UTS UNIVERSITY OF TECHNOLOGY SYDNEY

Bridging Theory and Algorithms for Open-Set and Heterogeneous Domain Adaptations

by *Zhen Fang*

Thesis submitted in fulfilment of the requirements for the degree of

Doctor of Philosophy in Computer Science

under the supervision of ***Professor Jie Lu***
and
Professor Guangquan Zhang

University of Technology Sydney
Faculty of Engineering and Information Technology
Australian Artificial Intelligence Institute

June, 2021

Faculty of Engineering and Information Technology
Australian Artificial Intelligence Institute
University of Technology Sydney

CERTIFICATE OF ORIGINAL AUTHORSHIP

I, Zhen Fang, declare that this thesis, is submitted in fulfilment of the requirements for the award of Doctor of Philosophy, in the Faculty of Engineering and Information Technology at the University of Technology Sydney. This thesis is wholly my own work unless otherwise referenced or acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis. This document has not been submitted for qualifications at any other academic institution. This research is supported by the Australian Government Research Training Program.

Signature: Zhen Fang

Production Note:
Signature removed prior to publication.

Date: 18 June. 2021

To my loving wife Yueting Peng, my parents and my son Junlin Fang.

ACKNOWLEDGMENTS

This thesis concludes a wonderful journey at University of Technology Sydney for pursuing my Ph.D. degree in the past three and half years. I would like to take the chance to express my sincere gratitude to the people who inspired and helped me in many ways.

First of all, I would like to express the most sincere gratitude to my principal supervisor Professor Jie Lu and co-supervisor Professor Guangquan Zhang, for their wisdom, vision, expertise, guidance, enthusiastic involvement and persistent encouragement during the planning and development of this research work. Their wide knowledge and logical way of thinking have been of great value for me. Their understanding, encouraging and intensive guidance make this dissertation become possible. I am very grateful for their consistent and generous support throughout my whole doctoral study.

Second, I am obliged to my parents for their moral support, love, encouragement and blessings to complete this task. I am especially thankful to my wife Yueting Peng for her patience, love and encouragement during this journey.

I wish to express my appreciation to Dr Feng Liu, Dr Anjin Liu, Zhaoqing Liu, Jiahua Dong, Dr Junyu Xuan, Dr Hua Zuo, Dr Feng Gu, Dr Yiliao Song, Dr Yi Zhang, Dr Hang Yu, Dr Qian Zhang, Bin Wang, Mengjia Wu and Tianyu Liu and grateful thanks to research fellows at department for their help and motivation throughout my research work. I would also like to extend my special thanks to Lily Qian, Camila Cremonese and other staff members of Australian Artificial Intelligence Institute (AAII) at University of Technology Sydney, for their timely help.

Zhen Fang

ABSTRACT

The availability of massive labeled datasets results in the successes of supervised learning in many applications such as object detection, speech recognition and natural language processing. However, the curse of domain mismatch arises if the test samples (target samples) and training samples (source samples) are from different domains. To overcome the mismatch between domains, researchers have proposed transfer learning, which aims to leverage knowledge from source samples with abundant labels to help train a classifier for target samples with insufficient labels.

Although many algorithms have been developed to solve transfer learning problem, most algorithms depend on additional assumptions to ensure their effectiveness. Many assumptions are difficult to be satisfied in real-world application. For example, the label sets of source and target samples are assumed to be same, however, in unsupervised setting, it is impossible to check this assumption. Those unrealistic assumptions limit the application of transfer learning.

This thesis mainly studies two important problems faced by many existing transfer learning algorithms: 1) how to transfer knowledge in the unsupervised setting, when the target label set is larger than the source label set; and 2) how to transfer knowledge, when the source and target samples are from different feature spaces.

Before solving problems 1) and 2), we first consider transfer learning which assumes that the feature spaces and label sets are both same. The setting is also called homogeneous domain adaptation. By studying the homogeneous domain adaptation, we aim to understand how the source knowledge can be used to help improve the target's performance under the simplest setting (i.e., homogeneous setting).

To solve problem 1), this thesis considers a challenging task: *unsupervised open-set domain adaptation* (UOSDA). First, to understand why source samples can be used to help target samples achieve good

performance in the open-set setting, we develop Probably approximately correct (PAC) learning theory for OSDA. Then, based on our theory, a kernel-based algorithm is proposed to solve problem 1). As an application of our OSDA theory, we also establish a theoretical foundation for *open-set learning* (OSL), an important sub-field in machine learning.

To solve problem 2), this thesis studies *semi-supervised heterogeneous domain adaptation* (SsHeDA) problem. Motivated by the compatibility condition in semi-supervised PAC theory, we explain the SsHeDA problem by proving its generalization error – why labeled source samples and unlabeled target samples help to reduce the target error. Guided by our theory, we devise a kernel-based algorithm.

