, \tag{6.6}$$

in which τ is a temperature value. To promote confident cross-domain instance-prototype matching, we minimize the entropy of $P_j^{i\to t}$: $H(P_j^{i\to t}) = -\sum_{c=1}^k P_{j,c}^{i\to t} \log P_{j,c}^{i\to t}$. Note that different from [265], we do not compute and minimize $H(P_j^{t\to i})$ on the other direction, since aligning one sample with prototypes in different domains would lead to unstable optimization. The final cross-multi-domain prototypical self-supervised loss is then:

$$\mathcal{L}_{\text{CPS}} = \sum_{i=1}^{M} \sum_{j=1}^{N_s+N_s^u} H(P_j^{i\to t}), \tag{6.7}$$

and the final objective for the multi-domain prototypical self-supervised learning is then:

$$\mathcal{L}_{\text{MPS}} = \mathcal{L}_{\text{IPS}} + \mathcal{L}_{\text{CPS}}. \tag{6.8}$$

Cross-domain Consistency via Support Sets

To further promote domain-invariant and class-discriminative features, we propose to enforce a cross-domain similarity consistency using a support set of labeled and pseudo-labeled data.

Support Set. We build a support set $S^{(i)}$ for each source domain i. The support samples in the set contains the labeled samples in S_i, and unlabeled samples in S_i^u with consistent high-confident predictions across all classifiers. Formally, $S^{(i)}$ is compuated as:

$$S^{(i)} = \mathcal{S}_i \cup \{(\mathbf{x}, y) | \, \mathbf{x} \in \mathcal{S}_i^u; \ \forall i', \max \mathbf{p}_{i'}(\mathbf{x}) > t, \arg\max \mathbf{p}_{i'}(\mathbf{x}) = y\}, \tag{6.9}$$

where $\mathbf{p}_i(\mathbf{x})$ is the softmax vector from C_i on $\mathbf{x}$, $y = \arg\max \mathbf{p}_i(\mathbf{x})$, and t is a confidence threshold.

Let $\boldsymbol{V}_{\mathcal{S}^{(i)}}$ denote the support representations computed from $S^{(i)}$, then for a scalar-valued distance function $d(\cdot,\cdot)$, the similarity vector between an input vector $\mathbf{f}_j$ and $S^{(i)}$ can be computed as:

$$\mathbf{s}_{i,j} = \sum_{(\mathbf{v}_{i,k}^s, y_k) \in \boldsymbol{V}_{\mathcal{S}^{(i)}}} \left(\frac{d(\mathbf{f}_j, \mathbf{v}_{i,k}^s)}{\sum_{(\mathbf{v}_{i,r}^s, y_r) \in \boldsymbol{V}_{\mathcal{S}^{(i)}}} d(\mathbf{f}_j, \mathbf{v}_{i,r}^s)} \right) y_k, \tag{6.10}$$

where y_k is the one-hot ground truth label vector associated with the k-th representation in $\mathbf{v}_{\mathcal{S}^{(i)}}$. In this chapter, we choose $d(a,b)$ to be $\exp(\frac{a\cdot b}{\|a\|\|b\|\psi})$, where ψ is a temperature value.

Given an image $\mathbf{x}_j \in \bigcup_{i=1}^M (\mathcal{S}_i \cup \mathcal{S}_i^u) \cup \mathcal{T}$, we want to enforce consistency on its similarity vectors across different source domains by minimizing the cross entropy. Finally, with the similarity vector to another domain i', $\mathbf{s}_{i',j}$, as the soft pseudo-label, the loss can be computed as:

$$\mathcal{L}_{\text{SSC}} = \sum_{\mathbf{x}_j \in \mathcal{D}} \sum_{1 \le i \ne i' \le M} \mathcal{L}_{CE}(\mathbf{s}_{i,j}, \mathbf{s}_{i',j}) \tag{6.11}$$

Multi-domain Prototypical Classifier Learning

MSFAN incorporates a multi-domain prototypical classifier to learn better domain-aligned, class-discriminative features. It accomplishes this through a simple cosine classifier for each source domain $\{C_i\}_{i=1}^{M}$. Each cosine classifier C_i consists of weight vectors $\mathbf{W}_i = [\mathbf{w}_i^1, \mathbf{w}_i^2, \ldots, \mathbf{w}_i^{n_c}]$, where n_c is the number of classes, and a temperature T. The output of C_i, $\frac{1}{T}\mathbf{W}_i{}^\intercal \mathbf{f}$, is fed into softmax layer σ to obtain the final probabilistic output $\mathbf{p}_i(\mathbf{x}) = \sigma(\frac{1}{T}\mathbf{W}_i{}^\intercal \mathbf{f})$. Most previous MDA works train the classifier of domain i with only the labeled data in domain i. However, with only few labeled samples per source in MFDA, C_i is prone to overfit to $\mathcal{S}_i$. Thus we train each C_i with labeled data from all source domains using a standard cross-entropy loss:

$$\mathcal{L}_{\text{cls}}^{(i)} = \mathbb{E}_{(\mathbf{x},y)\in\bigcup \mathcal{S}_i} \mathcal{L}_{CE}(\mathbf{p}_i(\mathbf{x}), y) \tag{6.12}$$

One drawback of training each C_i using the same set of all labeled data is that the predictive behavior of each C_i will likely be quite similar, which greatly impairs the ensembling effect of multiple classifiers during test time. To build a classifier set $\{C_i\}$ with more diverse predictive behavior, we desire to make each C_i have slightly higher accuracy on domain i than on other domains.

Looking closer at a cosine classifier C with weight matrix $\mathbf{W}$, in order for it to have high performance, the k-th weight vector $\mathbf{w}^k$ needs to be a representative vector of the corresponding class k. To promote a more diverse set of $\{C_i\}$, we directly update $\mathbf{W}_i$ with prototypes computed from the corresponding support set $S^{(i)}$, computed in Sec. 6.3. Specifically, we estimate prototype of class k in domain i as:

$$\hat{\mathbf{w}}_i^k = \frac{1}{|S_k^{(i)}|} \sum_{\mathbf{x}\in S_k^{(i)}} \boldsymbol{V}_i^{\boldsymbol{s}}(\mathbf{x}), \tag{6.13}$$

where $S_k^{(i)} = \{\mathbf{x} | y = k, \forall(\mathbf{x}, y) \in S^{(i)}\}$, $\boldsymbol{V}_i^{\boldsymbol{s}}(\mathbf{x})$ returns the representation of $\mathbf{x}$ stored in the memory bank, and $\mathbf{w}_i^k$ is then updated with $\frac{\hat{\mathbf{w}}_i^k}{\|\hat{\mathbf{w}}_i^k\|}$ frequently.

Classifier-wise Mutual Information To learn better domain-invariant and discriminative features, we maximize the mutual information between the input and output of each classifier with unlabeled images across all domains. For classifier C_i, the mutual information can be written as

$$\mathcal{I}_i(y; \mathbf{x}) = \mathcal{H}(\bar{\mathbf{p}}_i) - \mathbb{E}_{\mathbf{x}}[\mathcal{H}(p(y|\mathbf{x}; \theta_i))], \tag{6.14}$$

where $p(y|\mathbf{x}; \theta_i)$ denotes the output of C_i on $\mathbf{x}$ and $\bar{\mathbf{p}}_i$ is a prior distribution $\mathbb{E}_{\mathbf{x}}[p(y|\mathbf{x}; \theta_i)]$. We can get the mutual information maximization objective as:

$$\mathcal{L}_{\text{MI}} = -\sum_{i=1}^{M} \mathcal{I}_i(y; \mathbf{x}), \;\; \mathbf{x} \in \bigcup_i \mathcal{S}_i^u \cup \mathcal{T} \tag{6.15}$$

MSFAN Learning

The MSFAN learning framework performs multi-domain prototypical self-supervised learning, support-set-based cross-domain similarity consistency, and multi-domain prototypical classifier learning. Together with the classifier update with Eq. 6.13, the overall training objective is:

$$\mathcal{L}_{\text{MSFAN}} = \mathcal{L}_{\text{cls}} + \lambda_{\text{mps}} \cdot \mathcal{L}_{\text{MPS}} + \lambda_{\text{ssc}} \cdot \mathcal{L}_{\text{SSC}} + \lambda_{\text{mi}} \cdot \mathcal{L}_{\text{MI}} \tag{6.16}$$

Global Max-Similarity-based Inference For target data during test time, we propose a new inference method based on the max-similarity across all classifiers. With normalized weights from all classifiers $\mathcal{W} = \{\mathbf{w}_i^c | 1 \leq i \leq M, 1 \leq c \leq n_c\}$, given a test example feature $\mathbf{f}_t$, the most similar weight vector is identified as $\mathbf{w}_{i^*,c^*} = \arg\max_{\mathbf{w} \in \mathcal{W}} \mathbf{f}_i^\intercal \cdot \mathbf{w}$, and c^* is the final prediction.

6.4 Experiments

Experimental Setting

Datasets. We evaluate our method (MSFAN) in multi-source few-shot setting on three standard domain adaptation benchmarks, Office [186], Office-Home [227], and DomainNet [171]. The labeled data in each domain are chosen following [119, 265], and each domain is in turn regarded as the target domain, while the others in the same dataset are considered as source domains. **Office** [186] is a real-world dataset for domain adaptation tasks. It contains 3 domains (Amazon, DSLR, Webcam) with 31 classes. Experiments are conducted with 1-shot and 3-shots source labels per class in this dataset. **Office-Home** [227] is a more difficult dataset than Office, which consists of 4 domains (Art, Clipart, Product, Real) in 65 classes. Following [119, 265], we look into the settings with 3% and 6% labeled source images per class, which means each class has 2 to 4 labeled images on average. **DomainNet** [171] is a large-scale domain adaptation benchmark. Since some domains and classes are noisy, we follow [188, 265] and use a subset containing four domains (Clipart, Painting, Real, Sketch) with 126 classes. We show results on settings with 1-shot and 3-shots source labels on this dataset.

Implementation Details. We use ResNet-101 [94] (for DomainNet) and ResNet-50 (for other datasets) pre-trained on ImageNet [183] as backbones for all baselines and MSFAN. To enable a fair comparison with [119] and [265], we replaced the last fully connected layer with a 512-dimension randomly initialized linear layer. We use k-means GPU implementation in faiss [115] for efficient clustering. We use SGD with a momentum of 0.9, a learning rate of 0.01, a batch size of 64. More implementation details can be found in the supplementary material.

Table 6.1: Adaptation accuracy (%) with 1 and 3 labeled samples per class on Office dataset.

	Office								
		1-shot				3-shot			
	Method	D,W→A	A,W→D	A,D→W	Avg	D,W→A	A,W→D	A,D→W	Avg
Source Only	Single-best	41.1	62.0	65.2	56.1	55.3	86.1	85.5	75.6
	Combined	53.4	66.5	69.2	63.0	63.5	86.9	86.0	78.8
Single-best DA	CDAN [145]	39.7	66.8	66.5	57.7	65.1	89.8	91.6	82.2
	MME [188]	23.1	62.4	60.9	48.8	60.2	91.4	89.7	80.4
	MDDIA [113]	55.6	79.5	84.4	73.2	70.3	93.2	93.3	85.6
	CDS [119]	52.0	57.4	59.0	56.1	67.6	81.3	86.0	78.3
	PCS [265]	76.1	91.8	90.6	86.2	76.4	**96.0**	94.1	88.8
Source-combined DA	CDAN [145]	52.3	72.7	73.3	66.1	67.8	85.7	88.5	80.7
	MME [188]	34.6	64.9	74.1	57.9	61.5	91.2	91.4	81.4
	MDDIA [113]	63.4	91.4	87.2	80.7	74.7	96.6	94.9	88.7
	CDS [119]	67.1	73.9	88.2	76.4	72.2	88.2	90.9	83.8
	PCS [265]	72.8	89.0	92.1	84.6	76.5	**96.0**	94.8	89.1
Multi-source DA	SImpAl [226]	58.5	72.5	71.7	67.6	65.0	85.3	86.7	79.0
	MFSAN [287]	48.9	64.7	66.0	59.9	64.7	82.7	87.9	78.4
	PMDA [284]	56.3	66.5	71.4	64.7	68.4	86.5	91.8	82.2
	MSFAN (Ours)	**76.3**	**94.4**	**92.6**	**87.8**	**77.7**	95.4	**95.8**	**89.6**

Results on MFDA

Baselines. We compare MSFAN with the following methods. **(1) Source-only**, i.e. train on the labeled data in source domains and test on the target domain directly. **(2) Single-source DA**, perform multi-source DA via single-source DA, including CDAN [145], MDDIA [113], MME [188]; as well as CDS [119] and PCS [265] which are the strongest single-source baselines specifically designed for single-source few-shot DA (FUDA). **(3) Multi-source DA**, assume multiple fully-labeled sources and are designed for MDA, including MFSAN [287], SImpAl [226] and ProtoMDA [284]. SImpAl [226] and ProtoMDA [284] are the most recent state-of-the-art works, and ProtoMDA also leverages prototypes for MDA. We re-run all baseline methods in the new MFDA setting (multi-source domain adaptation with few labels in each source), and compare with the proposed MSFAN.

Extensive experiments are performed on Office, Office-Home, and DomainNet, with the results shown in Table 6.1, 6.2, 6.3, respectively. From the results, we have the following observations: (1) For single-best, source-only outperforms some UDA methods under various scenarios, e.g. 56.1% vs. 48.8% in Office under 1-shot per class. A similar observation is obtained on source-combined, e.g. 40.1% vs. 31.3% in Office-Home under 1-shot per class. (2) Under MFDA, naively combining multiple sources and perform single-source DA can lead to worse performance than using a single domain, e.g. 42.3% vs. 44.6% with 1-shot per class on DomainNet for PCS, which is specifically designed for single-source few-shot DA. (3) Under MFDA, conventional MDA methods perform even worse than single-source DA methods, e.g. 65.6% vs. 68.7% with 6% labels per class on Office-Home. (4) MSFAN

Table 6.2: Adaptation accuracy (%) with 3% and 6% labeled samples per class on Office-Home dataset.

	Office-Home										
		3%					6%				
	Method	Ar,Pr,Rw →Cl	Cl,Pr,Rw →Ar	Cl,Ar,Rw →Pr	Cl,Ar,Pr →Rw	Avg	Ar,Pr,Rw →Cl	Cl,Pr,Rw →Ar	Cl,Ar,Rw →Pr	Cl,Ar,Pr →Rw	Avg
Source Only	Single-best	29.0	41.2	52.3	43.1	41.4	36.0	49.9	61.8	54.6	50.6
	Combined	42.2	55.3	63.6	64.1	56.3	45.3	60.4	70.5	70.9	61.8
Single-best DA	CDAN [145]	27.0	38.7	44.9	40.3	37.7	40.1	54.9	63.6	59.3	54.5
	MME [188]	29.0	39.3	52.0	44.9	41.3	37.3	54.9	66.8	61.3	55.1
	MDDIA [113]	29.5	47.1	56.4	51.0	46.0	37.1	58.2	68.4	64.5	57.1
	CDS [119]	37.8	51.6	53.8	51.0	48.6	45.3	63.7	68.6	65.2	60.7
	PCS [265]	52.5	66.0	**75.6**	73.9	67.0	54.7	67.0	76.6	75.2	68.4
Source-combined DA	CDAN [145]	42.6	52.3	64.5	63.2	55.7	51.1	67.0	74.2	73.3	66.4
	MME [188]	42.5	55.4	67.4	64.5	57.5	46.0	67.1	75.5	75.7	66.1
	MDDIA [113]	55.3	66.9	72.3	75.3	67.5	**57.3**	67.2	79.0	74.4	69.5
	CDS [119]	54.9	66.2	71.6	73.4	66.5	54.9	67.5	76.1	77.5	69.0
	PCS [265]	49.4	67.0	75.0	76.3	66.9	50.4	67.0	77.8	**79.4**	68.7
Multi-source DA	SImpAl [226]	46.8	56.7	65.1	66.6	58.8	49.3	62.1	71.7	73.0	64.1
	MFSAN [287]	39.9	46.6	58.9	55.6	50.3	44.5	53.7	65.4	64.2	57.0
	PMDA [284]	50.8	56.8	64.2	66.8	59.7	54.4	65.8	70.4	71.8	65.6
	MSFAN (Ours)	**55.6**	**68.4**	**75.6**	**76.6**	**69.1**	56.3	**68.7**	**79.3**	79.1	**70.9**

outperforms all baselines under all experimental settings. Especially, compared to state-of-the-art MDA methods, we can see that MSFAN outperforms them across all benchmarks with large improvements: 20.2% and 7.4% on Office, 9.4% and 5.3% on Office-Home, 16.2% and 9.0% on DomainNet.

Ablation Study and Analysis

We now investigate the effectiveness of each component in MSFAN on Office-Home. Table 6.4 shows that adding each component contributes to the final MFDA performance without any accuracy degradation. To qualitatively show the effectiveness of domain alignment with MSFAN, we plot the learned features with t-SNE [152] on the Ar,Cl,Pr→Rl setting in Office-Home. In the top row, color represents the domain of each sample; while in the bottom row, color represents the class of each sample. Compared to ImageNet pre-training and Source-combined, it qualitatively shows that MSFAN clusters samples with the same class in the feature space; thus, MSFAN favors more discriminative features. Also, the features from MSFAN are more closely aggregated than ImageNet pre-training and Source-combined, which demonstrates that MSFAN learns a better semantic structure of the datasets.

6.5 Related Work and Discussion

Single-source UDA Single-source UDA [84] aims to transfer knowledge from a fully-labeled source domain to an unlabeled target domain. Most UDA methods focus on feature

Table 6.3: Adaptation accuracy (%) comparison with 1 and 3 labeled samples per class on DomainNet.

	DomainNet										
		1-shot					3-shot				
	Method	P,R,S →C	C,R,S →P	C,P,S →R	C,P,R →S	Avg	P,R,S →C	C,R,S →P	C,P,S →R	C,P,R →S	Avg
Source Only	Single Best	18.4	30.6	28.9	16.7	23.7	30.2	44.2	49.8	24.2	34.4
	Combined	30.8	49.4	43.3	36.9	40.1	45.3	57.4	64.7	42.6	50.0
Single-best DA	CDAN [145]	16.0	25.7	19.5	12.9	18.5	30.0	40.1	40.8	17.1	29.3
	MME [188]	16.0	29.2	26.0	13.4	21.2	25.1	46.5	50.0	20.1	32.6
	MDDIA [113]	18.0	30.6	27.4	15.9	23.0	41.4	50.7	52.9	23.1	38.2
	CDS [119]	16.7	24.4	15.9	13.4	17.6	35.0	43.8	36.8	31.1	32.9
	PCS [265]	39.0	51.7	38.8	39.8	42.3	45.2	59.1	66.6	41.9	51.0
Source-combined DA	CDAN [145]	25.7	33.0	40.0	26.4	31.3	47.8	54.1	65.6	49.1	49.6
	MME [188]	20.0	45.3	52.5	13.0	32.7	44.2	62.7	73.9	51.8	53.1
	MDDIA [113]	44.0	46.4	49.6	37.1	44.3	56.3	59.3	70.3	51.3	56.3
	CDS [119]	42.2	53.3	55.4	38.5	47.4	50.2	61.5	71.8	47.3	55.6
	PCS [265]	36.2	53.0	56.4	32.8	44.6	45.6	61.2	74.3	41.3	53.4
Multi-source DA	SImpAl [226]	48.0	40.3	45.7	35.3	42.3	51.5	47.4	68.8	45.3	51.1
	MFSAN [287]	41.6	33.5	38.8	29.6	35.9	43.5	42.3	63.2	41.1	45.2
	PMDA [284]	49.3	42.2	45.0	34.8	42.8	52.2	52.5	71.3	47.6	53.3
	MSFAN (Ours)	**57.3**	**68.7**	**64.8**	**45.2**	**59.0**	**57.8**	**65.5**	**75.8**	**53.6**	**62.3**

Table 6.4: Performance contribution of each part in MSFAN framework on Office-Home.

Office-Home 3%	Ar,Pr,Rw →Cl	Cl,Pr,Rw →Ar	Cl,Ar,Rw →Pr	Cl,Ar,Pr →Rw	Avg
Source-combined	42.2	55.3	63.6	64.1	56.3
+ Multi-classifier	44.9	55.5	65.1	68.2	58.4
+ $\mathcal{L}_{\text{MPS}}$	52.1	66.6	66.5	75.1	65.1
+ $\mathcal{L}_{\text{MI}}$	55.2	**68.4**	75.0	76.4	68.8
+ $\mathcal{L}_{\text{SSC}}$ (MSFAN)	**55.6**	**68.4**	**75.6**	**76.6**	**69.1**

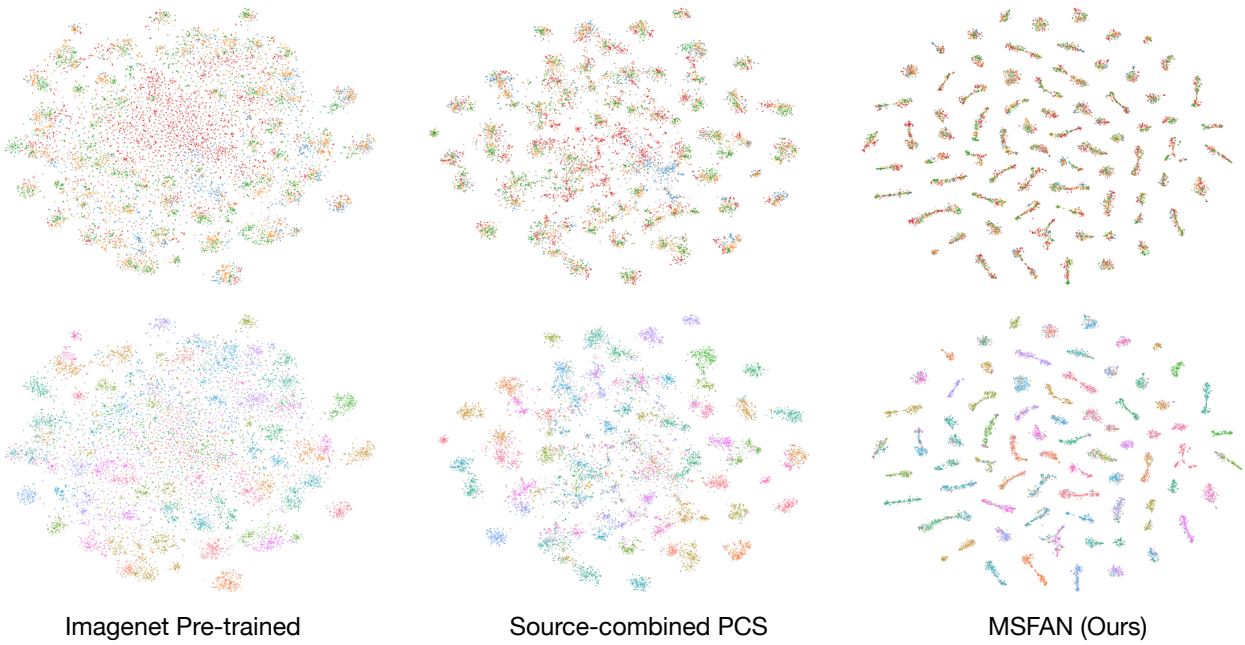


Figure 6.2: Visualization of baselines and our method via t-SNE on OfficeHome (Ar,Pr,Rw→Cl). Top row: Blue, Orange, Green and Red represents domain Art, Real, Product and Clipart, respectively. Bottom row: Coloring represents the class of each sample. Features from MSFAN are better-aligned across domains compared to other methods.

distribution alignment. Discrepancy-based methods utilize different metric learning schemas to diminish the domain shift between source and target. Inspired by the two-sample test [87], Maximum Mean Discrepancy (MMD) is leveraged to perform domain alignment in various methods [147, 222, 148, 75, 233, 146]. Sun *et al.* [205] and Zhuo *et al.* [289] further proposed to align second-order statistics of source and target features. After Generative Adversarial Network [83] was proposed, more works [68, 221, 100, 250, 145, 197] leverage a domain discriminator to encourage domain confusion by an adversarial objective. Recently, image translation methods [285, 139] have been adopted to further improve domain adaptation by performing domain alignment at pixel-level [100, 17, 184, 159, 263, 189, 199]. Instead of explicit feature alignment, Saito *et al.* [188] perform entropy optimization for adaptation. Though these methods achieves high performance, few of them consider the practical scenario of adapting from multiple sources.

Multi-source Domain Adaptation (MDA). MDA [209, 279] assumes the availability of multiple fully-labeled sources and aims to transfer knowledge to an unlabeled target domain. Various theoretical analyses [10, 42, 153, 98] have been proposed to support existing MDA algorithms. Early MDA methods usually either learn a shared feature space for all domains [63, 207, 61, 62], or combine pre-learned source classifier predictions to get final

predictions with an ensembling method. With the development of deep nerual networks, more deep-learning-based MDA methods are proposed, such as DCTN [252], M3SDA [171], MDAN [272], MFSAN [287], MDDA [280]. All these MDA methods aim to minimize this domain shift using auxiliary distribution alignment objectives. SImpAl [226] is proposed to perform implicit domain alignment with pseudo-labeling without additional training objectives for adaptation. Recently, ProtoMDA [284] is proposed to use prototypes for MDA and achieves state-of-the-art performance. While these methods have full supervision on the source domains, we focus on a new adaptation setting with only few labels in each source domain.

Self-supervised Learning (SSL) for Domain Adaptation. SSL is a subset of unsupervised learning where supervision is automatically generated from the data [114, 51, 267, 162, 78, 231]. One of the most common strategies for SSL is handcrafting auxiliary pretext tasks predicting future, missing or contextual information [267, 128, 51, 52, 162, 170, 78, 57]. Reconstruction was first utilized as a self-supervised task in some early works [77, 76], in which source and target share the same encoder to extract domain-invariant features. In [21], solving jigsaw puzzle [162] was leveraged as a self-supervision task to solve domain adaptation and generalization. Sun *et al.* [210] further proposed to perform adaptation by jointly learning multiple self-supervision tasks. Recently, contrastive learning has achieved state-of-the-art performance on representation learning [95, 88, 30, 35, 31, 177, 249, 134, 4, 22]. Based on instance discrimination [245] and prototypical contrastive learning, Kim *et al.* [119] and Yue *et al.* [265] proposed cross-domain SSL approaches for adaptation with few source labels. SSL has also been incorporated for adaptation in other fields, including point cloud recognition [1], medical imaging [108], action segmentation [28], robotics [110], facial tracking [257], *etc.*

6.6 Conclusion

Traditional Multi-source Domain Adaptation assumes multiple fully-labeled source domains. In this chapter, we investigate Multi-source Few-shot Domain Adaptation, a new domain adaptation task that is more practical and challenging, where each source domain only has a very small fraction of labeled samples. We proposed a novel framework, termed Multi-Source Few-shot Adaptation Network (MSFAN), that performs multi-domain prototypical self-supervised learning, support-set-based cross-domain similarity consistency, and multi-domain prototypical classifier learning. We perform extensive experiments on multiple benchmark datasets, which demonstrates the superiority of MSFAN over previous state-of-the-art methods.

Chapter 7

A LiDAR Point Cloud Generator: Generating Free Labels

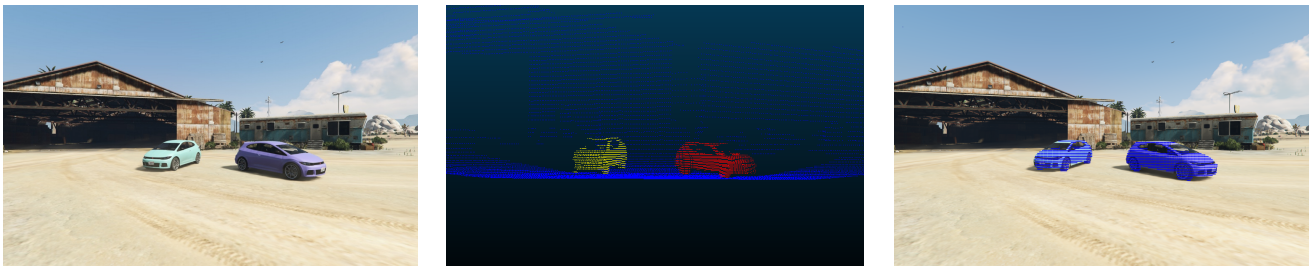
7.1 Chapter Overview

In this chapter, we present a LiDAR simulation environment in autonomous driving. The LiDAR simulator can be used to generate synthetic LiDAR point clouds, enabling a line of research in LiDAR point cloud segmentation adaptation. In addition, the simulator can be used to test robustness of neural networks.

7.2 Introduction

Autonomous driving is an important area in robotics and requires accurate and reliable perception of the environment [182, 25, 243]. As 3D data has some superiority in information representation over 2D data [70, 112], 3D LiDAR (Light Detection And Ranging) sensors are playing an increasingly important role of all the environment sensors for autonomous driving. On the one hand, 3D LiDAR sensors can provide direct distance measurements that allow detection of all kinds of obstacles, and their resolution and field of view exceed radar [161, 283] and ultrasonic sensors [23, 157]. On the other hand, LiDAR sensors are robust under a variety of conditions: day or night, with or without glare and shadows [241]. While LiDAR point clouds contain accurate depth measurement of the environment, navigation of autonomous vehicles also relies on correct understanding of the semantics of the environment. Most of the LiDAR-based perception tasks, such as semantic segmentation [49, 53, 241] and drivable area detection [143, 65], require significant amount of point-level labels for training and/or validation. Such annotation, however, is usually very expensive.

To facilitate the manual annotation process, much work has been done on interactive annotation. Annotation methods have been proposed for labeling 3D point clouds of both indoor scenes [200] and outdoor driving scenes [73]. These methods utilize little computer assistance during the annotation process and thus need a significant amount of human effort.



(a) (b) (c)

Figure 7.1: Sample data extracted from an in-game scene. (a): Image of the scene; (b): Extracted point cloud from the same scene; (c): Point cloud of car mapped to image after registration (Blue dots) matches car in image.

In [125, 224], approaches have been proposed to enhance the human-machine interaction to improve annotation efficiency. In [240, 196], annotation suggestions for indoor RGBD scenes are proposed by the system that are interactively corrected or refined by the user. In order to provide faster interactive labeling rates, Boyko *et al.* [19] proposed a group annotation approach for labeling objects in 3D LiDAR scans. Active learning has also been introduced in the annotation process to train a classifier with fewer interactions [118, 216], yet it requires users to interact with examples one-by-one. Other frameworks further take into account the risk of mislabeling and cost of annotation. In [239], Welinder *et al.* proposed a model of the labeling process and dynamically chooses which images will be labeled next in order to achieve a desired level of confidence.

Recently, video games have been used for creating large-scale ground truth data for training purposes. In [179], a video game is used to generate ground truth semantic segmentation for the synthesized in-game images. However, human effort is still required in the annotation process. In [116], the same game engine is used to generate ground truth 2D bounding boxes of objects in the images. Richter *et al.* [178] further extended the work of [179] so that various ground truth information (e.g. semantic segmentation, semantic instance segmentation, and optical flow) can be extracted from the game engine. In addition, many driving simulation environments [13, 91, 56] have been built in order to obtain various kinds of labeled data for autonomous driving purposes. Many of these work [116, 179, 178, 56] show the effectiveness of synthetic data in image-based learning tasks by showing improved performance after training with additional synthetic data. However, little work has been done on extracting annotated 3D LiDAR point clouds from simulators, not to mention showing the efficacy of the synthetic point clouds during the training process of neural networks. Note that even if we could provide large amounts of training data, it is still almost impossible for any algorithms to achieve 100% accuracy. For cyber-physical systems (CPS) used for safety-critical purposes, such as autonomous driving, verifying correctness is extremely important, but also very challenging [195, 198, 194, 66]. Dreossi et al. [58, 60] have proposed a framework to systematically analyze Convolutional Neural Networks (CNNs) used in objection detec-

tion in autonomous driving systems. However, the framework only takes into account cars from direct front/back view and thus has a very limited modification space. In addition, each background image needs to be manually annotated, making it expensive to generate a dataset with large diversity. To the best of our knowledge, no similar work has been done on LiDAR point clouds. In this chapter, we propose an extraction-annotation-CNN testing framework based on a popular video game. The main contributions of this chapter are as follows:

- The first published LiDAR point cloud simulation framework for autonomous driving (to our knowledge).
- The framework can automatically extract point-cloud data with ground truth labels together with the corresponding image frame of the in-game scene, as shown in Figure 7.1.
- The framework can do automatic registration between collected point clouds and images which can then be used together for sensor fusion tasks, e.g. inferring depth information from RGB images.
- Users can construct specified scenarios in the framework interactively and the collected data (point clouds and images) can then be used to systematically test, analyze and improve LiDAR-based and/or image-based learning algorithms for autonomous driving.

We conduct experiments on a Convolutional Neural Network (CNN)-based model for 3D LiDAR point cloud segmentation using the data collected from the proposed framework. The experiments show 1) significantly improved performance on KITTI dataset [73] after retraining with additional synthetic LiDAR point clouds, and 2) efficacy of using the data collected from user-configured scenes in the framework to test, analyze and improve the performance of the neural network. The performance improvements come from the fact that the data collected in the rich virtual world contains a lot of information that the neural network failed to learn from the limited amount of original training samples.

7.3 Technical Approach

In-Game Simulation Setup and Method for Data Collection

We choose to utilize the rich virtual world in Grand Theft Auto V (GTA-V), a popular video game, to obtain simulated point clouds as well as captured in-game images with high fidelity[1]. Our framework is based on DeepGTAV[2], which uses Script Hook V[3] as a plugin.

[1]The publisher of GTA-V allows non-commercial use of footage of gameplay [179]

[2]`https://github.com/aitorzip/DeepGTAV`

[3]`http://www.dev-c.com/gtav/scripthookv/`

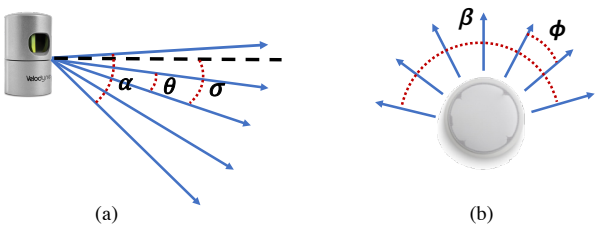


Figure 7.2: Sample configurable parameters of the virtual LiDAR. (a) shows front view of the virtual LiDAR: black dotted line is the horizontal line, α is the vertical field of view (FOV), θ is the vertical resolution, σ is the pitch angle; (b) shows top view of the virtual LiDAR, β is the horizontal FOV, and ϕ is the horizontal resolution.

In order to simulate realistic driving scenes, an ego car is used in the game with a virtual LiDAR scanner mounted atop, and it is set to drive autonomously in the virtual world with the AI interface provided in Script Hook V. While the car drives on a street, the system collects LiDAR point clouds and captures the game screen, simultaneously. We place the virtual LiDAR scanner and the game camera at the same position in the virtual 3D space. This set-up offers two advantages: 1) a sanity check can be easily done on the collected data, since point clouds and corresponding images must be consistent; 2) registration between the game camera and the virtual LiDAR scanner can be done automatically, and then collected point clouds and scene images can be combined together as training dataset for neural networks for sensor fusion tasks. Details of the proposed registration method will be described in Section 7.3.

Ray casting is used to simulate each laser ray emitted by the virtual LiDAR scanner. The ray casting API takes as input the 3D coordinates of the starting and ending point of the ray, and returns the 3D coordinates of the first point the ray hits. This point is used, with another series of API function calls, to calculate, among other data, the distance of the point, the category and instance ID of the object hit by the ray, thus allowing automatic annotation on the collected data.

In our framework, users can provide configurations of the LiDAR scanner including vertical field of view (FOV), vertical resolution, horizontal FOV, horizontal resolution, pitch angle, maximum range of laser rays, and scanning frequency. Some of the configurable parameters are shown in Figure 7.2.

Automatic Registration Method

The goal of the registration process is to find the corresponding pixel in the image for each LiDAR point. In our framework, the registration process can be done automatically by the system based on the parameters of the camera and LiDAR scanner. In addition, the centers of the camera and LiDAR scanner are set to the same position in the virtual world, making

the registration projection similar to the camera perspective projection model, as shown in Figure 7.3.

The problem is formulated as follows: for a certain laser ray with *azimuth* angle ϕ and *zenith* angle θ, calculate the index (i, j) of the corresponding pixel on image. $\mathcal{F}_c$, $\mathcal{F}_o$, P, P' and P_{far} are 3D coordinates of a) center of camera/LiDAR scanner, b) center of camera near clipping plane. c) point first hit by the virtual laser ray (in red), d) pixel on image corresponding to P, and e) a point far away in the laser direction (required as an input argument to the ray casting API), respectively. m and n are the width and height of the near clipping plane. γ is 1/2 vertical FOV of camera while ψ is 1/2 vertical FOV of the LiDAR scanner. Note that LiDAR scanner FOV is usually smaller than camera FOV, since there is usually no object in the top part of the image, and thus emitting laser to open space is not necessary. After a series of 3D geometry calculation, we can get:

$$\begin{aligned} i &= \frac{R_m}{m} \cdot (f \cdot \tan\gamma \cdot \frac{m}{n} - \frac{f}{\cos\theta} \cdot \tan\phi), \\ j &= \frac{R_n}{n} \cdot (f \cdot \tan\gamma + f \cdot \tan\theta), \end{aligned} \tag{7.1}$$

where $f = \left\| \overrightarrow{\mathcal{F}_c \mathcal{F}_o} \right\|$, and (R_m, R_n) is the pixel resolution of the image/near clipping plane.

Further, as an input argument to the ray casting API, the 3D coordinates of P_{far} are also required. Using similar 3D geometry calculations, we obtain:

$$\begin{aligned} P' &= \mathcal{F}_c + f \cdot \overrightarrow{x_c} - \frac{f}{\cos\theta} \cdot \tan\phi \cdot \overrightarrow{y_c} - f \cdot \tan\theta \cdot \overrightarrow{z_c}, \\ P_{far} &= \mathcal{F}_c + k \cdot (P' - \mathcal{F}_c), \end{aligned} \tag{7.2}$$

where k is a large coefficient, and $\overrightarrow{x_c}, \overrightarrow{y_c}, \overrightarrow{z_c}$ are unit vectors of the camera axis in the world coordinate system.

An example of the registration result is shown in Figure 7.1. After simulation, both image and point cloud of the specified in-game scene are collected by the framework (Figure 7.1 (a, b)). Then with the proposed registration method, we map all the points with category "Car" to the corresponding image. As shown in Figure 7.1 (c), the mapped car point cloud (blue dots) matches the car in the image fairly accurately.

Configurable In-game Scene

Besides the auto-driving mode for large-scale data collection, our framework offers a configurable mode, where the user can configure desired in-game scenes and collect data from them. One advantage of configurable scenes is generating training data of driving scenes that are dangerous or rare in real world. Another advantage is that we can systematically sample the modification space (e.g. number of cars, position and orientation of a car) of an in-game scene. The data can then be used to test a neural network, expose its vulnerabilities and improve its performance through retraining. Our framework offers a large modification

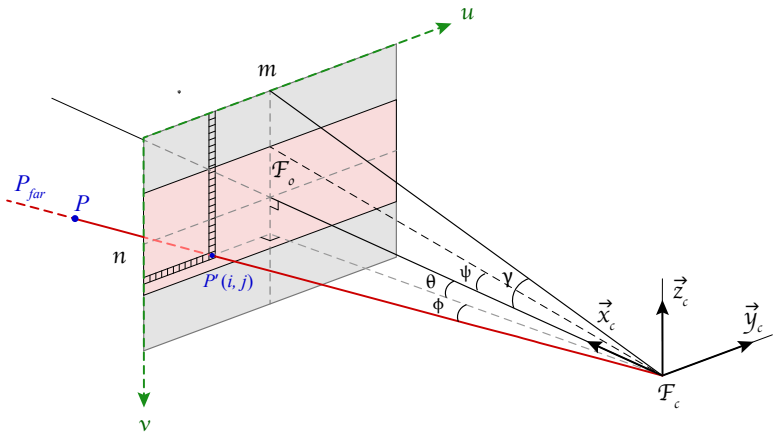


Figure 7.3: Projection for Registration. $\mathcal{F}_o$ is the center of the near clipping plane of the camera; $\mathcal{F}_c$ is the center of the camera and of the LiDAR scanner; the red line is the laser ray and P is the point hit by the ray; the calibrated on-image point has pixel index (i, j) and 3D coordinates P'; γ is 1/2 the camera vertical FOV and ψ is 1/2 the LiDAR vertical FOV; ϕ and θ are the *azimath* and *zenith* angles of the laser ray.

space of the in-game scene. As shown in Figure 7.4, the user can specify and change 8 dimensions of in-game scene: car model, car location, car orientation, number of cars, scene background, color of car, weather, and time of day. The first 5 dimensions affect both LiDAR point cloud and scene image, while the last three dimensions affect only the scene image. An example of sampling is shown in Figure 7.5, where the scenes are only sampled from the spatial dimensions (X, Y) with only one car in each scene. X and Y are the location offset of the car relative to the camera/LiDAR location in the left-right and forward-backward directions. Figure7.5 (b) shows collected point cloud of the samples shown in Figure7.5 (a). The red points represent car points while the blue points represent the scene background. The collected point clouds match the scenes well thus allowing the use of the data to test neural networks systematically.