List of Publications

Referred journals:

[1] **Zhen Fang**, Jie Lu, Feng Liu, Junyu Xuan, and Guangquan Zhang, Open Set Domain Adaptation Theoretical Bound and Algorithm, ***IEEE Transactions on Neural Networks and Learning Systems***, 2020. Published. (ERA **A***)

[2] Yiyang Zhang, Feng Liu, **Zhen Fang**, Bo Yuan, Guangquan Zhang and Jie Lu, Learning from a Complementary-label Source Domain: Theory and Algorithms, ***IEEE Transactions on Neural Networks and Learning Systems***, 2021. Accepted for publication. (ERA **A***)

International conferences:

[1] **Zhen Fang** and Jie Lu, Anjin Liu, Feng Liu, and Guangquan Zhang, Learning Bounds for Open-Set Learning, ***International Conference on Machine Learning*** (**ICML**), 2021. Accepted for publication. (CORA **A***)

[2] **Zhen Fang** and Jie Lu, Feng Liu, and Guangquan Zhang, Unsupervised Domain Adaptation with Sphere Retracting Transformation, ***International Joint Conference on Neural Networks*** (**IJCNN**), 2019. Published. (CORA **A**)

[3] Li Zhong, **Zhen Fang**, Feng Liu, Jie Lu, Bo Yuan, and Guangquan Zhang, How does the Combined Risk Affect the Performance of Unsupervised Domain Adaptation Approaches ?, ***Association for the Advancement of Artificial Intelligence*** (**AAAI**), 2020. Published. (CORA **A***)

[4] Yiyang Zhang, Feng Liu, **Zhen Fang**, Bo Yuan, Guangquan Zhang and Jie Lu, Clarinet: A One-step Approach Towards Budget-friendly Unsupervised Domain Adaptation, ***International Joint Conference on Artificial Intelligence*** (**IJCAI**), 2020. Published. (CORA **A***)

Papers submitted in referred journals:

[1] **Zhen Fang**, Jie Lu, Feng Liu and Guangquan Zhang, Semi-supervised Heterogeneous Domain Adaptation: Theory and Algorithms, ***IEEE Transactions on Pattern Analysis and Machine Intelligence***, 2021 (Revise and Resubmit). Submitted for publication. (ERA **A***)

[2] Feng Gu, Jie Lu, **Zhen Fang** and Guangquan Zhang, Neighbor-Searching Discrepancy-based Real Concept Drift, ***IEEE Transactions on Pattern Analysis and Machine Intelligence***, 2020 (Revise and Resubmit). Submitted for publication. (ERA **A***)

[3] Li Zhong, **Zhen Fang**, Feng Liu, Bo Yuan, Guangquan Zhang and Jie Lu, Bridging the Theoretical Bound and Deep Algorithms for Open Set Domain Adaptation, ***IEEE Transactions on Neural Networks and Learning Systems***, 2021 (Minor Revision). Submitted for publication. (ERA **A***)

Contents

List of Figures

List of Tables

List of Symbols

$\mathcal{X}$	Feature space
$\mathbf{x}$	Sample from $\mathcal{X}$
$\mathcal{Y}$	Label set
$\mathbf{y}$	Label (one-hot vector) from $\mathcal{Y}$
$\mathcal{X}_s$	Source feature space
$\mathbf{x}_s$	Source sample from $\mathcal{X}_s$
$\mathcal{S}$	Labeled source samples
$\mathcal{X}_t$	Target feature space
$\mathbf{x}_l$	Labeled target sample from $\mathcal{X}_t$
$\mathcal{T}_l$	Labeled target samples
$\mathbf{x}_u$	Unlabeled target sample from $\mathcal{X}_t$
$\mathcal{T}_u$	Unlabeled target samples
$\mathcal{Y}_s$	Source label set
$\mathbf{y}_s$	Label (one-hot vector) from $\mathcal{Y}_s$
$\mathcal{Y}_t$	Target label set
$\mathbf{y}_t$	Label (one-hot vector) from $\mathcal{Y}_t$
P_{XY}	Joint distribution
$P_{X_sY_s}$	Source joint distribution
$P_{X_tY_t}$	Target joint distribution
P_X	Marginal distribution
P_{X_s}	Source marginal distribution
P_{X_t}	Target marginal distribution
$P_{Y\|X}$	Conditional distribution
$P_{Y_s\|X_s}$	Source conditional distribution
$P_{Y_t\|X_t}$	Target conditional distribution