7.4 Experiments and Results

We performed experiments to show the efficacy of our data synthesis framework: 1) Data collected by the framework can be used in the training phase and help improve the validation accuracy; 2) Collected data can be used to systematically test a neural network and improve its performance via retraining.

Evaluation Metrics

Our experiments are performed on the task of LiDAR point cloud segmentation; specifically, given a point cloud detected by a LiDAR sensor, we wish to perform point-wise classification,

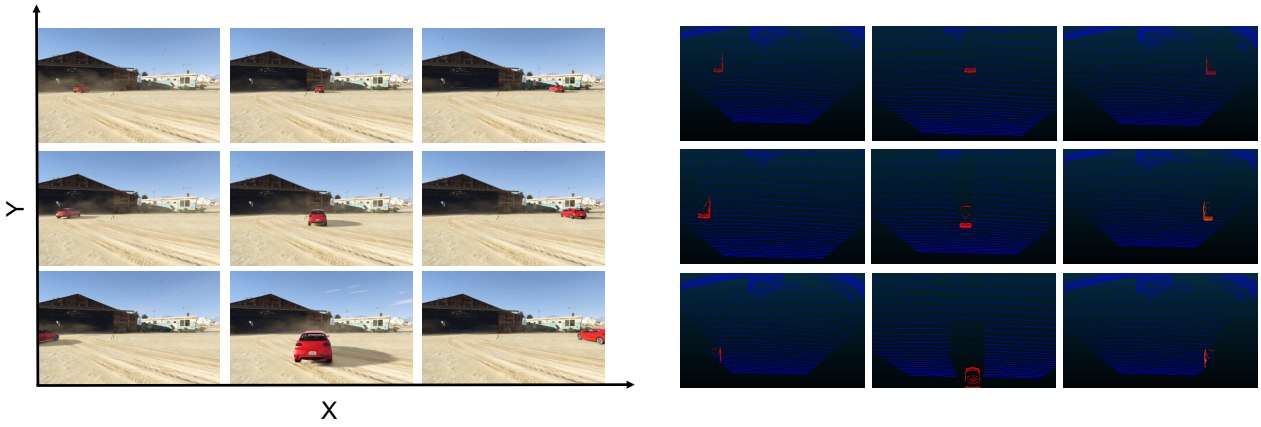
Figure 7.4: Modification dimensions of the framework with image in center showing the reference scene.

as shown in Figure 7.6. This task is an essential step for autonomous vehicles to perceive and understand the environment, and navigate accordingly.

To evaluate the accuracy of the point cloud segmentation algorithm, we compute *Intersection-over-Union* (IoU), *Precision* and *Recall* as:

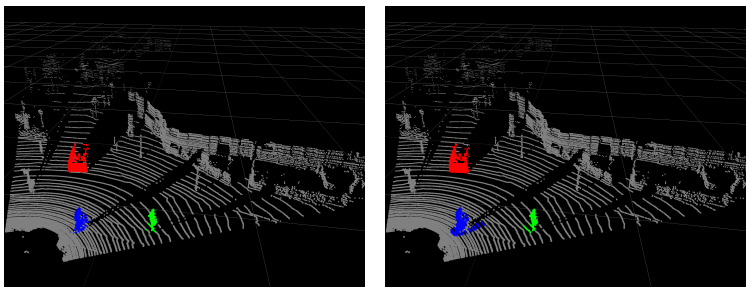
$$IoU_c = \frac{|\mathcal{P}_c \cap \mathcal{G}_c|}{|\mathcal{P}_c \cup \mathcal{G}_c|}, Pr_c = \frac{|\mathcal{P}_c \cap \mathcal{G}_c|}{|\mathcal{P}_c|}, Recall_c = \frac{|\mathcal{P}_c \cap \mathcal{G}_c|}{|\mathcal{G}_c|}.$$

Here, $\mathcal{P}_c$ denotes the set of points that our model predicted to be of class-c, $\mathcal{G}_c$ denotes the ground-truth set of points belonging to class-c, and $|\cdot|$ denotes the cardinality of a set. *Precision* and *Recall* measures accuracy with regard to false positives and false negatives, respectively; while IoU takes both into account. For this, $\boldsymbol{IoU}$ is used as the primary accuracy metric in our experiments.



(a) (b)

Figure 7.5: Scenes with one car sampled from spatial dimensions and corresponding point cloud. (a) shows the scene image while changing the location of the car on X (left-right) and Y (forward-backward) directions; (b) shows point clouds (red for car and blue for background) of scenes in (a).



Ground truth segmentation Predicted segmentation

Figure 7.6: LiDAR point cloud segmentation

Experimental Setup

Our analysis is based on SqueezeSeg [241], a convolutional neural network based model for point cloud segmentation. To collect the real-word dataset, we used LiDAR point cloud data from the KITTI dataset and converted its 3D bounding box labels to point-wise labels. Since KITTI dataset only provides reliable 3D bounding boxes for front-view LiDAR point clouds, we limit the horizontal field of view (FOV) to the forward-facing 90°. This way, we obtained 10,848 LiDAR scans with manual labels. We used 8,057 scans for training and 2,791 scans for validation. Each point in a KITTI LiDAR scan has 3 cartesian coordinates (x, y, z) and an intensity value, which measures the amplitude of the laser signal returned. Although the intensity measurement as an extra input feature is beneficial to improve the segmentation accuracy, simulating the intensity measurement is very difficult and not supported in our

Table 7.1: Segmentation Performance Comparison on the Car Category. Only data used in the first row has Intensity channel. All numbers are in percentage.

	Precision	Recall	**IoU**
KITTI w/ Intensity	66.7	95.4	64.6
KITTI w/o Intensity	58.9	95.0	57.1
GTA-V only	30.4	86.6	29.0
KITTI w/o Intensity + GTA-V	69.6	92.8	**66.0**

current framework. Therefore we excluded intensity as an input feature to the neural network for GTA-V synthetic LiDAR data. We use NVIDIA TITAN X GPUs for the experiments during both the training and validation phases.

Experimental Results

Synthetic Data for Training: For the first set of experiments, we used our data synthesis framework to generate 8,585 LiDAR point cloud scans in autonomous-driving scenes. The generated data contain (x, y, z) measurements but do not contain intensity. The horizontal FOV of the collected point clouds are set to be 90° to match the setting of KITTI point clouds described in Section 7.4.

To quantify the effect of training the model with synthetic data, we first trained two models on the KITTI training set with intensity included and excluded, and validated on the KITTI validation set. The performance is shown in the first 2 rows of Table 7.1 as the baseline. The model with intensity achieved better result. Then we trained another model with only GTA-V synthetic data. As shown in the third row of Table 7.1, the performance drops a lot. This is mostly because the distributions of the synthetic dataset and KITTI dataset are quite different. Therefore, through training purely on synthetic dataset, it is hard for the neural network to learn all the required details for the KITTI dataset, which might be missing or insufficient in the synthetic training dataset. Finally, we combined the KITTI data and GTA-V data together as the training set and train another model. As shown in the last row of Table 7.1, the performance gets improved significantly, almost 9% better than the accuracy achieved only using real-world data. Despite the loss of the intensity channel, the GTA+KITTI dataset gives better accuracy (66.0%) than if intensity is included (64.6%). This demonstrates the efficacy of the synthetic data extracted in our framework.

Neural Network Testing and Robustness Enhancement: For the second set of experiments, we first used our framework to systematically test SqueezeSeg. As an illustrative experiment, we only performed sampling in the car location X-Y dimensions as in Figure 7.5, rather than the whole modification space. 555 scenes were sampled to test SqueezeSeg, with the IoU results shown in Figure 7.7. The blue and green dots show the car locations resulting in low IoU. Most of the "blind spot" are locations far from the LiDAR scanner, but

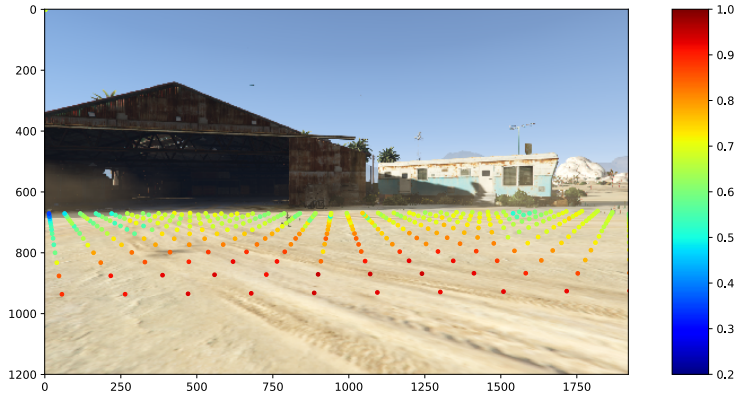


Figure 7.7: IoU scatter with the change of car location

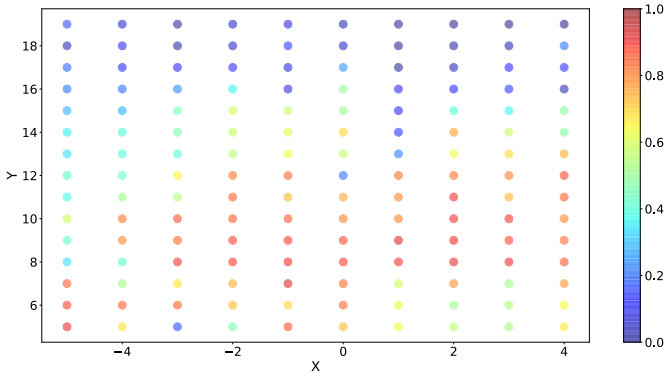


Figure 7.8: mIoU map of the validation set before retraining

there are also closer locations that result in low IoU scores. Close locations with low IoUs are dangerous in autonomous driving, since they can mislead the decision-making system of the autonomous vehicles and result in immediate accident.

Further experiments are then done to show the efficacy of using generated synthetic data to possibly improve performance and robustness of the network in the modification space. We synthesized totally 2,250 LiDAR point cloud scans in 15 different scene backgrounds. In each scene background, only one car is placed with the same orientation as the camera view. We obtained 150 point cloud scans in each scene background by changing the position of the car (X, Y) in the sampled space: $S = \{(x, y) \mid x \in \{-5, \cdots, 4\}, y \in \{5, \cdots, 19\}\}$, where X, Y are respectively the left-right and forward-backward offset relative to the position of the camera. For each scene background, the position and orientation of the camera were fixed.

We split the collected point cloud scans based on the scene background. 1200 point cloud

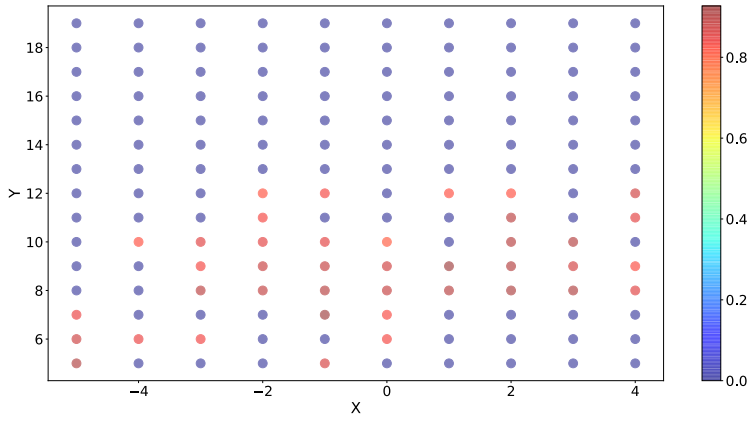


Figure 7.9: mIoU map of the validation set after selection with mIoU less than 0.65 set to 0. All the point clouds in the retraining set $\mathcal{R}$ corresponding to the blue positions in the new mIoU map will be added to the original training set.

scans in the first 8 backgrounds are used as validation set $\mathcal{V}$, and the rest 1050 scans from the other 7 backgrounds, which we call retraining set $\mathcal{R}$, are used for retraining purpose. First, we train a neural network with purely KITTI data and do evaluation on the synthetic 1200 scans in the validation set $\mathcal{V}$. We define *mean IoU (mIoU)* for each point in the 15×10 X-Y modification space as averaging IoUs over all the scene backgrounds in $\mathcal{V}$:

$$mIoU(i,j) = \frac{1}{n}\sum_{k=1}^{n} IoU(i,j,k),$$

where n is the number of scene backgrounds ($n = 8$ in this experiment), (i, j) is in $\{(i, j) \mid i \in [-5, 4], j \in [5, 19], i, j \in \mathbb{Z}\}$ and $IoU(i, j, k)$ refers to the IoU of the point cloud scan sampled at (i, j) in the X-Y modification with the k_{th} scene background.

The mIoU map of the validation set is computed, as shown in Figure 7.8. We can see that the pre-trained network performs poorly on positions that are far away, at the boundary of the FOV. But more surprisingly, we also observed that on some positions that are fairly close to the ego-vehicle, e.g. (-3, 5), the mIoU scores are also very low. Detection errors at such near distance can be very dangerous.

Based on the mIoU map, we choose positions with an mIoU smaller than a threshod to form a retraining set, as shown in Figure 7.9. Then all the point clouds in the retraining set $\mathcal{R}$ with a selected position are added to the original training set. After the retraining process, we re-evaluate the validation set $\mathcal{V}$, with the new mIoU map shown in Figure 7.10. As the figure shows, at almost all the close-to-center positions originally with low mIoU, the neural network performs much better than before the retraining. In order to visualize the performance improvements better, we plot the mIoU improvement after the retraining process for each position. The mIoU improvements are sorted and plotted in Figure 7.11. We see that after retraining, performance on point clouds at most of the positions gets much

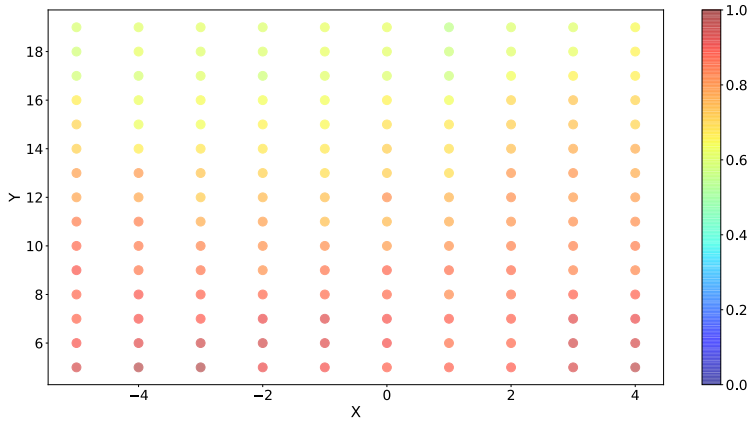


Figure 7.10: mIoU map of the validation set after retraining

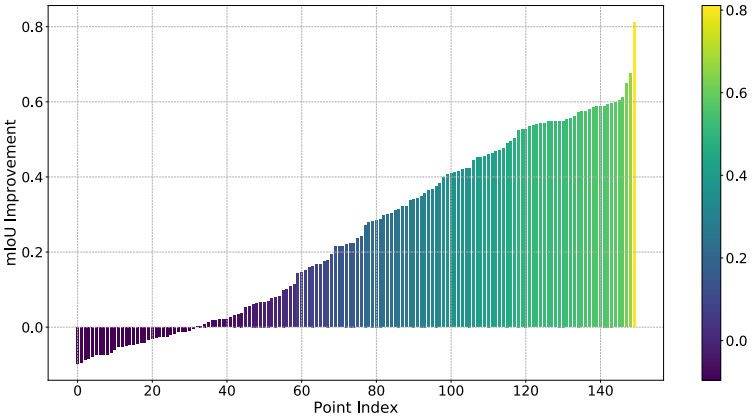


Figure 7.11: mIoU improvements in ascending order for all 150 positions

better, with slightly degraded performance at only a small fraction of positions. Meanwhile, the performance on KITTI dataset remained almost the same with IoU changing from 60.8% to 60.6%. These experiments show the efficacy of using synthetic data from user-configured scenes of the proposed framework to test, analyze and improve the performance of neural networks through retraining.

7.5 Conclusions and Future Work

In this chapter, we proposed a framework that synthesizes annotated LiDAR point clouds from a virtual world in a game, with a method to automatically calibrate the point cloud and scene image. Our framework can be used to: 1) obtain a large amount of annotated point cloud data, which can then be used to help neural network training; 2) systematically

test, analyze and improve performance of neural networks for tasks such as point cloud segmentation. Experiments show that for a point cloud segmentation task, synthesized data help improve the validation accuracy (IoU) by 9% on a real-world benchmark. Furthermore, the systematical sampling and testing framework can help us to identify potential weakness/blind spots of our neural network model and fix them. Our future works will focus on two directions: 1) Experiments show that the intensity channel in LiDAR point clouds is crucial for LiDAR-based perception tasks, but current framework does not support intensity synthesis. We plan to explore techniques to simulate realistic LiDAR intensity. 2) The effectiveness of using synthetic data to train perception algorithms is usually limited by domain shift. We plan to explore domain adaptation methods to diminish the gap between virtual world and real world.

Chapter 8

Conclusion

The presence of domain shift and the need for less annotations of datasets drive the research in transferring knowledge across domains. This dissertation presented a series of methods towards learning transferable representations under various scenarios. In Chapter 2, we introduced a framework to learn models for a target domain with only few labels in the source domain. In Chapter 3, we proposed to learn a segmentation model that could generalize well to multiple unseen domains. In Chapter 4, We learned segmentation models for a target domain from multiple source domains. In Chapter 5, we proposed a method for textual sentiment adaptation from multiple labeled sources. In Chapter 6, we further proposed a framework extending the proposed method in Chapter 2 for adaptation from multiple sources with few labels. In Chapter 7, we presented a LiDAR point cloud simulation environment to generate synthetic point clouds with annotations, which further promotes further research on LiDAR point cloud segmentation adaptation.

In our work, we demonstrated that there is no single adaptation/generalization method that works best for all scenarios. For instance, naively combining multiple source domains and directly applying single source domain adaptation can lead to worse performance than using a single source domain [279]; directly applying a model, adapted targeting one target domain, to another target domain can lead to degraded performance. In Chapters 2-5, We proposed different methods for learning transferable representation across different settings, different tasks, and different data modalities. In Chapter 6 which targets a combined setting (multi-source few-shot adaptation) of Chapter 2 (few-shot adaptation) and Chapter 4 (multi-source adaptation), we demonstrated that the previous proposed methods are extensible and can be easily combined to solve other similar domain transfer settings.

We believe there are many takeaways that can help researchers design adaptation/generalization methods for other knowledge transfer settings in the future. In Chapter 2, we demonstrated that although transferring knowledge with few annotations is hard, some techniques can significantly boost performance. Specifically, learning cross-domain aligned clustered representations is essential, and the cluster centers can well approximate the cosine classifier weights. In Chapter 3, we showed that for segmentation generalization, domain randomization can be used to learn domain-invariant representations. In addition, applying

representation consistency across domains and scales can further boost the representation learning. In Chapter 4, we showed that learning an aggregated representation space across multiple sources is essential for effective adaptation. In Chapter 5, we showed that curriculum instance-level adaptation can be an promising direction for multi-source adaptation. In Chapter 6, we demonstrated that our previous methods can be easily extended to more complex problem settings. With the work in Chapter 7, we would like to note that building simulation environments to get free-labeled synthetic data, together with effective adaptation/generalization approaches, is important to alleviate the need for manual annotations.

One interesting future direction on learning transferable representations, is to use the knowledge in foundation models [15] to help the model transfer better to diverse unseen domains. Recently, foundation models have shown surprising ability in learning knowledge for various downstream tasks. Yet, it is still unclear how to effectively leverage foundation models to aid the knowledge transfer process of models to unseen domains. This is essential to learn models that can generalize to dynamically changing environments. There are other more realistic problem settings that requires more attention. For example, in the real world, 1) we might be able to provide some annotations in the target domain, which can significantly boost the adaptation performance; 2) the annotations available in the source domain might not be evenly distributed, and the long-tailed labeled distribution can be a challenge to learn an unbiased classifier; 3) the target data may be streaming in with the distribution dynamically changing. Further research need to be conducted targeting these more challenging scenarios, either extending previous methods or designing novel approaches.

Bibliography

[1] Idan Achituve, Haggai Maron, and Gal Chechik. "Self-Supervised Learning for Domain Adaptation on Point-Clouds". In: *arXiv preprint arXiv:2003.12641* (2020).

[2] Sk Miraj Ahmed et al. "Unsupervised Multi-source Domain Adaptation Without Access to Source Data". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2021).

[3] Mikel Artetxe, Gorka Labaka, and Eneko Agirre. "Learning bilingual word embeddings with (almost) no bilingual data". In: *Annual Meeting of the Association for Computational Linguistics*. 2017, pp. 451–462.

[4] Yuki M. Asano, Christian Rupprecht, and Andrea Vedaldi. "Self-labelling via simultaneous clustering and representation learning". In: *International Conference on Learning Representations (ICLR)*. 2020.