ℓ	Loss function
$\boldsymbol{h}$	Hypothesis function
$\mathcal{H}$	Hypothesis space
$k(\cdot,\cdot), k_s(\cdot,\cdot), k_t(\cdot,\cdot)$	Reproducing kernel (RK), source RK, target RK
$\mathcal{H}_k, \mathcal{H}_{k_s}, \mathcal{H}_{k_t}$	RKHS with kernels $k(\cdot,\cdot), k_s(\cdot,\cdot), k_t(\cdot,\cdot)$
$d^{\ell}_{\mathcal{H}}, d^{\ell}_{\boldsymbol{h},\mathcal{H}}$	$\mathcal{H}\Delta\mathcal{H}$-divergence, disparity discrepancy
$D_{\boldsymbol{h}}, D_{\mathcal{F}}$	Projected MMD distance, MMD distance
χ	Compatibility
$\mathrm{err}, \widehat{\mathrm{err}}$	Transfer error rate, empirical transfer error rate
Λ	Combined risk
$\boldsymbol{T}_s, \boldsymbol{T}_t$	Source and target transformations
$\mathcal{F}_s, \mathcal{F}_t$	Source and target transformation spaces
R^{α}_P, R^{α}_Q	α-risks corresponding to P_{XY}, Q_{XY}
$R_{P,k}$, $R_{Q,k}$	Partial risks for known classes for P_{XY}, Q_{XY}
$R_{P,u}$, $R_{Q,u}$	Partial risks for unknown classes for P_{XY}, Q_{XY}
R_s, R_t	Source risk, target risk risk
$\widehat{R}_s, \widehat{R}_t$	Empirical source risk, empirical target risk
R^*_t	Partial risk on known target classes
R^{K+1}_t	Partial risk on unknown target classes

Chapter 1

Introduction

1.1 Background

Supervised learning is based on two implied assumptions: 1) that the training and test samples are drawn from the same distribution [1, 2]; and 2) that sufficient labeled training samples are available [3]. However, in the real world, it is often expensive, time consuming, even impossible to collect sufficient labeled training samples while guaranteeing training and test samples are from a same distribution [4, 5].

As a way to relax or possibly even circumvent the above two assumptions, researchers have considered transfer learning. In transfer learning, we have two related but different domains: source and target domains, where the source domain contains sufficient labeled samples and the target domain only contains few labeled samples or unlabeled samples. The success of transfer learning as a technique has been proven many times over, with applications such as Wi-Fi localization [5], object classification [6–9], and text classification [10, 11].

Among most transfer learning tasks, researchers pay more attentions on unsupervised homogeneous domain adaptation (UHoDA) [12–16], which assumes that source and target domains share the same feature spaces and same label sets, but the target samples have not any labels. To show the feasibility of UHoDA, Ben-David et al. [17] provide the pioneering theoretical work [18]. In [17], the authors have shown that the target risk is upper bounded by three terms: source risk, distribution discrepancy, and combined risk. The theoretical work has provided a guidance to promote the development of UHoDA [19–22].

However, UHoDA is based on two implied assumptions that 1) the feature spaces of source and target domains are same; and 2) the source and target domains share same label sets. In reality, it is not easy to seek a source domain with the same feature space (i.e., homogeneous) as the target domain of interest [23–26]. In addition, in an unsupervised setting (i.e., there are no labels in the target domain), since it is not known whether the classes of target samples are from the label set of the source domain, it may be that the target domain contains additional classes (*unknown classes*) that do not exist in the label set of the source domain [27, 28].

To solve the mismatch between feature spaces, a more general and challenging problem *heterogeneous domain adaptation* (HeDA) [29, 30] has been considered, where the source and target domains are from different feature spaces. To solve the mismatch between label sets of source and target domains, [27, 28] propose the *open-set domain adaptation* [28, 31], which assumes the target label set contains the source label set.

1.2 Problem Statement

Given feature spaces $\mathcal{X}_s$, $\mathcal{X}_t$ and label sets $\mathcal{Y}_s$, $\mathcal{Y}_t$, source samples and target samples are related to two different joint distributions $P_{X_sY_s}$ and $P_{X_tY_t}$, where $X_s \in \mathcal{X}_s$, $X_t \in \mathcal{X}_t$, $Y_s \in \mathcal{Y}_s$ and $Y_t \in \mathcal{Y}_t$ are random variables. We use one-hot vector $\mathbf{y}$ to present a label, i.e., if the c-th coordinate of $\mathbf{y}$ is 1 and other coordinates are 0, then $\mathbf{y}$ means the label c. Then, we define UHoDA as follows:

Definition 1 (Homogeneous Domain Adaptation)**.** *Assume that feature spaces $\mathcal{X}_s$, $\mathcal{X}_t$ and label sets $\mathcal{Y}_s$, $\mathcal{Y}_t$ satisfy $\mathcal{X}_s = \mathcal{X}_t$, $\mathcal{Y}_s = \mathcal{Y}_t$. Given sets of samples called labeled source, labeled target, unlabeled target samples:*

$$\mathcal{S} = \{(\mathbf{x}_s^i, \mathbf{y}_s^i)\}_{i=1}^{n_s} \sim P_{X_sY_s} \ i.i.d.$$
$$\mathcal{T}_l = \{(\mathbf{x}_l^i, \mathbf{y}_l^i)\}_{i=1}^{n_l} \sim P_{X_tY_t} \ i.i.d.$$
$$\mathcal{T}_u = \{\mathbf{x}_u^i\}_{i=1}^{n_u} \sim P_{X_t} \ \ i.i.d.$$

The aim of ***homogeneous domain adaptation*** *is to train a classifier $g : \mathcal{X}_t \to \mathcal{Y}_t$ such that g classifies the unlabeled target samples accurately by using labeled and unlabeled samples, where $P_{X_s} \neq P_{X_t}$ and P_{X_s}, P_{X_t} are the marginal distributions related to joint distribution $P_{X_sY_s}$ and $P_{X_tY_t}$, respectively. If the size of labeled target samples n_l is 0, then homogeneous domain adaptation is called* ***unsupervised homogeneous domain adaptation*** *(UHoDA).*

Homogeneous domain adaptation, as one of the basic problems in transfer learning, assumes that the source and target feature spaces, and the source and target label sets are both same, i.e., $\mathcal{X}_s = \mathcal{X}_t$ and $\mathcal{Y}_s = \mathcal{Y}_t$. In this thesis, we focus on the unsupervised homogeneous domain adaptation. The unsupervised homogeneous domain adaptation is one of the simplest transfer learning problems. Due to $\mathcal{X}_s = \mathcal{X}_t$ and $\mathcal{Y}_s = \mathcal{Y}_t$, we need not to match the feature spaces and consider the mismatch between the label sets. By investigating the simplest setting (i.e., homogeneous), we aim to study the basic techniques in domain adaptation. With those basic techniques, we can understand how the source knowledge can be used to help improve the performance of target domain under the simplest setting (i.e., homogeneous).

• **RESEARCH QUESTION 1 (RQ1):** *How to the transfer source knowledge to the target domain under the homogeneous setting?*

• **RESEARCH OBJECTIVE 1 (RO1):** *To design novel UHoDA algorithms to address unsupervised homogeneous domain adaptation.* (aims to answer RQ1)

Next, we define open-set domain adaptation problem.

Definition 2 (Open-Set Domain Adaptation)**.** *Assume that feature spaces $\mathcal{X}_s$, $\mathcal{X}_t$ and label sets $\mathcal{Y}_s$, $\mathcal{Y}_t$ satisfy $\mathcal{X}_s = \mathcal{X}_t$, $\mathcal{Y}_s \subset \mathcal{Y}_t$. Given sets of samples called the labeled source and unlabeled target samples:*

$$\mathcal{S} = \{(\mathbf{x}_s^i, \mathbf{y}_s^i)\}_{i=1}^{n_s} \sim P_{X_sY_s} \ i.i.d.$$

$$\mathcal{T}_l = \{(\mathbf{x}_l^i, \mathbf{y}_l^i)\}_{i=1}^{n_l} \sim P_{X_tY_t} \ i.i.d.$$

$$\mathcal{T}_u = \{\mathbf{x}_u^i\}_{i=1}^{n_u} \sim P_{X_t} \ \ i.i.d.$$

The aim of ***open-set domain adaptation*** *is to train a classifier* $g : \mathcal{X}_t \to \mathcal{Y}_t$ *such that* g *classifies 1) target samples, whose labels are from* $\mathcal{Y}_s$*, into the correct classes; 2) target samples, whose labels are from* $\mathcal{Y}_t - \mathcal{Y}_s$*, as unknown, where* $P_{X_s} \neq P_{X_t}$ *and* P_{X_s}, P_{X_t} *are the marginal distributions related to joint distribution* $P_{X_s Y_s}$ *and* $P_{X_t Y_t}$*, respectively. If the size of labeled target samples* n_l *is* 0*, then homogeneous domain adaptation is called* ***unsupervised open-set domain adaptation*** *(UOSDA).*

Different from homogeneous domain adaptation, open-set domain adaptation replaces the assumption $\mathcal{Y}_s = \mathcal{Y}_t$ by $\mathcal{Y}_s \subset \mathcal{Y}_t$, which is also called the ***open-set assumption*** [32]. Hence, OSDA can be regarded as a generalization of homogeneous domain adaptation. In this thesis, we focus on unsupervised open-set domain adaptation. We propose the second research question as follows:

• **RESEARCH QUESTION 2 (RQ2):** *Is it feasible to transfer knowledge from a labeled source domain to a unlabeled target domain in the open-set assumption?*