[5] Philip Bachman, R Devon Hjelm, and William Buchwalter. "Learning representations by maximizing mutual information across views". In: *Advances in Neural Information Processing Systems*. 2019, pp. 15535–15545.

[6] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. "Segnet: A deep convolutional encoder-decoder architecture for image segmentation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.12 (2017), pp. 2481–2495.

[7] Yogesh Balaji, Swami Sankaranarayanan, and Rama Chellappa. "MetaReg: Towards Domain Generalization using Meta-Regularization". In: *Advances in Neural Information Processing Systems*. 2018, pp. 1006–1016.

[8] Carlos J Becker, Christos M Christoudias, and Pascal Fua. "Non-linear domain adaptation with boosting". In: *Advances in Neural Information Processing Systems*. 2013, pp. 485–493.

[9] Serge Belongie and Pietro Perona. "Visipedia circa 2015". In: *Pattern Recognition Letters* (Dec. 10, 2015). URL: `http://www.sciencedirect.com/science/article/pii/S0167865515004092`.

[10] Shai Ben-David et al. "A theory of learning from different domains". In: *Machine Learning* 79.1-2 (2010), pp. 151–175.

[11] David Berthelot et al. "Mixmatch: A holistic approach to semi-supervised learning". In: *Advances in Neural Information Processing Systems*. 2019, pp. 5049–5059.

[12] Rhys Biddle et al. "Leveraging Sentiment Distributions to Distinguish Figurative From Literal Health Reports on Twitter". In: *The Web Conference*. 2020, pp. 1217–1227.

[13] Daniel Biedermann, Matthias Ochs, and Rudolf Mester. "Evaluating visual ADAS components on the COnGRATS dataset". In: *IEEE Intelligent Vehicles Symposium*. 2016, pp. 986–991.

[14] Piotr Bojanowski et al. "Enriching Word Vectors with Subword Information". In: *Transactions of the Association for Computational Linguistics* 5 (2017), pp. 135–146.

[15] Rishi Bommasani et al. "On the opportunities and risks of foundation models". In: *arXiv preprint arXiv:2108.07258* (2021).

[16] Konstantinos Bousmalis et al. "Domain separation networks". In: *Advances in Neural Information Processing Systems*. 2016, pp. 343–351.

[17] Konstantinos Bousmalis et al. "Unsupervised pixel-level domain adaptation with generative adversarial networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 3722–3731.

[18] Konstantinos Bousmalis et al. "Using simulation and domain adaptation to improve efficiency of deep robotic grasping". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 4243–4250.

[19] Aleksey Boyko and Thomas Funkhouser. "Cheaper by the Dozen: Group Annotation of 3D Data". In: *ACM Symposium on User Interface Software and Technology*. 2014, pp. 33–42.

[20] Xuetao Cao. "COVID-19: immunopathology and its implications for therapy". In: *Nature reviews immunology* 20.5 (2020), pp. 269–270.

[21] Fabio M Carlucci et al. "Domain generalization by solving jigsaw puzzles". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 2229–2238.

[22] Mathilde Caron et al. "Unsupervised learning of visual features by contrasting cluster assignments". In: *Advances in Neural Information Processing Systems* 33 (2020).

[23] Alessio Carullo and Marco Parvis. "An ultrasonic sensor for distance measurement in automotive applications". In: *IEEE Sensors journal* 1.2 (2001), p. 143.

[24] Rita Chattopadhyay et al. "Multisource domain adaptation and its application to early detection of fatigue". In: *ACM Transactions on Knowledge Discovery from Data* 6.4 (2012), p. 18.

[25] Chenyi Chen et al. "Deepdriving: Learning affordance for direct perception in autonomous driving". In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 2722–2730.

[26] Yi-Hsin Chen et al. "No more discrimination: Cross city adaptation of road scene segmenters". In: *IEEE International Conference on Computer Vision*. 2017, pp. 1992–2001.

[27] Liang-Chieh Chen et al. "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs". In: *IEEE transactions on pattern analysis and machine intelligence* 40.4 (2017), pp. 834–848.

[28] Min-Hung Chen et al. "Action segmentation with joint self-supervised temporal domain adaptation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 9454–9463.

[29] Minmin Chen et al. "Marginalized denoising autoencoders for domain adaptation". In: *International Conference on Machine Learning*. 2012, pp. 1627–1634.

[30] Ting Chen et al. "A simple framework for contrastive learning of visual representations". In: *arXiv preprint arXiv:2002.05709* (2020).

[31] Ting Chen et al. "Big Self-Supervised Models are Strong Semi-Supervised Learners". In: *arXiv preprint arXiv:2006.10029* (2020).

[32] Wei-Yu Chen et al. "A closer look at few-shot classification". In: *arXiv preprint arXiv:1904.04232* (2019).

[33] Xi Chen et al. "FiDo: Ubiquitous Fine-Grained WiFi-based Localization for Unlabelled Users via Domain Adaptation". In: *The Web Conference*. 2020, pp. 23–33.

[34] Xilun Chen et al. "Multi-Source Cross-Lingual Model Transfer: Learning What to Share". In: *Annual Meeting of the Association for Computational Linguistics*. 2019, pp. 3098–3112.

[35] Xinlei Chen et al. "Improved baselines with momentum contrastive learning". In: *arXiv preprint arXiv:2003.04297* (2020).

[36] Yuhua Chen, Wen Li, and Luc Van Gool. "ROAD: Reality oriented adaptation for semantic segmentation of urban scenes". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 7892–7901.

[37] Zhenpeng Chen et al. "Emoji-powered representation learning for cross-lingual sentiment classification". In: *International World Wide Web Conference*. 2019, pp. 251–262.

[38] Kyunghyun Cho et al. "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation". In: *Conference on Empirical Methods in Natural Language Processing*. 2014, pp. 1724–1734.

[39] Özgün Çiçek et al. "3D U-Net: learning dense volumetric segmentation from sparse annotation". In: *International Conference on Medical Image Computing and Computer Assisted Intervention*. 2016, pp. 424–432.

[40] Dan Ciregan, Ueli Meier, and Jurgen Schmidhuber. "Multi-column deep neural networks for image classification". In: *2012 IEEE Conference on Computer Vision and Pattern Recognition*. 2012, pp. 3642–3649.

[41] Marius Cordts et al. "The cityscapes dataset for semantic urban scene understanding". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 3213–3223.

[42] Koby Crammer, Michael Kearns, and Jennifer Wortman. "Learning from Multiple Sources." In: *Journal of Machine Learning Research* 9.8 (2008).

[43] Ekin D Cubuk et al. "AutoAugment: Learning Augmentation Policies from Data". In: *arXiv preprint arXiv:1805.09501* (2018).

[44] Abhishek Das et al. "Embodied question answering". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Vol. 5. 2018, p. 6.

[45] Jia Deng et al. "Imagenet: A large-scale hierarchical image database". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2009, pp. 248–255.

[46] Jan Deriu et al. "Leveraging large amounts of weakly supervised data for multi-language sentiment classification". In: *International World Wide Web Conference*. 2017, pp. 1045–1052.

[47] Jacob Devlin et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *Annual Conference of the North American Chapter of the Association for Computational Linguistics*. 2019, pp. 4171–4186.

[48] Terrance DeVries and Graham W Taylor. "Dataset augmentation in feature space". In: *arXiv preprint arXiv:1702.05538* (2017).

[49] Ayush Dewan, Gabriel L Oliveira, and Wolfram Burgard. "Deep Semantic Classification for 3D LiDAR Data". In: *arXiv preprint arXiv:1706.08355* (2017).

[50] Zhengming Ding, Ming Shao, and Yun Fu. "Incomplete multisource transfer learning". In: *IEEE Transactions on Neural Networks and Learning Systems* 29.2 (2018), pp. 310–323.

[51] Carl Doersch, Abhinav Gupta, and Alexei A Efros. "Unsupervised visual representation learning by context prediction". In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1422–1430.

[52] Carl Doersch and Andrew Zisserman. "Multi-task self-supervised visual learning". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 2051–2060.

[53] David Dohan, Brian Matejek, and Thomas Funkhouser. "Learning hierarchical semantic segmentations of lidar data". In: *International Conference on 3D Vision*. 2015, pp. 273–281.

[54] Jeff Donahue et al. "Decaf: A deep convolutional activation feature for generic visual recognition". In: *International conference on machine learning*. 2014, pp. 647–655.

[55] Li Dong et al. "Adaptive recursive neural network for target-dependent twitter sentiment classification". In: *Annual Meeting of the Association for Computational Linguistics*. 2014, pp. 49–54.

[56] Alexey Dosovitskiy et al. "CARLA: An Open Urban Driving Simulator". In: *Proceedings of the 1st Annual Conference on Robot Learning*. 2017, pp. 1–16.

[57] Alexey Dosovitskiy et al. "Discriminative unsupervised feature learning with convolutional neural networks". In: *Advances in neural information processing systems*. 2014, pp. 766–774.

[58] Tommaso Dreossi, Alexandre Donzé, and Sanjit A Seshia. "Compositional Falsification of Cyber-Physical Systems with Machine Learning Components". In: *NASA Formal Methods Symposium*. 2017, pp. 357–372.

[59] Tommaso Dreossi et al. "Counterexample-guided data augmentation". In: *arXiv preprint arXiv:1805.06962* (2018).

[60] Tommaso Dreossi et al. "Systematic testing of convolutional neural networks for autonomous driving". In: *arXiv preprint arXiv:1708.03309* (2017).

[61] Lixin Duan, Dong Xu, and Shih-Fu Chang. "Exploiting web images for event recognition in consumer videos: A multiple source domain adaptation approach". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2012, pp. 1338–1345.

[62] Lixin Duan, Dong Xu, and Ivor Wai-Hung Tsang. "Domain adaptation from multiple sources: A domain-dependent regularization approach". In: *IEEE Transactions on Neural Networks and Learning Systems* 23.3 (2012), pp. 504–518.

[63] Lixin Duan et al. "Domain adaptation from multiple sources via auxiliary classifiers". In: *International Conference on Machine Learning*. 2009, pp. 289–296.

[64] Aysegul Dundar et al. "Domain stylization: A strong, simple baseline for synthetic to real image domain adaptation". In: *arXiv:1807.09384* (2018).

[65] R. Fernandes et al. "Road Detection Using High Resolution LIDAR". In: *IEEE Vehicle Power and Propulsion Conference*. 2014, pp. 1–6.

[66] Daniel Fremont et al. "Scenic: Language-based scene generation". In: *arXiv preprint arXiv:1809.09310* (2018).

[67] Chuang Gan, Tianbao Yang, and Boqing Gong. "Learning attributes equals multi-source domain generalization". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 87–97.

[68] Yaroslav Ganin and Victor Lempitsky. "Unsupervised domain adaptation by backpropagation". In: *International conference on machine learning*. PMLR. 2015, pp. 1180–1189.

[69] Yaroslav Ganin et al. "Domain-adversarial training of neural networks". In: *The Journal of Machine Learning Research* 17.1 (2016), pp. 2096–2030.

[70] Yue Gao et al. "3-D object retrieval and recognition with hypergraph analysis". In: *IEEE Transactions on Image Processing* 21.9 (2012), pp. 4290–4303.

[71] Vivien Sainte Fare Garnot and Loic Landrieu. "Metric-Guided Prototype Learning". In: *arXiv preprint arXiv:2007.03047* (2020).

[72] Andreas Geiger, Philip Lenz, and Raquel Urtasun. "Are we ready for autonomous driving? the kitti vision benchmark suite". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2012, pp. 3354–3361.

[73] Andreas Geiger et al. "Vision meets Robotics: The KITTI Dataset". In: *International Journal of Robotics Research* 32.11 (2013), pp. 1231–1237.

[74] Spiros V Georgakopoulos et al. "Convolutional neural networks for toxic comment classification". In: *Hellenic Conference on Artificial Intelligence*. 2018, pp. 1–6.

[75] Muhammad Ghifary, W Bastiaan Kleijn, and Mengjie Zhang. "Domain adaptive neural networks for object recognition". In: *Pacific Rim international conference on artificial intelligence*. Springer. 2014, pp. 898–904.

[76] Muhammad Ghifary et al. "Deep reconstruction-classification networks for unsupervised domain adaptation". In: *European Conference on Computer Vision*. Springer. 2016, pp. 597–613.

[77] Muhammad Ghifary et al. "Domain generalization for object recognition with multi-task autoencoders". In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 2551–2559.

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

[79] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. "Domain adaptation for large-scale sentiment classification: A deep learning approach". In: *International Conference on Machine Learning*. 2011, pp. 513–520.

[80] Boqing Gong, Kristen Grauman, and Fei Sha. "Reshaping visual datasets for domain adaptation". In: *Advances in Neural Information Processing Systems*. 2013, pp. 1286–1294.

[81] Lin Gong, Benjamin Haines, and Hongning Wang. "Clustered model adaption for personalized sentiment analysis". In: *International World Wide Web Conference*. 2017, pp. 937–946.

[82] Lin Gong and Hongning Wang. "When sentiment analysis meets social network: A holistic user behavior modeling in opinionated data". In: *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2018, pp. 1455–1464.

[83] Ian Goodfellow et al. "Generative adversarial nets". In: *Advances in neural information processing systems*. 2014, pp. 2672–2680.

[84] Raghuraman Gopalan, Ruonan Li, and Rama Chellappa. "Domain adaptation for object recognition: An unsupervised approach". In: *2011 international conference on computer vision*. IEEE. 2011, pp. 999–1006.

[85] Raghuraman Gopalan, Ruonan Li, and Rama Chellappa. "Unsupervised adaptation across domain shifts by generating intermediate data representations". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36.11 (2014), pp. 2288–2302.

[86] Yves Grandvalet and Yoshua Bengio. "Semi-supervised learning by entropy minimization". In: *Advances in neural information processing systems*. 2005, pp. 529–536.

[87] Arthur Gretton et al. "A kernel two-sample test". In: *The Journal of Machine Learning Research* 13.1 (2012), pp. 723–773.

[88] Jean-Bastien Grill et al. "Bootstrap your own latent: A new approach to self-supervised learning". In: *arXiv preprint arXiv:2006.07733* (2020).

[89] Varun Gulshan et al. "Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs". In: *Jama* 316.22 (2016), pp. 2402–2410.

[90] Jiang Guo, Darsh Shah, and Regina Barzilay. "Multi-Source Domain Adaptation with Mixture of Experts". In: *Conference on Empirical Methods on Natural Language Processing*. 2018, pp. 4694–4703.

[91] Vladimir Haltakov, Christian Unger, and Slobodan Ilic. "Framework for generation of synthetic ground truth data for driver assistance applications". In: *German Conference on Pattern Recognition*. 2013, pp. 323–332.

[92] Stephanie A Harmon et al. "Artificial intelligence for the detection of COVID-19 pneumonia on chest CT using multinational datasets". In: *Nature communications* 11.1 (2020), pp. 1–7.

[93] Haodi He et al. "MLSeg: Image and Video Segmentation as Multi-Label Classification and Selected-Label Pixel Classification". In: *European conference on computer vision*. 2022.

[94] Kaiming He et al. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[95] Kaiming He et al. "Momentum contrast for unsupervised visual representation learning". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 9729–9738.

[96] Kaiming He et al. "Spatial pyramid pooling in deep convolutional networks for visual recognition". In: *European conference on computer vision*. 2014, pp. 346–361.

[97] Sepp Hochreiter and Jürgen Schmidhuber. "Long short-term memory". In: *Neural Computation* 9.8 (1997), pp. 1735–1780.

[98] Judy Hoffman, Mehryar Mohri, and Ningshan Zhang. "Algorithms and theory for multiple-source adaptation". In: *Advances in Neural Information Processing Systems*. 2018, pp. 8246–8256.

[99] Judy Hoffman et al. "CyCADA: Cycle Consistent Adversarial Domain Adaptation". In: *International Conference on Machine Learning (ICML)*. 2018.

[100] Judy Hoffman et al. "Cycada: Cycle-consistent adversarial domain adaptation". In: *International conference on machine learning*. PMLR. 2018, pp. 1989–1998.

[101] Judy Hoffman et al. "Fcns in the wild: Pixel-level adversarial and constraint-based adaptation". In: *arXiv preprint arXiv:1612.02649* (2016).

[102] Weixiang Hong et al. "Conditional generative adversarial network for structured domain adaptation". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 1335–1344.

[103] Zhang-Wei Hong et al. "Virtual-to-real: learning to control in visual semantic segmentation". In: *International Joint Conference on Artificial Intelligence*. 2018, pp. 4912–4920.

[104] Lanqing Hu et al. "Duplex generative adversarial network for unsupervised domain adaptation". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 1498–1507.

[105] Minqing Hu and Bing Liu. "Mining and summarizing customer reviews". In: *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2004, pp. 168–177.

[106] Gao Huang et al. "Densely connected convolutional networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 4700–4708.

[107] Haoshuo Huang, Qixing Huang, and Philipp Krahenbuhl. "Domain transfer through deep activation matching". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 590–605.

[108] Sontje Ihler et al. "Self-Supervised Domain Adaptation for Patient-Specific, Real-Time Tissue Tracking". In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer. 2020, pp. 54–64.

[109] Shatha Jaradat. "Deep cross-domain fashion recommendation". In: *ACM Conference on Recommender Systems*. 2017, pp. 407–410.

[110] Rae Jeong et al. "Self-supervised sim-to-real adaptation for visual robotic manipulation". In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 2718–2724.

[111] I-Hong Jhuo et al. "Robust visual domain adaptation with low-rank reconstruction". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2012, pp. 2168–2175.

[112] Rongrong Ji et al. "Mining compact bag-of-patterns for low bit rate mobile visual search". In: *IEEE Transactions on Image Processing* 23.7 (2014), pp. 3099–3113.

[113] Xiang Jiang et al. "Implicit Class-Conditioned Domain Alignment for Unsupervised Domain Adaptation". In: *International Conference on Machine Learning*. 2020.

[114] Longlong Jing and Yingli Tian. "Self-supervised visual feature learning with deep neural networks: A survey". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2020).

[115] Jeff Johnson, Matthijs Douze, and Hervé Jégou. "Billion-scale similarity search with GPUs". In: *arXiv preprint arXiv:1702.08734* (2017).

[116] Matthew Johnson-Roberson et al. "Driving in the Matrix: Can virtual worlds replace human-generated annotations for real world tasks?" In: *IEEE International Conference on Robotics and Automation*. 2017, pp. 746–753.

[117] Guoliang Kang et al. "Contrastive adaptation network for unsupervised domain adaptation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 4893–4902.

[118] Ashish Kapoor et al. "Active learning with gaussian processes for object categorization". In: *IEEE International Conference on Computer Vision*. 2007, pp. 1–8.

[119] Donghyun Kim et al. "Cross-domain Self-supervised Learning for Domain Adaptation with Few Source Labels". In: *arXiv preprint arXiv:2003.08264* (2020).

[120] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization". In: *International Conference on Learning Representations*. 2015.

[121] Diederik P. Kingma and Jimmy Ba. "Adam: A Method for Stochastic Optimization". In: *CoRR* abs/1412.6980 (2014).

[122] Alexander Kirillov et al. "Panoptic segmentation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 9404–9413.

[123] Svetlana Kiritchenko, Xiaodan Zhu, and Saif M Mohammad. "Sentiment analysis of short informal texts". In: *Journal of Artificial Intelligence Research* 50 (2014), pp. 723–762.

[124] Eric Kolve et al. "AI2-THOR: An interactive 3d environment for visual AI". In: *arXiv preprint arXiv:1712.05474* (2017).

[125] Regis Kopper, Felipe Bacim, and Doug A Bowman. "Rapid and accurate 3D selection by progressive refinement". In: *IEEE Symposium on 3D User Interfaces*. 2011, pp. 67–74.

[126] Wouter Marco Kouw and Marco Loog. "A review of domain adaptation without target labels". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2019).

[127] Shanu Kumar et al. "Adversarial adaptation of scene graph models for understanding civic issues". In: *International World Wide Web Conference*. 2019, pp. 2943–2949.

[128] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. "Learning representations for automatic colorization". In: *European conference on computer vision*. Springer. 2016, pp. 577–593.

[129] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. "Beyond bags of features: spatial pyramid matching for recognizing natural scene categories". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2006, pp. 2169–2178.

[130] Joseph Lemley, Shabab Bazrafkan, and Peter Corcoran. "Smart Augmentation Learning an Optimal Data Augmentation Strategy." In: *IEEE Access* (2017), pp. 5858–5869.

[131] Da Li et al. "Deeper, broader and artier domain generalization". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 5542–5550.

[132] Da Li et al. "Learning to generalize: Meta-learning for domain generalization". In: *Thirty-Second AAAI Conference on Artificial Intelligence*. 2018.

[133] Haoliang Li et al. "Domain generalization with adversarial feature learning". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 5400–5409.

[134] Junnan Li et al. "Prototypical Contrastive Learning of Unsupervised Representations". In: *arXiv preprint arXiv:2005.04966* (2020).

[135] Junnan Li et al. "Prototypical Contrastive Learning of Unsupervised Representations". In: *ICLR* (2021).

[136] Shaohua Li et al. "Laplacian-steered neural style transfer". In: *Proceedings of the 2017 ACM on Multimedia Conference*. ACM. 2017, pp. 1716–1724.

[137] Yunsheng Li, Lu Yuan, and Nuno Vasconcelos. "Bidirectional Learning for Domain Adaptation of Semantic Segmentation". In: *arXiv preprint arXiv:1904.10620* (2019).

[138] Guosheng Lin et al. "Efficient piecewise training of deep structured models for semantic segmentation". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 3194–3203.

[139] Ming-Yu Liu and Oncel Tuzel. "Coupled generative adversarial networks". In: *Advances in neural information processing systems*. 2016, pp. 469–477.

[140] Qiao Liu et al. "Content attention model for aspect based sentiment analysis". In: *International World Wide Web Conference*. 2018, pp. 1023–1032.

[141] Ruijun Liu et al. "A survey of sentiment analysis based on transfer learning". In: *IEEE Access* 7 (2019), pp. 85401–85412.

[142] Ziwei Liu et al. "Semantic image segmentation via deep parsing network". In: *IEEE International Conference on Computer Vision*. 2015, pp. 1377–1385.

[143] Ziyi Liu et al. "Detecting Drivable Area for Self-driving Cars: An Unsupervised Approach". In: *arXiv 1705.00451* (2017).

[144] Jonathan Long, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 3431–3440.

[145] Mingsheng Long et al. "Conditional adversarial domain adaptation". In: *Advances in Neural Information Processing Systems*. 2018, pp. 1640–1650.

[146] Mingsheng Long et al. "Deep transfer learning with joint adaptation networks". In: *International conference on machine learning*. PMLR. 2017, pp. 2208–2217.

[147] Mingsheng Long et al. "Learning transferable features with deep adaptation networks". In: *International conference on machine learning*. PMLR. 2015, pp. 97–105.

[148] Mingsheng Long et al. "Unsupervised domain adaptation with residual transfer networks". In: *Advances in neural information processing systems*. 2016, pp. 136–144.

[149] Christos Louizos et al. "The variational fair autoencoder". In: *arXiv:1511.00830* (2015).

[150] Washington Luiz et al. "A feature-oriented sentiment rating for mobile app reviews". In: *International World Wide Web Conference*. 2018, pp. 1909–1918.

[151] Yawei Luo et al. "Taking A Closer Look at Domain Shift: Category-level Adversaries for Semantics Consistent Domain Adaptation". In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019.

[152] Laurens van der Maaten and Geoffrey Hinton. "Visualizing data using t-SNE". In: *Journal of machine learning research* 9.Nov (2008), pp. 2579–2605.

[153] Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. "Domain adaptation with multiple sources". In: *Advances in neural information processing systems* 21 (2008), pp. 1041–1048.

[154] Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. "Multiple source adaptation and the rényi divergence". In: *arXiv preprint arXiv:1205.2628* (2012).

[155] Ishan Misra and Laurens van der Maaten. "Self-supervised learning of pretext-invariant representations". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 6707–6717.

[156] Saif Mohammad, Svetlana Kiritchenko, and Xiaodan Zhu. "NRC-Canada: Building the State-of-the-Art in Sentiment Analysis of Tweets". In: *International Workshop on Semantic Evaluation (SemEval)*. 2013, pp. 321–327.

[157] Frank Moosmann, Oliver Pink, and Christoph Stiller. "Segmentation of 3D lidar data in non-flat urban environments using a local convexity criterion". In: *IEEE Intelligent Vehicles Symposium*. 2009, pp. 215–220.

[158] Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. "Domain generalization via invariant feature representation". In: *International Conference on Machine Learning*. 2013, pp. 10–18.

[159] Zak Murez et al. "Image to image translation for domain adaptation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 4500–4509.

[160] Gerhard Neuhold et al. "The mapillary vistas dataset for semantic understanding of street scenes". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 4990–4999.

[161] Felix Nobis et al. "A deep learning-based radar and camera sensor fusion architecture for object detection". In: *2019 Sensor Data Fusion: Trends, Solutions, Applications (SDF)*. IEEE. 2019, pp. 1–7.

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

[163] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. "Representation learning with contrastive predictive coding". In: *arXiv preprint arXiv:1807.03748* (2018).

[164] Sinno Jialin Pan and Qiang Yang. "A survey on transfer learning". In: *IEEE Transactions on Knowledge and Data Engineering* 22.10 (2010), pp. 1345–1359.

[165] Xingang Pan et al. "Two at Once: Enhancing Learning and Generalization Capacities via IBN-Net". In: *European Conference on Computer Vision*. 2018, pp. 484–500.

[166] Bo Pang and Lillian Lee. "Opinion mining and sentiment analysis". In: *Foundations and Trends in Information Retrieval* 2.1-2 (2008), pp. 1–135.

[167] Bo Pang and Lillian Lee. "Seeing Stars: Exploiting Class Relationships for Sentiment Categorization with Respect to Rating Scales". In: *Annual Meeting of the Association for Computational Linguistics*. 2005, pp. 115–124.

[168] Adam Paszke et al. "Pytorch: An imperative style, high-performance deep learning library". In: *Advances in neural information processing systems*. 2019, pp. 8026–8037.

[169] Vishal M Patel et al. "Visual domain adaptation: A survey of recent advances". In: *IEEE Signal Processing Magazine* 32.3 (2015), pp. 53–69.

[170] Deepak Pathak et al. "Context encoders: Feature learning by inpainting". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 2536–2544.

[171] Xingchao Peng et al. "Moment matching for multi-source domain adaptation". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pp. 1406–1415.

[172] Xingchao Peng et al. "Visda: The visual domain adaptation challenge". In: *arXiv preprint arXiv:1710.06924* (2017).

[173] Aayush Prakash et al. "Structured Domain Randomization: Bridging the Reality Gap by Context-Aware Synthetic Data". In: *arXiv preprint arXiv:1810.10093* (2018).

[174] Qiao Qian et al. "Learning tag embeddings and tag-specific composition functions in recursive neural network". In: *Annual Meeting of the Association for Computational Linguistics*. 2015, pp. 1365–1374.

[175] Ievgen Redko et al. "Optimal transport for multi-source domain adaptation under target shift". In: *International Conference on Artificial Intelligence and Statistics*. 2019, pp. 849–858.

[176] Joseph Redmon et al. "You only look once: Unified, real-time object detection". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 779–788.

[177] Colorado J Reed et al. "Self-Supervised Pretraining Improves Self-Supervised Pretraining". In: *arXiv preprint arXiv:2103.12718* (2021).

[178] Stephan R Richter, Zeeshan Hayder, and Vladlen Koltun. "Playing for benchmarks". In: *IEEE International Conference on Computer Vision*. 2017, pp. 2232–2241.

[179] Stephan R Richter et al. "Playing for data: Ground truth from computer games". In: *European Conference on Computer Vision*. 2016, pp. 102–118.

[180] Matthew Riemer et al. "Learning to Learn without Forgetting By Maximizing Transfer and Minimizing Interference". In: *International Conference on Learning Representations*. 2019.

[181] German Ros et al. "The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 3234–3243.

[182] Juan Rosenzweig and Michael Bartl. "A review and analysis of literature on autonomous driving". In: *E-Journal Making-of Innovation* (2015), pp. 1–57.

[183] Olga Russakovsky et al. "Imagenet large scale visual recognition challenge". In: *International journal of computer vision* 115.3 (2015), pp. 211–252.

[184] Paolo Russo et al. "From source to target and back: symmetric bi-directional adaptive gan". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 8099–8108.

[185] Fereshteh Sadeghi and Sergey Levine. "CAD2RL: Real single-image flight without a single real image". In: *arXiv preprint arXiv:1611.04201* (2016).

[186] Kate Saenko et al. "Adapting visual category models to new domains". In: *European conference on computer vision*. Springer. 2010, pp. 213–226.

[187] Kuniaki Saito et al. "Maximum classifier discrepancy for unsupervised domain adaptation". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 3723–3732.

[188] Kuniaki Saito et al. "Semi-supervised domain adaptation via minimax entropy". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pp. 8050–8058.

[189] Swami Sankaranarayanan et al. "Generate to adapt: Aligning domains using generative adversarial networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 8503–8512.

[190] Swami Sankaranarayanan et al. "Learning from synthetic data: Addressing domain shift for semantic segmentation". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018.

[191] Ikuro Sato, Hiroki Nishimura, and Kensuke Yokoi. "Apac: Augmented pattern classification with neural networks". In: *arXiv preprint arXiv:1505.03229* (2015).

[192] Gabriele Schweikert et al. "An empirical analysis of domain adaptation algorithms for genomic sequence analysis". In: *Advances in Neural Information Processing Systems*. 2009, pp. 1433–1440.

[193] Alice Schoenauer Sebag et al. "Multi-Domain Adversarial Learning". In: *International Conference on Learning Representations*. 2019.

[194] Sanjit A Seshia et al. "Formal specification for deep neural networks". In: *International Symposium on Automated Technology for Verification and Analysis*. Springer. 2018, pp. 20–34.

[195] Sanjit A. Seshia, Dorsa Sadigh, and S. Shankar Sastry. "Towards Verified Artificial Intelligence". In: *ArXiv e-prints* (2016). arXiv: 1606.08514.

[196] Tianjia Shao et al. "An Interactive Approach to Semantic Modeling of Indoor Scenes with an RGBD Camera". In: *ACM Transactions on Graphics* 31.6 (2012), 136:1–136:11.

[197] Jian Shen et al. "Wasserstein distance guided representation learning for domain adaptation". In: *arXiv preprint arXiv:1707.01217* (2017).

[198] Jay Shenoy et al. "A customizable dynamic scenario modeling and data generation platform for autonomous driving". In: *arXiv preprint arXiv:2011.14551* (2020).

[199] Ashish Shrivastava et al. "Learning from simulated and unsupervised images through adversarial training". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 2107–2116.

[200] Nathan Silberman et al. "Indoor segmentation and support inference from rgbd images". In: *European Conference on Computer Vision*. 2012, pp. 746–760.

[201] PY Simard, D Steinkraus, and JC Platt. "Best practices for convolutional neural networks applied to visual document analysis". In: *Document Analysis and Recognition, 2003. Proceedings. Seventh International Conference on*. 2003, pp. 958–963.

[202] K. Simonyan and A. Zisserman. "Very Deep Convolutional Networks for Large-Scale Image Recognition". In: *CoRR* abs/1409.1556 (2014).

[203] Richard Socher et al. "Recursive deep models for semantic compositionality over a sentiment treebank". In: *Conference on Empirical Methods in Natural Language Processing*. 2013, pp. 1631–1642.

[204] Richard Socher et al. "Semi-supervised recursive autoencoders for predicting sentiment distributions". In: *Conference on Empirical Methods on Natural Language Processing*. 2011, pp. 151–161.

[205] Baochen Sun, Jiashi Feng, and Kate Saenko. "Correlation alignment for unsupervised domain adaptation". In: *Domain Adaptation in Computer Vision Applications*. Springer, 2017, pp. 153–171.

[206] Baochen Sun, Jiashi Feng, and Kate Saenko. "Return of frustratingly easy domain adaptation". In: *AAAI Conference on Artificial Intelligence*. 2016, pp. 2058–2065.

[207] Qian Sun et al. "A two-stage weighting framework for multi-source domain adaptation". In: *Advances in Neural Information Processing Systems*. 2011, pp. 505–513.

[208] Shi-Liang Sun and Hong-Lei Shi. "Bayesian multi-source domain adaptation". In: *International Conference on Machine Learning and Cybernetics*. Vol. 1. 2013, pp. 24–28.

[209] Shiliang Sun, Honglei Shi, and Yuanbin Wu. "A survey of multi-source domain adaptation". In: *Information Fusion* 24 (2015), pp. 84–92.

[210] Yu Sun et al. "Unsupervised domain adaptation through self-supervision". In: *arXiv preprint arXiv:1909.11825* (2019).

[211] Martin Sundermeyer et al. "Implicit 3D orientation learning for 6D object detection from RGB images". In: *European Conference on Computer Vision*. 2018, pp. 712–729.

[212] Kai Sheng Tai, Richard Socher, and Christopher D Manning. "Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks". In: *Annual Meeting of the Association for Computational Linguistics*. 2015, pp. 1556–1566.

[213] Duyu Tang, Bing Qin, and Ting Liu. "Document modeling with gated recurrent neural network for sentiment classification". In: *Conference on Empirical Methods on Natural Language Processing*. 2015, pp. 1422–1432.

[214] Yonglong Tian, Dilip Krishnan, and Phillip Isola. "Contrastive multiview coding". In: *arXiv preprint arXiv:1906.05849* (2019).

[215] Josh Tobin et al. "Domain randomization for transferring deep neural networks from simulation to the real world". In: *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*. 2017, pp. 23–30.

[216] Andrew Top, Ghassan Hamarneh, and Rafeef Abugharbieh. "Active learning for interactive 3D image segmentation". In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. 2011, pp. 603–610.

[217] Antonio Torralba and Alexei A Efros. "Unbiased look at dataset bias". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2011, pp. 1521–1528.

[218] Toan Tran et al. "A bayesian data augmentation approach for learning deep models". In: *Advances in Neural Information Processing Systems*. 2017, pp. 2797–2806.

[219] Jonathan Tremblay et al. "Training Deep Networks With Synthetic Data: Bridging the Reality Gap by Domain Randomization". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2018, pp. 969–977.

[220] Yi-Hsuan Tsai et al. "Learning to adapt structured output space for semantic segmentation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 7472–7481.

[221] Eric Tzeng et al. "Adversarial discriminative domain adaptation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 7167–7176.

[222] Eric Tzeng et al. "Deep domain confusion: Maximizing for domain invariance". In: *arXiv preprint arXiv:1412.3474* (2014).

[223] Eric Tzeng et al. "Simultaneous deep transfer across domains and tasks". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2015, pp. 4068–4076.

[224] Manuel Veit and Antonio Capobianco. "Go'Then'Tag: A 3-D point cloud annotation technique". In: *IEEE Symposium on 3D User Interfaces*. 2014, pp. 193–194.

[225] Thirumalaisamy P Velavan and Christian G Meyer. "The COVID-19 epidemic". In: *Tropical medicine & international health* 25.3 (2020), p. 278.

[226] Naveen Venkat et al. "Your Classifier can Secretly Suffice Multi-Source Domain Adaptation". In: *arXiv preprint arXiv:2103.11169* (2021).

[227] Hemanth Venkateswara et al. "Deep hashing network for unsupervised domain adaptation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 5018–5027.

[228] Thuy-Trang Vu, Dinh Phung, and Gholamreza Haffari. "Effective Unsupervised Domain Adaptation with Adversarially Trained Language Models". In: *Conference on Empirical Methods in Natural Language Processing*. 2020.

[229] Li Wan et al. "Regularization of neural networks using dropconnect". In: *International Conference on Machine Learning*. 2013, pp. 1058–1066.

[230] Feng Wang et al. "Additive margin softmax for face verification". In: *IEEE Signal Processing Letters* 25.7 (2018), pp. 926–930.

[231] Hanchen Wang et al. "Unsupervised point cloud pre-training via occlusion completion". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2021, pp. 9782–9792.

[232] Hang Wang et al. "Learning to Combine: Knowledge Aggregation for Multi-Source Domain Adaptation". In: *European Conference on Computer Vision*. Springer. 2020, pp. 727–744.

[233] Jindong Wang et al. "Visual Domain Adaptation with Manifold Embedded Distribution Alignment". In: *ACM International Conference on Multimedia*. 2018, pp. 402–410.

[234] Junxiang Wang and Liang Zhao. "Multi-instance domain adaptation for vaccine adverse event detection". In: *International World Wide Web Conference*. 2018, pp. 97–106.

[235] Panqu Wang et al. "Understanding convolution for semantic segmentation". In: *IEEE Winter Conference on Applications of Computer Vision*. 2018, pp. 1451–1460.

[236] Qian Wang and Toby Breckon. "Unsupervised domain adaptation via structured prediction based selective pseudo-labeling". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 04. 2020, pp. 6243–6250.

[237] Yequan Wang et al. "Aspect-level sentiment analysis using as-capsules". In: *International World Wide Web Conference*. 2019, pp. 2033–2044.

[238] Yequan Wang et al. "Sentiment analysis by capsules". In: *International World Wide Web Conference*. 2018, pp. 1165–1174.

[239] Peter Welinder and Pietro Perona. "Online crowdsourcing: rating annotators and obtaining cost-effective labels". In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*. 2010, pp. 25–32.

[240] Yu-Shiang Wong, Hung-Kuo Chu, and Niloy J Mitra. "Smartannotator: An interactive tool for annotating RGBD indoor images". In: *arXiv preprint arXiv:1403.5718* (2014).

[241] Bichen Wu et al. "SqueezeSeg: Convolutional Neural Nets with Recurrent CRF for Real-Time Road-Object Segmentation from 3D LiDAR Point Cloud". In: *IEEE International Conference on Robotics and Automation*. 2018.

[242] Bichen Wu et al. "Squeezesegv2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a lidar point cloud". In: *IEEE International Conference on Robotics and Automation*. 2019, pp. 4376–4382.

[243] Junmin Wu, Xiangyu Yue, and Wei Li. "Integration of hardware and software designs for object grasping and transportation by a mobile robot with navigation guidance via a unique bearing-alignment mechanism". In: *IEEE/ASME Transactions on Mechatronics* 21.1 (2015), pp. 576–583.

[244] Man Wu et al. "Unsupervised Domain Adaptive Graph Convolutional Networks". In: *The Web Conference*. 2020, pp. 1457–1467.

[245] Zhirong Wu et al. "Unsupervised feature learning via non-parametric instance discrimination". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 3733–3742.

[246] Zuxuan Wu et al. "DCAN: Dual Channel-wise Alignment Networks for Unsupervised Scene Adaptation". In: *arXiv preprint arXiv:1804.05827* (2018), pp. 518–534.

[247] Dongbo Xi et al. "Domain Adaptation with Category Attention Network for Deep Sentiment Analysis". In: *The Web Conference*. 2020, pp. 3133–3139.

[248] Fei Xia et al. "Gibson env: real-world perception for embodied agents". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2018.

[249] Tete Xiao et al. "Region Similarity Representation Learning". In: *arXiv preprint arXiv:2103.12902* (2021).

[250] Shaoan Xie et al. "Learning semantic representations for unsupervised domain adaptation". In: *International Conference on Machine Learning*. 2018, pp. 5423–5432.

[251] Jiaolong Xu, Liang Xiao, and Antonio M López. "Self-Supervised Domain Adaptation for Computer Vision Tasks". In: *IEEE Access* 7 (2019), pp. 156694–156706.

[252] Ruijia Xu et al. "Deep cocktail network: Multi-source unsupervised domain adaptation with category shift". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 3964–3973.

[253] Zhijie Xu and Shiliang Sun. "Multi-source transfer learning with multi-view adaboost". In: *International Conference on Neural Information Processing*. 2012, pp. 332–339.

[254] Ashima Yadav and Dinesh Kumar Vishwakarma. "Sentiment analysis using deep learning architectures: a review". In: *Artificial Intelligence Review* 53.6 (2020), pp. 4335–4385.

[255] Jun Yang, Rong Yan, and Alexander G Hauptmann. "Cross-domain video concept detection using adaptive svms". In: *ACM International Conference on Multimedia*. 2007, pp. 188–197.

[256] Luyu Yang et al. "Curriculum Manager for Source Selection in Multi-Source Domain Adaptation". In: *European Conference on Computer Vision*. 2020, pp. 608–624.

[257] Jae Shin Yoon et al. "Self-supervised adaptation of high-fidelity face models for monocular performance tracking". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 4601–4609.

[258] Chaohui Yu et al. "Transfer learning with dynamic adversarial adaptation network". In: *IEEE International Conference on Data Mining*. 2019, pp. 778–786.

[259] Fisher Yu and Vladlen Koltun. "Multi-scale context aggregation by dilated convolutions". In: *International Conference on Learning Representations*. 2016.

[260] Fisher Yu et al. "BDD100K: A Diverse Driving Video Database with Scalable Annotation Tooling". In: *arXiv preprint arXiv:1805.04687* (2018).

[261] Jianfei Yu and Jing Jiang. "Learning sentence embeddings with auxiliary tasks for cross-domain sentiment classification". In: *Conference on Empirical Methods in Natural Language Processing*. 2016, pp. 236–246.

[262] Xiangyu Yue et al. "A LiDAR Point Cloud Generator: from a Virtual World to Autonomous Driving". In: *Proceedings on International Conference on Multimedia Retrieval*. 2018, pp. 458–464.