• **RESEARCH OBJECTIVE 2 (RO2):** *To build the theory to study the feasibility of UOSDA and propose a new algorithm to address UOSDA.* (aims to answer RQ2)

For convenience, we call target samples, whose labels are from $\mathcal{Y}_t - \mathcal{Y}_s$, as unknown target samples, and target samples, whose labels are from $\mathcal{Y}_s$, as known target samples. There are two key challenges [28] in addressing the unsupervised open-set domain adaptation (UOSDA) problem. The first challenge is that there is no enough knowledge in the target domain to classify the unknown target samples. So how should these samples be labeled? The solution is to mine deeper information in the target domain to delineate a boundary between the known and unknown target samples. The second challenge in UOSDA is the difference in distributions. Unknown target samples should not be matched when the overall distribution is matched, otherwise negative transfer may occur.

Last, we define the heterogeneous domain adaptation problem.

Definition 3 (Heterogeneous Domain Adaptation)**.** *Assume that feature*

spaces $\mathcal{X}_s$, $\mathcal{X}_t$ and label sets $\mathcal{Y}_s$, $\mathcal{Y}_t$ satisfy $\mathcal{X}_s \neq \mathcal{X}_t$, $\mathcal{Y}_s = \mathcal{Y}_t$. Given sets of samples called the labeled source and unlabeled target samples:

$$\mathcal{S} = \{(\mathbf{x}_s^i, \mathbf{y}_s^i)\}_{i=1}^{n_s} \sim P_{X_s Y_s} \ i.i.d.$$
$$\mathcal{T}_l = \{(\mathbf{x}_l^i, \mathbf{y}_l^i)\}_{i=1}^{n_l} \sim P_{X_t Y_t} \ i.i.d.$$
$$\mathcal{T}_u = \{\mathbf{x}_u^i\}_{i=1}^{n_u} \sim P_{X_t} \ \ i.i.d.$$

The aim of heterogeneous domain adaptation is to train a classifier $g : \mathcal{X}_t \to \mathcal{Y}_t$ such that g classifies the unlabeled target samples accurately by using labeled and unlabeled samples. If $n_l \ll n_u$, $n_l \ll n_s$, then heterogeneous domain adaptation is called ***semi-supervised heterogeneous domain adaptation*** *(SsHeDA).*

Without any labels in target samples, heterogeneous domain adaptation is called **unsupervised heterogeneous domain adaptation** (UHeDA). To achieve good performance for UHeDA, however, only binary classification task [25] is investigated, since it is extremely difficult to bridge the connection between heterogeneous source and target domains without any labeled target samples [25]. Note that collecting a few labeled target samples is affordable. We consider semi-supervised heterogeneous domain adaptation in this thesis. Hence, we propose the third research question as follows:

• **RESEARCH QUESTION 3 (RQ3):** *How to transfer knowledge from a source domain to a heterogeneous target domain with few labels?*

To address the semi-supervised heterogeneous domain adaptation, we need to understand why labeled heterogeneous source samples and unlabeled target samples can help to reduce the need for labeled target samples. This is related to the learning theory for SsHeDA:

• **RESEARCH OBJECTIVE 3 (RO3):** *To build the theory to explain why labeled heterogeneous source samples and unlabeled target samples can help to reduce the need for labeled target samples and design an algorithm based theory to address SsHeDA.*

1.3 Research Innovation and Contributions

The research innovation and contributions of this thesis are summarized in the section.

1.3.1 Research Innovation

Innovation 1. Based on the unsupervised homogeneous domain adaptation, this research proposes a new setting, termed as source domain reconstruction. By constructing new source domain with labeled source samples and unlabeled target samples, a source domain reconstruction algorithm is developed to improve the performance of many existing UHoDA algorithms.

Innovation 2. This study is the first to build a theoretical foundation for open-set domain adaptation problem and propose a theoretical-guided unsupervised open-set domain adaptation algorithm. The theoretical work introduced in the thesis provides a theoretical guidance for the development of open-set domain adaptation [33, 34].

Innovation 3. As an important application of our open-set domain adaptation theory, this study is the first to provide a PAC-type generalization analysis for open-set learning (OSL) (i.e., supervised learning under the open-set assumption), which is an important sub-field of machine learning. Based on the theoretical work for OSL, a theoretical-guided OSL algorithm is proposed.

Innovation 4. To develop the heterogeneous domain adaptation theory, the research proposes several novel concepts, i.e., compatibility, transfer error rates and uniform sample complexity as new tools for estimating the need for labeled target samples. Co-opting these concepts provides a thoroughly new perspective for theoretically analyzing transfer learning problems.

Innovation 5. This study is the first to build a theoretical foundation for semi-supervised heterogeneous domain adaptation (SsHeDA) and explain why combining labeled source samples with unlabeled target samples can reduce the need for labeled target samples. Then, a theoretical-guided SsHeDA algorithm is proposed.