[263] Xiangyu Yue et al. "Domain randomization and pyramid consistency: Simulation-to-real generalization without accessing target domain data". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pp. 2100–2110.

[264] Xiangyu Yue et al. "Multi-source Few-shot Domain Adaptation". In: *arXiv preprint arXiv:2109.12391* (2021).

[265] Xiangyu Yue et al. "Prototypical Cross-domain Self-supervised Learning for Few-shot Unsupervised Domain Adaptation". In: *arXiv preprint arXiv:2103.16765* (2021).

[266] Lei Zhang, Shuai Wang, and Bing Liu. "Deep learning for sentiment analysis: A survey". In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 8.4 (2018), e1253.

[267] Richard Zhang, Phillip Isola, and Alexei A Efros. "Colorful image colorization". In: *European conference on computer vision*. Springer. 2016, pp. 649–666.

[268] Yang Zhang, Philip David, and Boqing Gong. "Curriculum Domain Adaptation for Semantic Segmentation of Urban Scenes". In: *IEEE International Conference on Computer Vision*. 2017, pp. 2039–2049.

[269] Yang Zhang et al. "A Curriculum Domain Adaptation Approach to the Semantic Segmentation of Urban Scenes". In: *IEEE transactions on pattern analysis and machine intelligence* (2019).

[270] Yiheng Zhang et al. "Fully convolutional adaptation networks for semantic segmentation". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 6810–6818.

[271] Zufan Zhang, Yang Zou, and Chenquan Gan. "Textual sentiment analysis via three different attention convolutional neural networks and cross-modality consistent regression". In: *Neurocomputing* 275 (2018), pp. 1407–1415.

[272] Han Zhao et al. "Adversarial multiple source domain adaptation". In: *Advances in Neural Information Processing Systems*. 2018, pp. 8568–8579.

[273] Hengshuang Zhao et al. "Pyramid scene parsing network". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 2881–2890.

[274] Sicheng Zhao et al. "A Review of Single-Source Deep Unsupervised Visual Domain Adaptation". In: *IEEE Transactions on Neural Networks and Learning Systems* (2020).

[275] Sicheng Zhao et al. "Curriculum cyclegan for textual sentiment domain adaptation with multiple sources". In: *Proceedings of the Web Conference 2021*. 2021, pp. 541–552.

[276] Sicheng Zhao et al. "CycleEmotionGAN: Emotional Semantic Consistency Preserved CycleGAN for Adapting Image Emotions". In: *AAAI Conference on Artificial Intelligence*. 2019, pp. 2620–2627.

[277] Sicheng Zhao et al. "EmotionGAN: unsupervised domain adaptation for learning discrete probability distributions of image emotions". In: *ACM International Conference on Multimedia*. 2018, pp. 1319–1327.

[278] Sicheng Zhao et al. "Multi-source Distilling Domain Adaptation". In: *AAAI Conference on Artificial Intelligence*. 2020, pp. 12975–12983.

[279] Sicheng Zhao et al. "Multi-source Domain Adaptation for Semantic Segmentation". In: *Advances in Neural Information Processing Systems*. 2019, pp. 7285–7298.

[280] Sicheng Zhao et al. "Multi-source Domain Adaptation in the Deep Learning Era: A Systematic Survey". In: *arXiv preprint arXiv:2002.12169* (2020).

[281] Sicheng Zhao et al. "Predicting personalized emotion perceptions of social images". In: *ACM International Conference on Multimedia*. 2016, pp. 1385–1394.

[282] Shuai Zheng et al. "Conditional random fields as recurrent neural networks". In: *IEEE International Conference on Computer Vision*. 2015, pp. 1529–1537.

[283] Zangwei Zheng et al. "Scene-aware Learning Network for Radar Object Detection". In: *Proceedings of the 2021 International Conference on Multimedia Retrieval*. 2021, pp. 573–579.

[284] Lihua Zhou et al. "Prototype-Based Multisource Domain Adaptation". In: *IEEE Transactions on Neural Networks and Learning Systems* (2021).

[285] Jun-Yan Zhu et al. "Unpaired image-to-image translation using cycle-consistent adversarial networks". In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2223–2232.

[286] Xinge Zhu et al. "Penalizing top performers: Conservative loss for semantic segmentation adaptation". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 568–583.

[287] Yongchun Zhu, Fuzhen Zhuang, and Deqing Wang. "Aligning domain-specific distribution and classifier for cross-domain classification from multiple sources". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33. 01. 2019, pp. 5989–5996.

[288] Yuke Zhu et al. "Target-driven visual navigation in indoor scenes using deep reinforcement learning". In: *Robotics and Automation (ICRA), 2017 IEEE International Conference on*. IEEE. 2017, pp. 3357–3364.

[289] Junbao Zhuo et al. "Deep unsupervised convolutional domain adaptation". In: *Proceedings of the 25th ACM international conference on Multimedia*. 2017, pp. 261–269.

[290] Yang Zou et al. "Unsupervised domain adaptation for semantic segmentation via class-balanced self-training". In: *European Conference on Computer Vision*. 2018, pp. 289–305.

Appendix A

Supporting Materials for Chapter 2

A.1 Proof of Equation (2.12)

As mentioned in Chapter 2, in order for the prototype-classifier learning paradigm to work well, the network is desired to have enough confident predictions for all classes to get robust $\hat{\mathbf{w}}_i^s$ and $\hat{\mathbf{w}}_i^t$. First, to promote the network to have diversified outputs, we propose to maximize the entropy of expected network prediction $\mathcal{H}(\mathbb{E}_{\mathbf{x}}[p(y|\mathbf{x};\theta)])$. Second, to get high-confident prediction for each sample, we perform entropy minimization on the network output. So the overall objective is:

$$\max \mathcal{H}(\mathbb{E}_{\mathbf{x}}[p(y|\mathbf{x};\theta)]) - \mathbb{E}_{\mathbf{x}}[\mathcal{H}(p(y|\mathbf{x};\theta))]. \tag{A.1}$$

Now we show that this objective equals maximizing the mutual information between input and output, i.e. $\mathcal{I}(y;\mathbf{x})$:

$$\mathcal{H}(\mathbb{E}_{\mathbf{x}}[p(y|\mathbf{x};\theta)]) - \mathbb{E}_{\mathbf{x}}[\mathcal{H}(p(y|\mathbf{x};\theta))] \tag{A.2}$$

$$= \mathbb{E}_{\mathbf{x}}\left[\sum_{i=1}^{L} p(y_i|\mathbf{x}) \log p(y_i|\mathbf{x})\right] - \sum_{i=1}^{L} \mathbb{E}_{\mathbf{x}}[p(y_i|\mathbf{x})] \log \mathbb{E}_{\mathbf{x}}[p(y_i|\mathbf{x})]] \tag{A.3}$$

$$= \mathbb{E}_{\mathbf{x}}\left[\sum_{i=1}^{L} p(y_i|\mathbf{x}) \log p(y_i|\mathbf{x})\right] - \mathbb{E}_{\mathbf{x}}\left[\sum_{i=1}^{L} p(y_i|\mathbf{x}) \log \mathbb{E}_{\mathbf{x}}[p(y_i|\mathbf{x})]\right] \tag{A.4}$$

$$= \mathbb{E}_{\mathbf{x}}\left[\sum_{i=1}^{L} p(y_i|\mathbf{x}) \log \frac{p(y_i|\mathbf{x})}{\mathbb{E}_{\mathbf{x}}[p(y_i|\mathbf{x})]}\right] \tag{A.5}$$

Table A.1: Dataset statistics and labeled source used

Dataset	Domain	# total image	# labeled images	# classes
Office [186]	Amazon (A)	2817	1-shot and 3-shots labeled source	31
	DSLR (D)	498		
	Webcam (W)	795		
Office-Home [227]	Art (Ar)	2427	3% and 6% labeled source	65
	Clipart (Cl)	4365		
	Product (Pr)	4439		
	Real (Rw)	4357		
VisDA [172]	Synthetic (Syn)	152K	0.1% and 1% labeled source	12
	Real (Rw)	55K		
DomainNet [171]	Clipart (C)	18703	1-shot and 3-shots labeled source	126
	Painting (P)	31502		
	Real (R)	70358		
	Sketch (S)	24582		

$$= \mathbb{E}_{\mathbf{x}} \left[\int p(y|\mathbf{x}) \log \frac{p(y|\mathbf{x})}{\mathbb{E}_{\mathbf{x}}[p(y|\mathbf{x})]} \, \mathrm{d}y \right] \tag{A.6}$$

$$= \int p(\mathbf{x}) \, \mathrm{d}\mathbf{x} \int p(y|\mathbf{x}) \log \frac{p(y|\mathbf{x})}{\int p(\mathbf{x}) p(y|\mathbf{x}) \, \mathrm{d}\mathbf{x}} \, \mathrm{d}y \tag{A.7}$$

$$= \int p(\mathbf{x}) \, \mathrm{d}\mathbf{x} \int p(y|\mathbf{x}) \log \frac{p(y|\mathbf{x})}{p(y)} \, \mathrm{d}y \tag{A.8}$$

$$= \iint p(y, \mathbf{x}) \log \frac{p(y, \mathbf{x})}{p(y) p(\mathbf{x})} \, \mathrm{d}y \, \mathrm{d}\mathbf{x} \quad = \quad \mathcal{I}(y; \mathbf{x}) \tag{A.9}$$

In addition, we estimate $\mathcal{H}(\mathbb{E}_{\mathbf{x}}[p(y|\mathbf{x};\theta)])$ with $\sum_{x \in \mathcal{D}} p(y|\mathbf{x};\theta) \log \hat{\mathbf{p}}_0$, where $\hat{\mathbf{p}}_0$ is a moving average of $p(y|\mathbf{x};\theta)$.

A.2 Additional Datasets Details

Overall statistics of the datasets and the number of labeled source examples used in our experiments can be found in Table A.1. For Office [186], Office-Home [227] and VisDA [172], we follow the same setting in [119], randomly sampling labeled images from the source domain and ensure that each class has at least one labeled example. For DomainNet [171], we use the same split files as [188] and further select 1-shot and 3-shots labeled samples in the training set for each class.

Table A.2: Accuracy of cross-domain weighted kNN with different SSL methods.

Method	D→A	Rw→Cl
ImageNet pre-train	62.5	40.6
ID [245]	70.3	51.9
CDS [119]	72.5	53.7
protoNCE [134]	72.3	49.3
$\mathcal{L}_{\text{InSelf}} + \mathcal{L}_{\text{CrossSelf}}$	**75.5**	**55.3**

A.3 Additional Implementation Details

We implemented our model in PyTorch [168]. We choose batch size of 64 for both source and target in self-supervised learning and batch size of 32 for the classification loss. The learning rate ratio between linear layer and convolution layer is set to $1:0.1$. We use SGD with weight decay rate $5e^{-4}$. For Office and Office-Home, we adaptively set temperature ϕ according to [134]. For VisDA and DomainNet, we fix ϕ to be 0.1 for more stable training. We set temperature τ to be 0.1 in all experiments. We choose hyper-parameters λ_{in} and $\lambda_{\text{cross}} \in \{1, 0.5\}$, and the weight $\in \{0.05, 0.01\}$. As for parameters m (momentum for memory bank update) and M (number of k-means in $\mathcal{L}_{\text{InSelf}}$), we set $m = 0.5$ and $M = 20$.

We use spherical k-means for clustering and set half of the number of clusters in k-means to be the number of the classes n_c, and the rest to be $2n_c$. We compute the weight for cosine classifier only using source images for the first 5 epochs and set t_w to be around half of the average number of images per class. New prototypes (i.e. centroids of clusters and weights of cosine classifier) are computed per epoch for both self-supervised learning and classification.

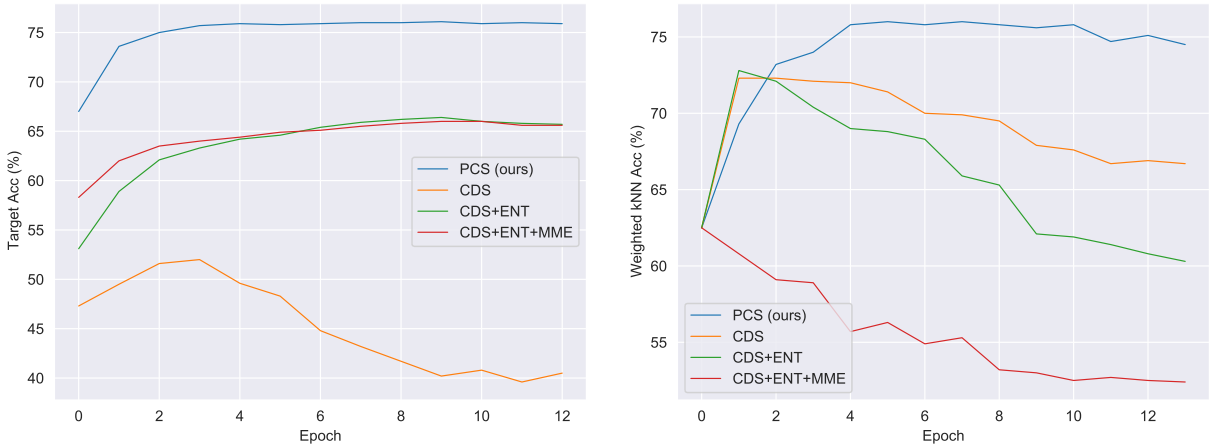
A.4 Quantitative Feature Analysis

To quantitatively compare the quality of learned features with different approaches, we perform classification with weighted k-nearest neighbor (kNN) classifier proposed by Wu *et al.* [245] in a cross-domain manner. Specifically, given a test image $\mathbf{x}^t$, we first compute its normalized feature $\mathbf{f}^t = F(\mathbf{x}^t)$, and then compare it again embeddings of all source images in the source memory bank $\boldsymbol{V^s}$ using cosine similarity $s_i = \cos(\mathbf{f}^t, \mathbf{v}_i^s)$. The top k nearest neighbors in the source domain, $\mathcal{N}_k$, would be used to make the final prediction with weighted voting. Specifically, class c would get weight $w_c = \sum_{i \in \mathcal{N}_k} \alpha_i \cdot 1(c_i = c)$, in which α_i is the contributing weight of neighbor $\mathbf{v}_i^s$ defined as $\alpha_i = \exp(s_i/\tau)$. We set $\tau = 0.07$ and $k = 200$ as in [245].

We perform the above cross-domain kNN classification on models trained with 1) only cross-domain self-supervised learning methods, and 2) Few-shot Unsupervised Domain Adaptation methods, with the results shown in Table A.2 and Table A.3, respectively. From the

Table A.3: Accuracy of cross-domain weighted kNN with different FUDA methods.

Method	D→A (1-shot)	Rw→Cl (3%)
CDS [119]	72.3	57.6
CDS + ENT	72.8	58.6
CDS + MME + ENT	60.8	59.2
PCS (Ours)	**76.0**	**59.3**



a Target accuracy with cosine classifier vs. training epochs

b Target accuracy with Weighted kNN vs. training epochs

Figure A.1: Stability of Target Accuracy during training procedure.

results, we can see that both the proposed cross-domain prototypical self-supervised learning method and the whole PCS framework outperforms previous approaches.

A.5 Stability Analysis of PCS

To show the performance stability of PCS, we conduct multiple runs with three different random seeds. Table A.4 reports the averaged accuracy and standard deviation of the three runs on the 1-shot and 3-shots settings of Office.

Figure A.1 shows adaptation accuracy vs. training epochs using cosine classifier (Figure A.1a) and weighted kNN classifier (Figure A.1b). From the plots, we have the following observations. (1) The target accuracy of PCS increases more steadily and robustly compared to other methods. In Figure A.1a, CDS starts decreasing at Epoch 3. In Figure A.1b, CDS and CDS+ENT starts decreasing at Epoch 1; while CDS+ENT+MME decreases from the

Table A.4: Averaged accuracy and standard deviation of PCS on three runs of 1-shot and 3-shots on Office dataset.

Labeled Source	A→D	A→W	D→A	D→W	W→A	W→D
1-shot	60.2±1.9	69.8±0.8	76.1±0.4	90.6±0.8	71.2±1.0	91.8±1.9
3-shots	78.2±1.8	82.9±1.1	76.4±0.5	94.1±0.1	76.3±0.7	96.0±0.7

Table A.5: Sum of pair-wise cosine-similarity between prototypes in Office and Office-Home.

Method	D→A (1-shot)	Rw→Pr (3%)
SO	0.44	-0.71
CDS [119]	0.43	-0.71
PCS w/o APCU	-53.3	-22.8
PCS (Ours)	**-58.4**	**-26.5**

beginning of training. In contrast, the performance of PCS increases smoothly until the end of training. (2) PCS converges much faster than other methods. We can see in Figure A.1a that PCS plateaus at around Epoch 3, while CDS+ENT and CDS+ENT+MME reaches the best performance at Epoch 9 and 10.

A.6 Prototype Quality Comparison

To further compare how well source and target are aligned, we provide more t-SNE [152] visualizations on Office (D→A) and Office-Home (Rw→Cl) in Figure A.2a and A.2b, comparing ImageNet Pre-training, CDS [119] and PCS. Specifically, we plot representations for all samples (top in both Figures), as well as the prototypes (normalized average representation) for each class. In top rows of both figures, the color of a sample represents its class, and samples from different domains are represented by different shapes (circles for source and crosses for target, best view after zooming in). In bottom rows of both figures, the number of a prototype represents its class index, and color represent the domain of the prototype (Cyan for source, Red for target, and Black for prototype weight of the classifier). As we can see from Figure A.2, for each class, the prototypes of source, target and the weight vector of classifier get more aggregated with PCS than other methods, which demonstrates that PCS could better align source and target representations for each category.

In a well-learned feature embedding space, prototypes of different classes should be far different from each other. To quantitatively measure the similarity of the learned prototypes, we compute the sum of cosine similarities between all pairs of prototypes. From the results shown in Table A.5, we can see that the prototypes learned with PCS have the least

Table A.6: Adaptation accuracy (%) comparison on fully-labeled setting on the Office-Home dataset.

Method	Ar→Cl	Ar→Pr	Ar→Rw	Cl→Ar	Cl→Pr	Cl→Rw	Pr→Ar	Pr→Cl	Pr→Rw	Rw→Ar	Rw→Cl	Rw→Pr	Avg
SO	34.9	50.0	58.0	37.4	41.9	46.2	38.5	31.2	60.4	53.9	41.2	59.9	46.1
DANN [69]	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
CDAN [145]	50.7	70.6	76.0	57.6	70.0	70.0	57.4	50.9	77.3	70.9	56.7	81.6	65.8
MMDIA [113]	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
MME [188]	54.2	72.8	78.3	57.9	70.2	71.8	58.5	52.9	77.9	72.7	58.1	81.8	67.3
CDS / MME [119]	56.9	73.3	76.5	62.8	73.1	71.1	63.0	57.9	79.4	72.5	62.5	83.0	69.3
PCS (Ours)	55.8	76.9	**80.3**	**67.9**	**74.0**	**75.7**	**67.0**	52.9	**81.0**	**74.5**	**58.3**	82.8	**70.6**

Table A.7: Adaptation accuracy (%) comparison on fully-labeled setting on the Office dataset.

Method	A→D	A→W	D→A	D→W	W→A	W→D	Avg
SO	68.9	68.4	62.5	96.7	60.7	99.3	76.1
DANN [69]	79.7	82	68.2	96.9	67.4	99.1	82.2
CDAN [145]	92.9	94.1	71	98.6	69.3	100	87.7
MMDIA [113]	92.1	90.3	75.3	98.7	74.9	99.8	88.8
MME [188]	88.8	87.3	69.2	98.7	65.6	100	84.9
CDS + MME [119]	86.9	88.3	75.9	98.6	73.3	100	87.1
PCS (Ours)	**94.6**	92.1	**77.4**	97.7	**77.0**	99.8	**89.8**

similarities, indicating that PCS learns an embedding space with better semantic structure.

A.7 Image Retrieval Results

We present cross-domain image retrieval results in Figure A.3. Given a query feature $\mathbf{f}_q$ in the target domain, we measure the pairwise cosine similarity between $\mathbf{f}_q$ and all features in the source domain. The source images with the most similar features as $\mathbf{f}_q$ are returned as the top retrieval results. We compare image retrieval results of PCS with CDS in Figure A.3. As shown in Figure A.3, features from model trained with CDS are biased to some wrong attributes, e.g. color, texture and other visual clues; and quantitatively similar features do not correspond to semantically similar images in different domains. In contrast, we can see that PCS could extract features that are more discriminative and semantically meaningful across domains.

Table A.8: Performance contribution of each part in PCS framework on Office-Home.