1.3.2 Research Contributions

Contribution 1. A new unsupervised homogeneous domain adaptation algorithm, called Manifold Fuzzy Clustering (MFC), is proposed.

• This study presents an effective unsupervised homogeneous domain adaptation algorithm, called MFC. MFC is able to transfer knowledge from labeled source samples to unlabeled target samples by exploring the intrinsic manifold structure and fuzzy information between domains.

Contribution 2. Source domain reconstruction (SDR), a new unsupervised homogeneous domain adaptation setting, is proposed.

• This study considers how to construct a new source domain by labeled source samples and unlabeled target samples. A novel source domain reconstruction (SDR) algorithm is proposed. After using the UHoDA algorithm, source and target samples as the inputs, our source domain reconstruction algorithm outputs a new source domain. With the new source domain, we improve the transfer performance of many existing UHoDA algorithms.

• To further study the feasibility of SDR, we develop a novel concept named as unsaturated algorithm space. Based on the novel concept, we prove an interesting and important theorem by Axiom of Choice, i.e., the necessary and sufficient condition for the existence of a SDR algorithm corresponding to an algorithm space is that the algorithm space is unsaturated. With this result, we establish a solid theoretical guarantee for SDR, which provides a positive answer for the feasibility of SDR.

Contribution 3. A novel algorithm, called Distribution Alignment with Open Difference (DAOD), is proposed to address unsupervised open-set domain adaptation.

• This study first provides a learning bound for open-set domain adaptation, which we do by theoretically investigating the risk of the target classifier on unknown classes (i.e., classes from $\mathcal{Y}_t - \mathcal{Y}_s$). The proposed learning bound has a special term, namely open-set difference, which reflects the risk of the target classifier on unknown classes.

• The study provides an UOSDA algorithm, Distribution Alignment with Open Difference (DAOD), which is based on regularizing a special term, called open-set difference. The algorithm enables the unknown target samples to be separated from samples using open-set difference.

Contribution 4. This study presents an algorithm, called Auxiliary Open-Set Risk (AOSR), to address open-set learning (OSL) problem (i.e., supervised learning under the open-set assumption).

• Motivated by the learning theory of open-set domain adaptation, this study makes a bold attempt to study OSL by proving its generalization error−given training samples with size n, there exists an algorithm, whose estimation error will get close to order $O_p(1/\sqrt{n})$. This is the first study to provide a generalization bound for OSL.

• According to our OSL theory, a novel algorithm, called Auxiliary Open-Set Risk (AOSR), is proposed to address the OSL problem. AOSR mainly utilizes the instance-weighting strategy to align training samples and auxiliary samples generated by an auxiliary domain. Then, minimizing the auxiliary risk developed by our theory, AOSR learns how to recognize unknown classes (i.e., unobserved classes during training process).

Contribution 5. This study presents a novel algorithms called Kernel Heterogeneous Domain Alignment (KHDA) to solve semi-supervised heterogeneous domain adaptation (SsHeDA).

• This study develops a theory of SsHeDA from a thoroughly new perspective compared to previous domain adaptation theories. Our bold strategy is to explain why the SsHeDA problem can be addressed by proving a novel generalization error of SsHeDA. By reducing the size of the target feature transformation space, the generalization error illustrates how the labeled source and unlabeled target samples, together with a suitable compatibility condition, can reduce the need for the labeled target samples.

• Guided by our theory, this study devises a novel SsHeDA algorithm to bring the proposed SsHeDA theory into the reality. Kernel Heterogeneous Domain Alignment (KHDA) is a kernel-based algorithm, which maintains two main branches, where the first branch aims to transfer knowledge from the source to the target domain, and the second branch aims to transfer knowledge from the labeled target samples to the unlabeled target samples.

1.4 Organization of the Thesis

The research work presented in the thesis is organized and structured in the form of seven chapters, which are briefly described as follows:

i) **Chapter 2** provides a comprehensive review of transfer learning and domain adaptation.

ii) **Chapter 3** presents a novel UHoDA algorithm Manifold Fuzzy Clustering (MFC). This chapter addresses RQ1 to achieve RO1.

iii) **Chapter 4** introduces a novel setting called source domain reconstruction. Theory about the feasibility of source domain reconstruction is established. An algorithm for source domain reconstruction is proposed. The chapter addresses RQ1 to achieve RO1.

iv) **Chapter 5** deals with a more challenging problem setting called unsupervised open-set domain adaptation (UOSDA), which replaces the closed-set assumption ($\mathcal{Y}_s = \mathcal{Y}_t$) in UHoDA by the open-set assumption ($\mathcal{Y}_s \subset \mathcal{Y}_t$). We show the theoretical work of OSDA and propose DAOD to address UOSD problem. This chapter addresses RQ2 to achieve RO2.