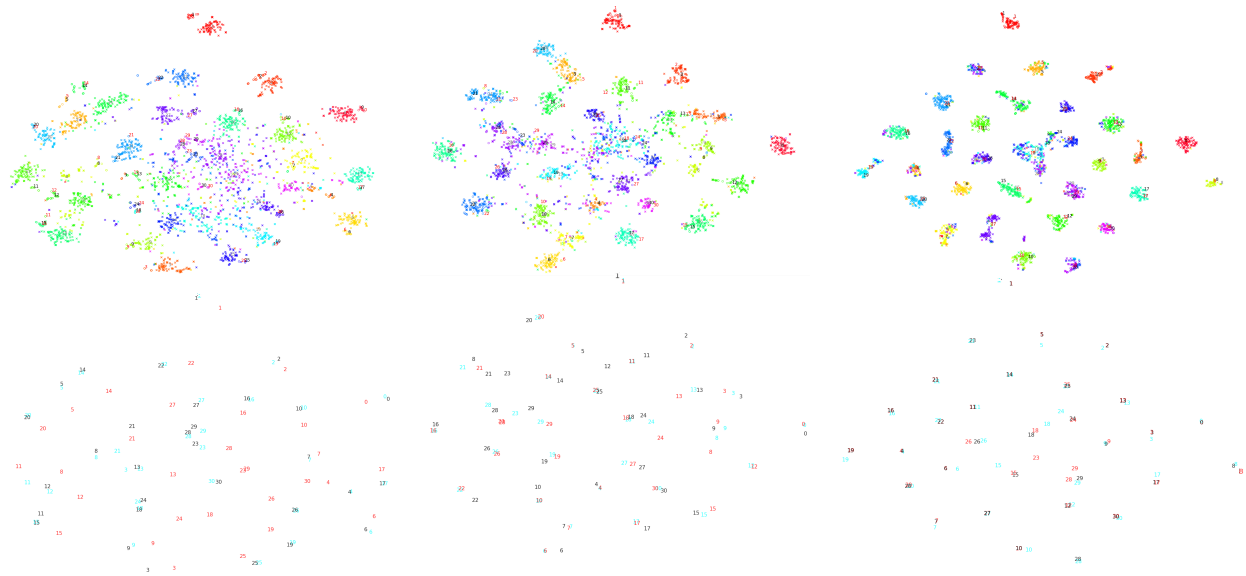
Method	Office-Home: Target Acc.												
	Ar →Cl	Ar →Pr	Ar →Rw	Cl →Ar	Cl →Pr	Cl →Rw	Pr →Ar	Pr →Cl	Pr →Rw	Rw →Ar	Rw →Cl	Rw →Pr	Avg
3% labeled source													
$\mathcal{L}_{\text{cls}}$	24.4	38.3	43.1	26.4	34.7	33.7	27.5	26.5	42.6	41.2	29.0	52.3	35.0
$+\mathcal{L}_{\text{InSelf}}$	34.6	48.3	54.7	49.2	53.1	57.1	48.2	40.6	62.9	57.9	44.9	68.8	51.7
$+\mathcal{L}_{\text{CrossSelf}}$	36.5	53.7	56.6	51.2	57.9	58.8	51.2	42.8	66.2	61.5	50.1	72.2	54.9
$+\mathcal{L}_{\text{MIM}}$	37.2	55.9	58.8	51.5	59.4	59.0	53.2	43.0	68.2	62.0	50.2	72.5	55.9
+APCU (PCS)	**42.1**	**61.5**	**63.9**	**52.3**	**61.5**	**61.4**	**58.0**	**47.6**	**73.9**	**66.0**	**52.5**	**75.6**	**59.7**
6% labeled source													
$\mathcal{L}_{\text{cls}}$	28.7	45.7	51.2	31.9	39.8	44.1	37.6	30.8	54.6	49.9	36.0	61.8	42.7
$+\mathcal{L}_{\text{InSelf}}$	40.8	57.6	65.5	54.5	62.4	62.7	54.6	43.1	73.6	64.2	44.7	75.9	58.3
$+\mathcal{L}_{\text{CrossSelf}}$	40.8	59.5	66.9	55.5	64.1	63.1	57.2	46.2	73.9	65.0	52.0	76.9	60.1
$+\mathcal{L}_{\text{MIM}}$	42.1	60.2	68.5	55.9	64.4	63.5	59.1	47.1	74.4	66.6	52.1	77.0	60.9
+APCU (PCS)	**46.1**	**65.7**	**69.2**	**57.1**	**64.7**	**66.2**	**61.4**	**47.9**	**75.2**	**67.0**	**53.9**	**76.6**	**62.6**

A.8 Performance Comparison with UDA Methods using Full Source Labels

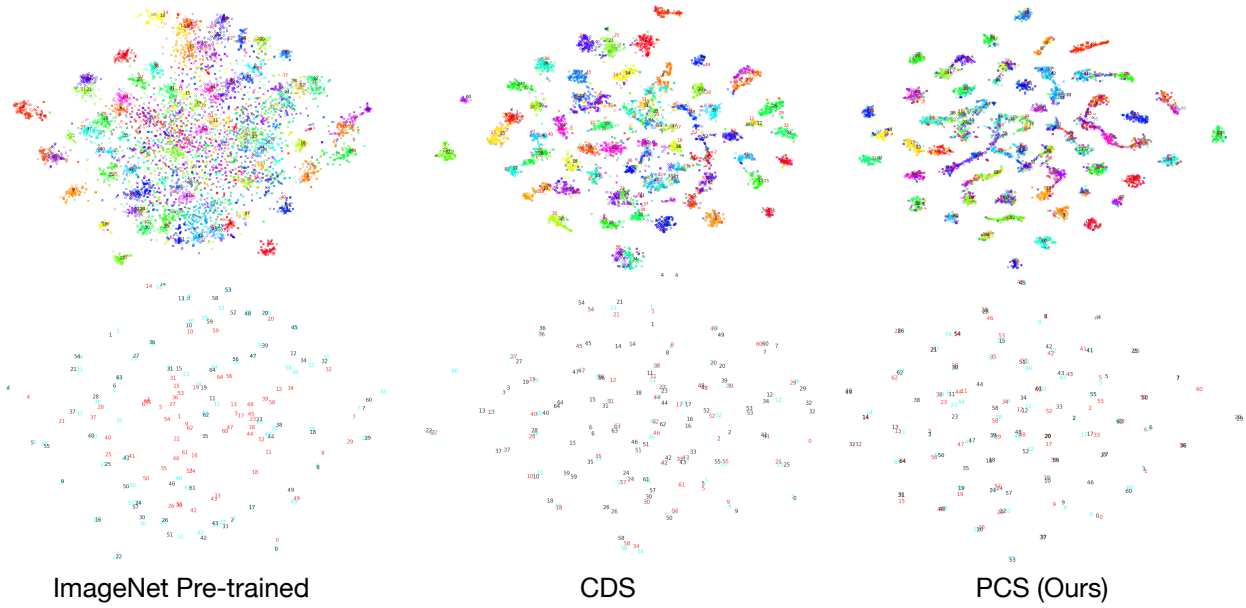
We have shown the superiority of PCS in label-scarce setting (FUDA), and we further conduct experiments with fully-labeled source domain (UDA). The performance comparison with other UDA methods on Office and Office-Home are presented in Table A.7 and Table A.6, respectively. We can see that PCS achieves the best results even with fully-labeled source, which demonstrates that the proposed PCS could potentially be applied to a wider range of domain adaptation settings.

A.9 More Ablation Study Results

In this section, we provide more ablation study results. Ablation experiments similar to Table 2 in the main paper are performed on Office-Home, with results shown in Table A.8. As we can see in the table, adding each component contributes to the final adaptation accuracy without any performance degradation, which demonstrates the effectiveness of all components in our PCS framework.



a Office (D→A with 1-shot labeled source per class)



b Office-Home (Rw→Cl with 3% labeled source per class)

Figure A.2: t-SNE visualization of ours and baselines on Office (a) and Office-Home (b). Top row: Coloring represents the class of each sample, and shape represents domain (circle for source and cross for target). Features with PCS are more discriminative than the ones with other methods. Bottom row: each number represents a centroid for corresponding class. **Cyan** represents centroids of source images based on ground truth and **Red** for target. **Black** represents prototypes of the classifier. Centroids from PCS are better-aligned between domains compared to other methods. (Zoom in for more details).

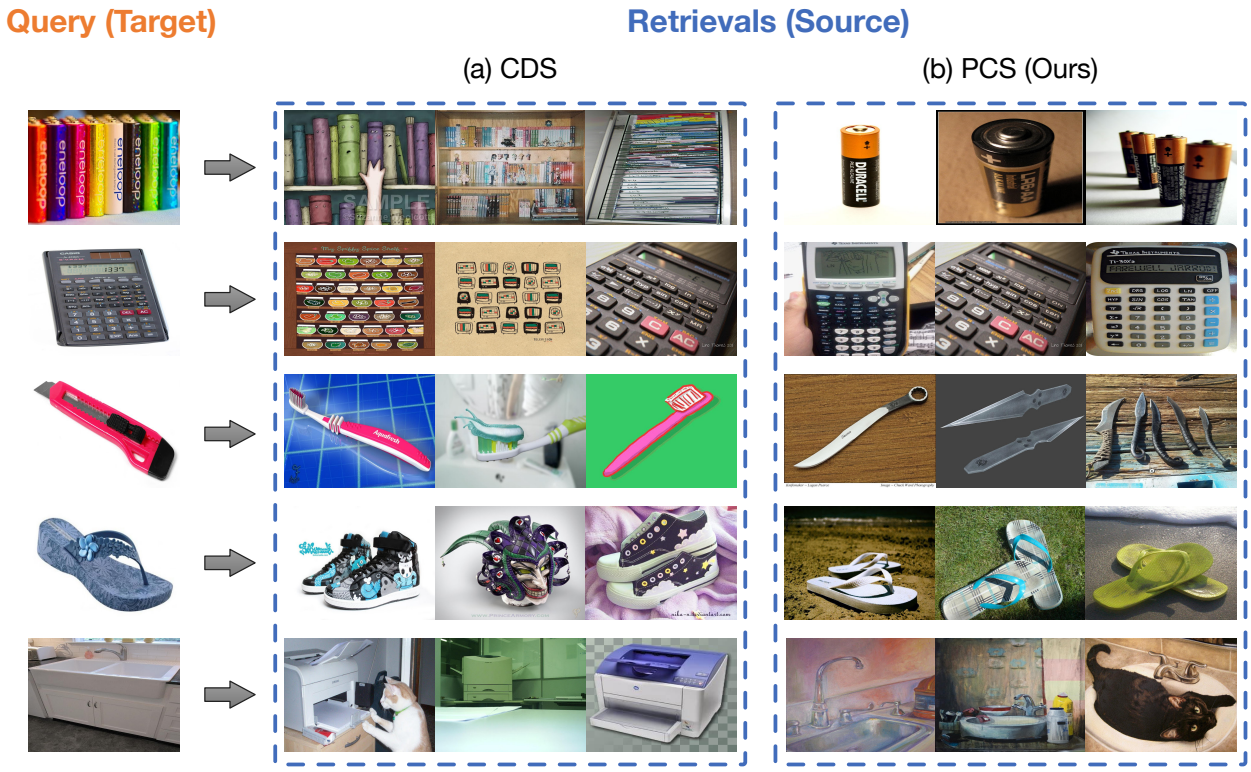


Figure A.3: Image retrieval examples of the closest cross-domain neighbors using CDS (a) and PCS (b) in Office-Home (Target: Real, Source: Art).

Appendix B

Supporting Materials for Chapter 3

B.1 Detailed Comparison with Other Works.

In Section 3.5 of Chapter 6, we provide comparison of the overall performance (mean IoU) of the models, specifically comparison with other domain generalization works from GTA to Cityscapes, and with other domain adaptation works from GTA to Cityscapes as well as from SYNTHIA to Cityscapes. Here, we provide more detailed comparison of the class-wise accuracies in Table B.2, Table B.3, and Table B.4. From the detailed tables, we can see that our method provides better performance in many classes and outperforms the state-of-the-art methods in terms of $mIoU$ under both domain generalization and domain adaptation, which shows the efficacy and superiority of our method.

B.2 Additional Experiments on auxiliary domains and color augmentation.

Two more experiments are conducted with FCN8s-VGG16 in this section. First, we re-run our approach with 15 real-world styles from the BDD dataset, including different weather conditions, time of day (TOD), etc. Then, we replace the style transfer step with 15 color augmentations [1], varying the hue, saturation, grayscale, contrast, etc. These changes preserve the semantics of the objects.

Table B.1 shows the new results (last two rows) along with those reported in Chapter 3. "Random" stands for the styles randomly selected from ImageNet and Artworks, and "Semantics" are the styles of the Cityscapes classes (e.g., Car, Road, etc.). The results are close to each other except that the color augmentation is a little worse than the others. The pyramid consistency is effective for all the test cases.

[1] `https://github.com/aleju/imgaug`

Table B.1: Adaptation from GTA with different style sets. We report results (mIoU%) both without / with the pyramid consistency.

Style Set	Semantics Safe?	Cityscapes	Mapillary
Random	✗	34.64 / 36.11	31.64 / 32.25
Semantics	✗	34.84 / 35.62	31.29 / 32.18
Weather-TOD	✗	34.51 / 35.89	31.24 / 32.18
Color Change	✓	33.56 / 34.52	30.27 / 32.06

Table B.2: Class-wise Performance comparison on Domain Generalization from GTA to Cityscapes with ResNet-50 base network.

Network	Method	Train w/ Tgt	Val on Tgt	road	sidewalk	building	wall	fence	pole	traffic light	traffic sign	vegetation	terrain	sky	person	rider	car	truck	bus	train	motorbike	bicycle	mIoU
ResNet-50	NonAdapt [165]	✗	✗	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	22.17
	IBN-Net [165]			–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	29.64
	NonAdapt	✗	✗	84.5	12.3	75.4	19.2	9.1	18.7	19.2	7.5	81.6	30.9	73.8	42.7	8.9	76.4	17.2	27.8	1.8	8.6	1.2	32.45
	Ours			90.1	21.6	79.4	25.6	18.2	22.6	26.4	16.5	82.9	34.3	77.1	46.1	13.5	78.3	24.4	29.1	3.6	13.4	7.8	**37.42**

B.3 More Discussion.

Table B.1 shows that the color augmentation performs a little worse than the style transfers probably for two reasons. One is that it does not bring to the synthetic images any appearances of the real images by design. The other is that it randomizes the images only by color (almost uniformly) and no texture. Learning an optimal non-uniform color shift policy is another future direction to explore.

Table B.1 shows that different style sets, including the real styles (i.e. weather) suggested by R3, lead to similar results. Together with Figure 3.4 in Chapter 3, we find that "how many domains" influences the results more than "what domains".

Table B.3: Class-wise Performance comparison from GTA to Cityscapes with VGG base network. All the best accuracies with respect to VGG-16 base network are in bold.

Network	Method	Train w/ Tgt	Val on Tgt	road	sidewalk	building	wall	fence	pole	traffic light	traffic sign	vegetation	terrain	sky	person	rider	car	truck	bus	train	motorbike	bicycle	mIoU
VGG19	NonAdapt [268]	✓	✓	18.1	6.8	64.1	7.3	8.7	21.0	14.9	16.8	45.9	2.4	64.4	41.6	17.5	55.3	8.4	5.0	6.9	4.3	13.8	22.3
	Curriculum [268]			74.9	22.0	71.7	6.0	11.9	8.4	16.3	11.1	75.7	13.3	66.5	38.0	9.3	55.2	18.8	18.9	0.0	16.8	16.6	28.9
	CGAN [102]	✓	✓	89.2	49.0	70.7	13.5	10.9	38.5	29.4	33.7	77.9	37.6	65.8	75.1	32.4	77.8	39.2	45.2	0.0	25.5	35.4	44.5
VGG16	NonAdapt [101]	✓	✓	31.9	18.9	47.7	7.4	3.1	16.0	10.4	1.0	76.5	13.0	58.9	36.0	1.0	67.1	9.5	3.7	0.0	0.0	0.0	21.1
	FCNs Wld [101]			70.4	32.4	62.1	14.9	5.4	10.9	14.2	2.7	79.2	21.3	64.6	44.1	4.2	70.4	8.0	7.3	0.0	3.5	0.0	27.1
	NonAdapt [190]	✓	✓	73.5	21.3	72.3	18.9	14.3	12.5	15.1	5.3	77.2	17.4	64.3	43.7	12.8	75.4	24.8	7.8	0.0	4.9	1.8	29.6
	LSD [190]			88.0	30.5	78.6	25.2	**23.5**	16.7	23.5	11.6	78.7	27.2	71.9	51.3	19.5	80.4	19.8	18.3	0.9	**20.8**	18.4	37.1
	NonAdapt [36]	✓	✓	29.8	16.0	56.6	9.2	17.3	13.5	13.6	9.8	74.9	6.7	54.3	41.9	2.9	45.0	3.3	13.1	1.3	6.8	0.0	21.9
	ROAD [36]			85.4	31.2	78.6	27.9	22.2	21.9	23.7	11.4	80.7	29.3	68.9	48.5	14.1	78.0	19.1	23.8	**9.4**	8.3	0.0	35.9
	NonAdapt [99]	✓	✓	26.0	14.9	65.1	5.5	12.9	8.9	6.0	2.5	70.0	2.9	47.0	24.5	0.0	40.0	12.1	1.5	0.0	0.0	0.0	17.9
	CyCADA [99]			85.2	37.2	76.5	21.8	15.0	23.8	22.9	**21.5**	80.5	31.3	60.7	50.5	9.0	76.9	17.1	28.2	4.5	9.8	0.0	35.4
	NonAdapt [187]	✓	✓	25.9	10.9	50.5	3.3	12.2	25.4	28.6	13	78.3	7.3	63.9	52.1	7.9	66.3	5.2	7.8	0.9	13.7	0.7	24.9
	MCD [187]			86.4	8.5	76.1	18.6	9.7	14.9	7.8	0.6	82.8	32.7	71.4	25.2	1.1	76.3	16.1	17.1	1.4	0.2	0.0	28.8
	I2I [159]	✓	✓	85.3	38.0	71.3	18.6	16	18.7	12	4.5	72	**43.4**	63.7	43.1	3.3	76.7	14.4	12.8	0.3	9.8	0.6	31.8
	NonAdapt [290]	✓	✓	64.0	22.1	68.6	13.3	8.7	19.9	15.5	5.9	74.9	13.4	37.0	37.7	10.3	48.2	6.1	1.2	1.8	10.8	2.9	24.3
	CBST-SP [290]			**90.4**	**50.8**	72.0	18.3	9.5	27.2	28.6	14.1	82.4	25.1	70.8	42.6	14.5	76.9	5.9	12.5	1.2	14.0	**28.6**	36.1
	NonAdapt [246]	✓	✓	72.5	25.1	71.2	6.6	13.4	12.3	11.0	4.7	76.1	16.4	67.7	43.1	8.0	70.4	11.3	4.8	0.0	13.9	0.4	27.8
	DCAN [246]			82.3	26.7	77.4	23.7	20.5	20.4	**30.3**	15.9	80.9	25.4	69.5	52.6	11.1	79.6	24.9	21.2	1.3	17.0	6.7	36.2
	NonAdapt [151]	✓	✓	26.0	14.9	65.1	5.5	12.9	8.9	6.0	2.5	70.0	2.9	47.0	24.5	0.0	40.0	12.1	1.5	0.0	0.0	0.0	17.9
	CLAN [151]			88.0	30.6	79.2	23.4	20.5	26.1	23.0	14.8	81.6	34.5	72.0	45.8	7.9	80.5	26.6	**29.9**	0.0	10.7	0.0	36.6
	BDL [137]	✓	✓	89.2	40.9	**81.2**	**29.1**	19.2	14.2	29.0	19.6	**83.7**	35.9	**80.7**	**54.7**	**23.3**	**82.7**	25.8	28.0	2.3	25.7	19.9	**41.3**
	NonAdapt [286]	✓	✓	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	30.0
	PTP [286]			–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	38.1
	AdaptSeg [220]	✓	✓	87.3	29.8	78.6	21.1	18.2	22.5	21.5	11.0	79.7	29.6	71.3	46.8	6.5	80.1	23.0	26.9	0.0	10.6	0.3	35.0
	NonAdapt [107]	✓	✓	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	18.8
	DAM [107]			–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	32.6
	NonAdapt	✗	✓	68.4	24.7	68.9	18.1	15.2	18.1	16.7	9.6	78.4	18.3	65.7	43.6	12.3	69.1	18.7	16.1	0.4	5.3	3.2	30.0
	Ours			86.6	38.4	79.8	26.4	18.1	**34.7**	21.3	16.3	81.2	28.7	76.5	50.1	16.6	80.7	**28.3**	21.4	2.3	14.3	10.9	38.6
	NonAdapt	✗	✗	66.4	23.9	69.1	16.3	15.8	19.6	15.8	8.6	77.7	19.5	66.1	43.2	12.1	68.9	17.3	17.2	0.3	4.8	2.9	29.8
	Ours			84.6	31.5	76.3	25.4	17.2	28.2	21.5	13.7	80.7	26.8	74.9	47.5	15.8	77.1	22.2	22.7	1.7	8.9	9.7	36.1

Table B.4: Class-wise Performance comparison from SYNTHIA to Cityscapes with VGG base network. All the best accuracies with respect to VGG-16 base network are in bold.