v) **Chapter 6** presents an important application for our OSDA theory. We use OSDA theory to address an open theoretical problem in open-set learning (OSL), which assumes test samples contains unseen classes during the training process. With the OSL theory, an algorithm AOSR is proposed to address OSL.

vi) **Chapter 7** proposes the semi-supervised heterogeneous domain adaptation (SsHeDA) theory, which aims to illustrate how the labeled source and unlabeled target samples, together with a suitable compatibility condition, can reduce the need for labeled target samples. Then, we design a theory-guided algorithm. This chapter addresses RQ3 to achieve RO3.

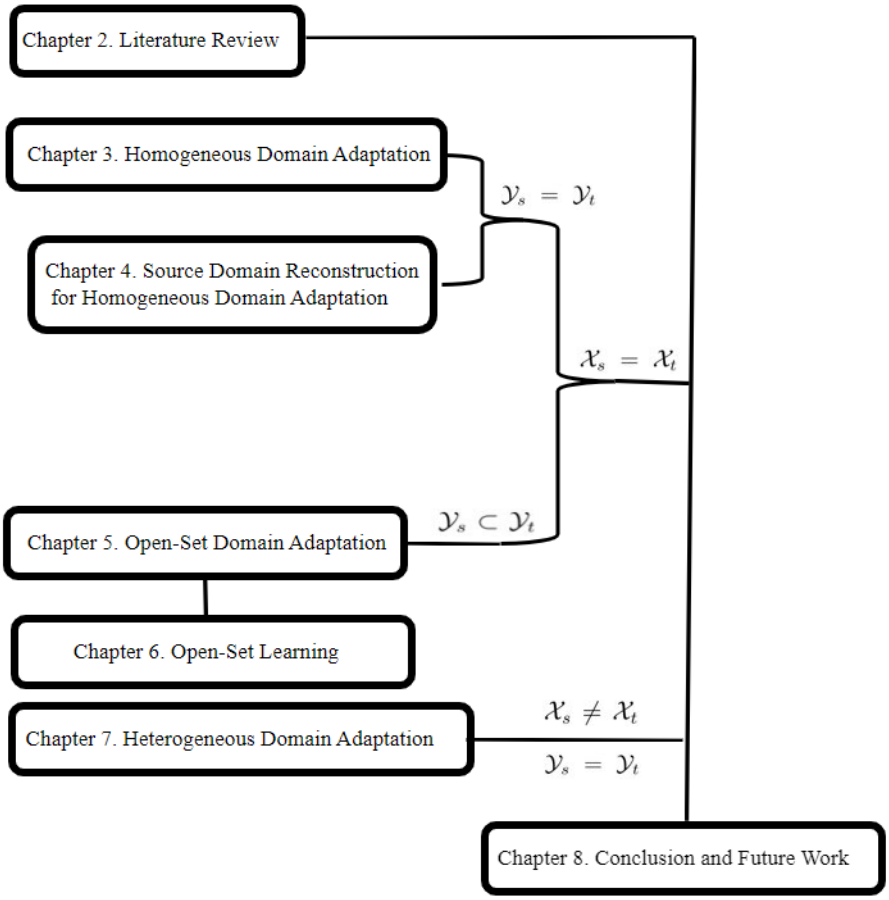


Figure 1.1: Thesis Structure. $\mathcal{X}_s$ and $\mathcal{X}_t$ are the feature spaces for source and target domains, respectively. $\mathcal{Y}_s$ and $\mathcal{Y}_t$ are the label sets for source and target domains, respectively.

vii) **Chapter 8** concludes the thesis with overall discoveries of the present research work and points to directions for future work.

The thesis structure is presented in Fig. 1.1.

Chapter 2

Literature Review

This chapter presents a survey related to this thesis, which includes two main parts: transfer learning and domain adaptation.

2.1 Transfer Learning

Transfer learning is used to improve the performance of the task from a domain (target domain) by transferring information from a related domain (source domain). Since the ability of transfer learning to leverage knowledge from related domains, transfer learning is widely used to address many problems, such as computer version [9,28,35], natural language processing [36,37], recommender system [38,39] and reinforcement learning [40, 41].

To formalize transfer learning, [42] gives a definition for transfer learning. Before giving the definition of transfer learning, we need to review the definitions of a domain and a task.