Network	Method	Train w/ Tgt	Val on Tgt	road	sidewalk	building	wall	fence	pole	traffic light	traffic sign	vegetation	sky	person	rider	car	bus	motorbike	bicycle	mIoU
VGG19	NonAdapt [268]	✓	✓	5.6	11.2	59.6	0.8	0.5	21.5	8.0	5.3	72.4	75.6	35.1	9.0	23.6	4.5	0.5	18.0	22.0
	Curriculum [268]			65.2	26.1	74.9	0.1	0.5	10.7	3.7	3.0	76.1	70.6	47.1	8.2	43.2	20.7	0.7	13.1	29.0
	CGAN [102]	✓	✓	85.0	25.8	73.5	3.4	3.0	31.5	19.5	21.3	67.4	69.4	68.5	25.0	76.5	41.6	17.9	29.5	41.2
VGG16	NonAdapt [101]	✓	✓	6.4	17.7	29.7	1.2	0.0	15.1	0.0	7.2	30.3	66.8	51.1	1.5	47.3	3.9	0.1	0.0	17.4
	FCNs Wld [101]			11.5	19.6	30.8	4.4	0.0	20.3	0.1	11.7	42.3	68.7	51.2	3.8	54.0	3.2	0.2	0.6	20.2
	NonAdapt [190]	✓	✓	30.1	17.5	70.2	5.9	0.1	16.7	9.1	12.6	74.5	76.3	43.9	13.2	35.7	14.3	3.7	5.6	26.8
	LSD [190]			**80.1**	29.1	77.5	2.8	0.4	26.8	11.1	18.0	78.1	76.7	48.2	15.2	70.5	17.4	8.7	16.7	36.1
	NonAdapt [36]	✓	✓	4.7	11.6	62.3	10.7	0.0	22.8	4.3	15.3	68.0	70.8	49.7	6.4	60.5	11.8	2.6	4.3	25.4
	ROAD [36]			77.7	30.0	77.5	9.6	0.3	25.8	10.3	15.6	77.6	79.8	44.5	16.6	67.8	14.5	7.0	23.8	36.2
	NonAdapt [290]	✓	✓	17.2	19.7	47.3	1.1	0.0	19.1	3.0	9.1	71.8	78.3	37.6	4.7	42.2	9.0	0.1	0.9	22.6
	CBST [290]			69.6	28.7	69.5	**12.1**	0.1	25.4	**11.9**	13.6	**82.0**	**81.9**	49.1	14.5	66.0	6.6	3.7	32.4	35.4
	NonAdapt [246]	✓	✓	10.8	11.4	66.6	1.6	0.1	16.9	5.5	14.1	74.2	76.2	46.0	11.5	45.4	15.1	6.0	13.4	25.9
	DCAN [246]			79.9	30.4	70.8	1.6	**0.6**	22.3	6.7	23.0	76.9	73.9	41.9	16.7	61.7	11.5	**10.3**	38.6	35.4
	BDL [137]	✓	✓	72.0	30.3	74.5	0.1	0.3	24.6	10.2	**25.2**	80.5	80.0	**54.7**	**23.2**	**72.7**	**24.0**	7.5	**44.9**	**39.0**
	DAM [107]	✓	✓	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	30.7
	NonAdapt [286]	✓	✓	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	24.9
	PTP [286]			–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	34.2
	NonAdapt	✗	✓	15.6	12.3	70.3	6.7	0.2	20.4	5.6	15.3	73.5	76.2	47.2	10.5	54.3	12.1	5.3	10.6	27.3
	Ours			78.9	**31.4**	**79.3**	9.6	0.2	**27.3**	10.1	15.6	76.2	78.5	45.1	16.4	69.8	13.6	8.3	22.7	36.4
	NonAdapt	✗	✗	14.7	11.8	68.5	7.3	0.1	19.6	4.6	14.4	71.8	73.2	48.5	9.1	56.1	11.7	4.9	11.7	26.8
	Ours			77.5	30.7	78.6	5.6	0.2	26.7	10.6	16.1	75.2	76.5	44.1	15.8	69.9	14.7	8.6	17.6	35.5

Appendix C

Supporting Materials for Chapter 4

C.1 Datasets

Cityscapes [41] contains vehicle-centric urban street images collected from a moving vehicle in 50 cities from Germany and neighboring countries. There are 5,000 images with pixel-wise annotations, including a training set with 2,975 images, a validation set with 500 images, and a test set with 1,595 images. The images have resolution of 2048×1024 and are labeled into 19 classes.

GTA [179] is a vehicle-egocentric image dataset collected in the high-fidelity rendered computer game GTA-V with pixel-wise semantic labels. It contains 24,966 images (video frames) with the resolution 1914×1052. There are 19 classes compatible with Cityscapes.

SYNTHIA [181] is a large synthetic dataset. To pair with Cityscapes, a subset, named SYNTHIA-RANDCITYSCAPES, is designed with 9,400 images with resolution 960×720 which are automatically annotated with 16 object classes, one void class, and some unnamed classes.

BDDS [260] contains thousands of real-world dashcam video frames with accurate pixel-wise annotations. It has a compatible label space with Cityscapes and the image resolution is 1280×720. There are 7,000, 1,000 and 2,000 images for training, validation, and testing, respectively.

C.2 Evaluation Metrics

Following [101, 268, 100], we employ class-wise intersection-over-union (cwIoU) and mean IoU (mIoU) to evaluate the segmentation results of each class and all classes. Let $\mathcal{P}_l$ and $\mathcal{G}_l$ respectively denote the predicted and ground-truth pixels that belong to class l, and then $cwIoU_l = \frac{|\mathcal{P}_l \cap \mathcal{G}_l|}{|\mathcal{P}_l \cup \mathcal{G}_l|}$, $mIoU = \frac{1}{L}\sum_{l=1}^{L} cwIoU_l$, where $|\cdot|$ denotes the cardinality of a set. Larger cwIoU and mIoU values represent better performances.

C.3 Implementation Details

Although the proposed MADAN could be trained in an end-to-end manner, due to the constrain of hardware resources, i.e. GPU memory, we train it in three stages. First, we train two CycleGANs [285] without semantic consistency losses for GTA $\leftrightarrow$ CityScapes and SYNTHIA $\leftrightarrow$ Cityscapes, and then train an FCN F on the adapted images with corresponding labels from the source domains. Second, after updating F_A with F trained above, we generate adapted images using CycleGAN with the proposed semantic consistency loss in Eq. (4) and aggregate different adapted domains using Cross-domain Cycle Discriminator and Sub-domain Aggregation Discriminator. Finally, we train an FCN on the newly adapted images in the aggregated domain with feature-level alignment. The above stages are trained iteratively. We leave the end-to-end training as future work by deploying model parallelism or experimenting with larger GPU memory.

In our experiments, we choose to use FCN [144] as our semantic segmentation network. To make it easier to compare with most of other methods, we use VGG-16 [202] as FCN backbone. The weights of the feature extraction layers in the networks are initialized from models trained on ImageNet [45]. The network is implemented in PyTorch and trained with Adam optimizer [121] using a batch size of 8 with initial learning rate 1e-4. Our machines are equipped with 4 NVIDIA Tesla P40 GPUs and 20 Intel(R) Xeon(R) CPU E5-2630 v4@2.20GHz.

All the images in SYNTHIA, GTA and Cityscapes are resized to 600×1080, and are then cropped to 400×400 during the training of the pixel-level adaptation for 20 epochs. Sub-domain aggregation discriminator and cross-domain cycle discriminator are frozen in the first 5 and 10 epochs, respectively. Please note that there are 16 and 19 different classes in SYNTHIA and GTA, respectively. In the experiments, we take the 16 intersection classes for all mIoU evaluations.

C.4 More Visualization on Semantic Segmentation Results

Here we visualize more segmentation results of single-source DA from GTA to Cityscapes in Figure C.1 and the proposed multi-source adaptation method, i.e. MADAN, in Figure C.2.

C.5 More Visualization on Pixel-level Alignment

Here, we show more visual pixel-level alignment results on both GTA and SYNTHIA to Cityscapes in Figure C.3.

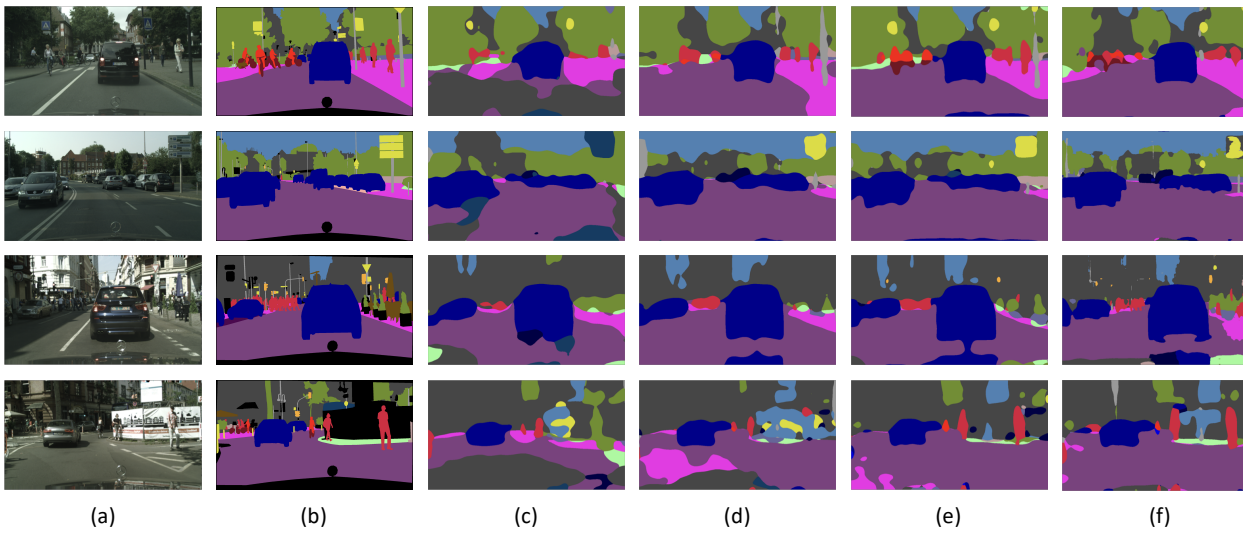


Figure C.1: Qualitative semantic segmentation result from GTA to Cityscapes. From left to right are: (a) original image, (b) ground truth annotation, (c) CycleGAN, (d) CycleGAN+DSC, (e) CycleGAN+DSC+Feat.

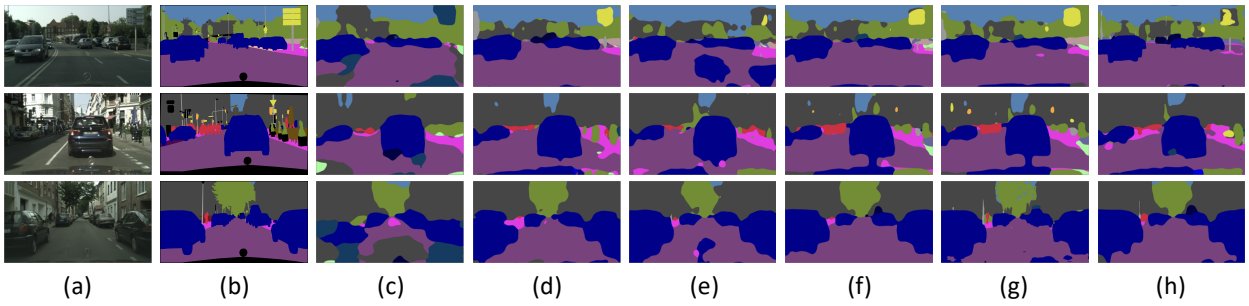


Figure C.2: Qualitative semantic segmentation result from GTA and SYNTHIA to Cityscapes. From left to right are: (a) original image, (b) ground truth annotation, (c) source only from GTA, (d) CycleGANs on GTA and SYNTHIA, (e) +CCD+DSC, (f) +SAD+DSC, (g) +CCD+SAD+DSC, and (h) +CCD+SAD+DSC+Feat (MADAN).

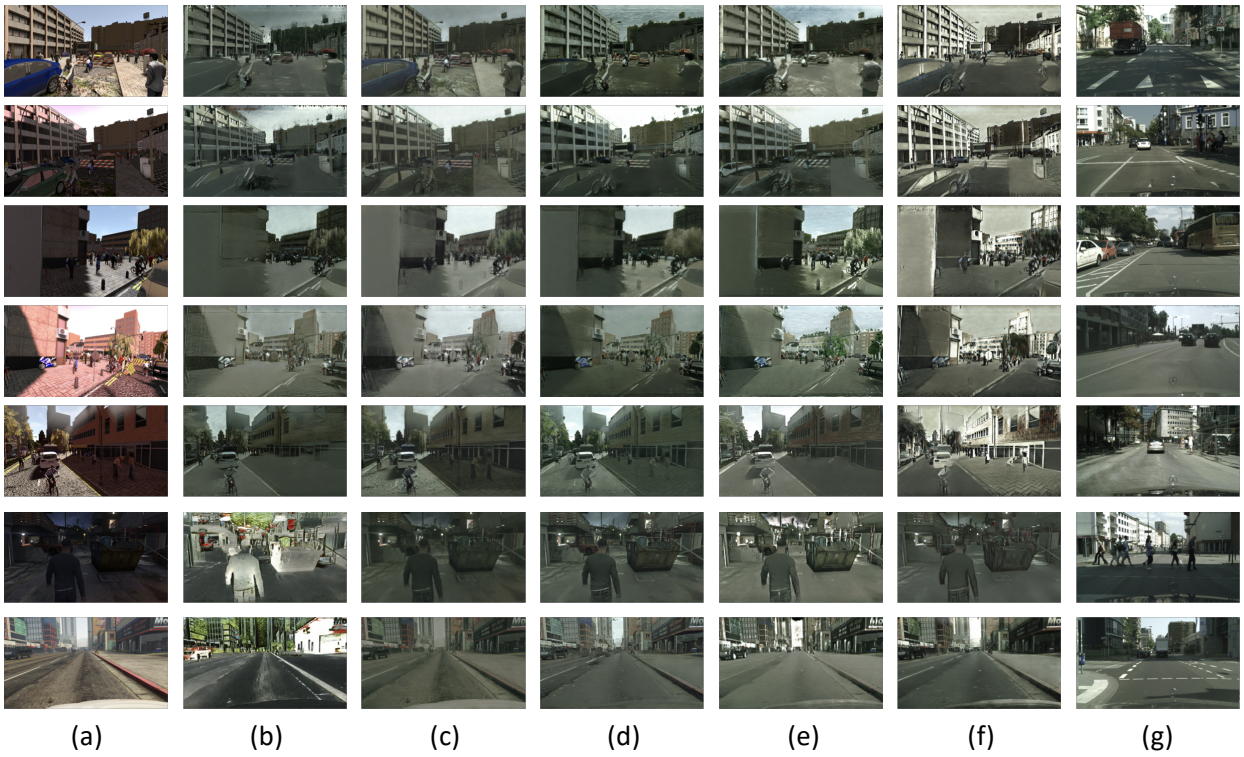


Figure C.3: Visualization of image translation. From left to right are: (a) original source image, (b) CycleGAN, (c) CycleGAN+DSC, (d) CycleGAN+CCD+DSC, (e) CycleGAN+SAD+DSC, (f) CycleGAN+CCD+SAD+DSC, and (g) target Cityscapes image.

Appendix D

Supporting Materials for Chapter 6

D.1 Additional Implementation Details

We implemented our model in PyTorch [168]. The training setting is adapted from [265]. For Office and Office-Home, the temperature ϕ is set adaptively according while for DomainNet, ϕ is fixed to 0.1 for more stable training. The margin m is always set to 0.1. We set temperature τ to be 0.1 in all experiments according to [265]. The weights for different loss are $\lambda_{\text{mps}} = 1, \lambda_{\text{ssc}} = 0.1, \lambda_{\text{mi}} = 0.1$.

We use a batch size of 64 and train our model on two NVIDIA P100 GPUs. The setting for clustering is same with [265] except that we found more frequent clustering yields better results on DomainNet and generate new prototypes every 200 iterations.

D.2 Stability Analysis of MSFAN

To show the performance stability of MSFAN, we conduct multiple runs with three different random seeds. Table D.1 reports the averaged accuracy and standard deviation on the 1-shot and 3-shot labels per class settings of Office. From the variance, we can see that the proposed MSFAN framework is experimentally stable.

Table D.1: Averaged accuracy (%) and standard deviation of three runs of 1-shot and 3-shots settings on the Office dataset.

1-shot			3-shot		
D,W→A	A,W→D	A,D→W	D,W→A	A,W→D	A,D→W
76.3 ± 0.32	94.4 ± 1.17	92.6 ± 0.08	77.7 ± 0.6	95.4 ± 0.06	95.8 ± 0.10

D.3 Multi-Source DA with Full Source Labels

We also apply MSFAN to the traditional MDA setting with full source labels. Table D.2 shows the performance comparison with state-of-the-art UDA and MDA methods on Office. We can see from the results that the proposed MSFAN framework still outperforms all previous methods with fully labeled source domains. This shows the potential wider application of MSFAN, not only in the label-scarce setting, but also in label-abundant setting. We hope to test the potential usage of MSFAN to other DA settings, such as multi-source semi-supervised DA, multi-source partial DA, multi-source open-set DA etc.

Table D.2: Adaptation accuracy (%) with full labels of source domains on Office dataset.

Office full-labeled					
	Method	D,W→A	A,W→D	A,D→W	Avg
Source Only	Single-best	62.5	99.3	96.7	86.2
	Combined	66.3	98.8	97.7	87.6
Single-best DA	CDAN [145]	71.0	100	98.6	89.9
	MME [188]	69.2	100	98.7	89.3
	MDDIA [113]	75.3	99.8	98.7	91.3
	CDS [119]	75.9	100	98.6	91.5
	PCS [265]	**77.4**	99.8	97.7	91.6
Source-combined DA	CDAN [145]	73.0	99.4	98.8	90.4
	MME [188]	69.2	98.7	99.2	89.0
	MDDIA [113]	76.1	99.4	98.2	91.2
	CDS [119]	73.9	98.8	98.4	90.4
	PCS [265]	76.3	98.3	97.1	90.6
Multi-source DA	SImpAl [226]	70.6	99.2	97.4	89.0
	MFSAN [287]	72.7	99.5	98.5	90.2
	PMDA [284]	75.2	99.7	98.3	91.1
	MSFAN (Ours)	77.0	**100**	**98.9**	**92.0**

D.4 Ablation Study on Cross-multi-domain Prototypical SSL Design ($\mathcal{L}_{\text{CPS}}$)

For $\mathcal{L}_{\text{CPS}}$ in "Cross-multi-domain Prototypical Self-supervised Learning" of Section 6.3 in Chapter 6, we propose to only minimize $H(P_j^{i\mapsto t})$, without considering $H(P_j^{t\mapsto i})$ or the similarity vector entropy between source domains.

In Table D.3, we compare the final performances of different designs of $\mathcal{L}_{\text{CPS}}$ to validate the proposed $\mathcal{L}_{\text{CPS}}$. All experiments are conducted with $\mathcal{L}_{\text{cls}} + \mathcal{L}_{\text{IPS}} + \mathcal{L}_{\text{CPS}}$, with the same

$\mathcal{L}_{\text{cls}}$ and $\mathcal{L}_{\text{IPS}}$, and different $\mathcal{L}_{\text{CPS}}$ for each row: i) "Every domain pair" refers to minimizing the entropy of instance-prototype similarity vectors between every domain pair of the $M+1$ domains (M source domains and 1 target domain). ii) "Tgt↔Src" refers to only minimizing the entropy of similarity vectors between each source domain and the target domain including both $H(P_j^{i\to t})$ and $H(P_j^{t\to i})$. iii) "Tgt→Src" means only considering the $H(P_j^{t\to i})$ between each source and the target. iv) "Src→Tgt" means only considering the $H(P_j^{i\to t})$ between each source domain and the target domain. From the results in Table D.3, we can see that the proposed design (only minimizing $H(P_j^{i\to t})$) achieves the best results.

Table D.3: Performance of different designs of $\mathcal{L}_{\text{CPS}}$ in Cross-multi-domain Prototypical SSL.

Office 1-shot	D,W→ A	A,W→ D	A,D→ W	Avg
Every domain pair	74.7	89.7	92.1	85.5
Tgt↔Src	75.4	89.7	91.7	85.6
Tgt→Src	72.0	85.3	91.1	82.8
Src→Tgt	**75.6**	89.7	**91.8**	**85.7**