Definition 4 (Domain [42,43])**.** *A domain $\mathcal{D} = \{\mathcal{X}, P_X\}$ consists of two parts: a feature space $\mathcal{X}$ and a marginal distribution P_X, where $X \in \mathcal{X}$ is a random variable.*

Definition 5 (Task [42, 43])**.** *A task $\mathcal{T} = \{\mathcal{Y}, P_{Y|X}\}$ consists of two parts: a label set $\mathcal{Y}$ and a conditional distribution $P_{Y|X}$, where $Y \in \mathcal{Y}$ is a random variable and random variable X is introduced in Definition 4.*

In this thesis, we also call $(\mathcal{D}, \mathcal{T})$ as a domain.

Definition 6 (Transfer Learning [42,43])**.** *Let source domain and task be $(\mathcal{D}_s, \mathcal{T}_s)$, and target domain and task be $(\mathcal{D}_t, \mathcal{T}_t)$. Given sets of samples*

called the labeled source and unlabeled target samples:

$$\mathcal{S}_l = \{(\mathbf{x}_s^i, \mathbf{y}_s^i)\}_{i=1}^{n_s} \sim P_{X_s Y_s} \ i.i.d.$$
$$\mathcal{T}_l = \{(\mathbf{x}_l^i, \mathbf{y}_l^i)\}_{i=1}^{n_l} \sim P_{X_t Y_t} \ i.i.d.$$
$$\mathcal{S}_u = \{\mathbf{x}_{s,u}^i\}_{i=1}^{n_{s,u}} \sim P_{X_s} \ i.i.d.$$
$$\mathcal{T}_u = \{\mathbf{x}_u^i\}_{i=1}^{n_u} \sim P_{X_t} \ \ i.i.d.$$

where $n_s, n_l, n_{s,u}, n_u \geq 0$, $P_{X_s Y_s} = P_{X_s} \cdot P_{Y_s|X_s}$, *and* $P_{X_t Y_t} = P_{X_t} \cdot P_{Y_s|X_t}$, *the aim of transfer learning is to train a learner by given samples to solve target task* $\mathcal{T}_t$.

Remark 1. *By adjusting the sizes of samples (i.e.,* $n_s, n_l, n_{s,u}, n_u$*), different types of transfer learning can be obtained. For example, if* $n_l > 0$ *and* $n_s = 0$*, transfer learning becomes self-taught learning [42]. If* $n_s = 0$ *and* $n_l = 0$*, transfer learning becomes unsupervised transfer learning [43].*

Remark 2. *In Definition 6, we only consider single source domain. In fact, if we have multiple source domains, then transfer learning is termed as multiple source transfer learning [44].*

2.1.1 Categorization of Transfer Learning

Transfer learning can be categorized by different criterions.

Based on the availability of labeled samples, transfer learning can be roughly divided into three different categorizes [43]: inductive transfer learning (i.e., $n_l > 0$), transductive transfer learning (i.e., $n_s > 0, n_l = 0$) and unsupervised transfer learning (i.e., $n_s = n_l = 0$). According to the above criterion, unsupervised homogeneous domain adaptation and unsupervised open-set domain adaptation belong to transductive transfer learning, but semi-supervised heterogeneous domain adaptation belongs to inductive transfer learning.

Another criterion is based on the consistency between the source and target feature spaces and label sets [43]. If the feature spaces and label sets between source and target domains are same (i.e., $\mathcal{X}_s = \mathcal{X}_t$ and

$\mathcal{Y}_s = \mathcal{Y}_t$), then transfer learning is called homogeneous transfer learning. If the feature spaces or label sets between source and target domains are different (i.e., $\mathcal{X}_s \neq \mathcal{X}_t$ or $\mathcal{Y}_s \neq \mathcal{Y}_t$), then transfer learning is called heterogeneous transfer learning. It is clear that homogeneous domain adaptation belongs to homogeneous transfer learning, and open-set domain adaptation and heterogeneous domain adaptation belong to heterogeneous transfer learning.

The types of different target tasks can also be a criterion to category transfer learning. If the target task focuses on clustering task, transfer learning is called transfer learning for clustering [45, 46]. If the target task is classification task, transfer learning is known as transfer learning for classification [47, 48]. When we target on a regression task, transfer learning is regarded as transfer learning for regression [49,50]. This thesis focuses on transfer learning for classification.

2.1.2 Basic Strategies for Transfer Learning

Guided by transfer learning theory [17,51–53], many prior transfer learning algorithms consider to reduce the gap between domains to bridge source and target domains. To reduce the gap between domains, two basic strategies [43] have been investigated. One strategy is called ***instance weighting***, and the other is called ***feature matching***.

2.1.2.1 Instance Weighting

The basic idea of instance weighting is to construct weights to match source and target domains. As an effective strategy, the instance weighting strategy is often used in covariate shift [54,55], which is a sub-field of transductive transfer learning with assumption $\mathcal{T}_s = \mathcal{T}_t$, but $P_{X_s} \neq P_{X_t}$. The weighting strategy is based on the following equation [56]: given a loss function $\ell : \mathcal{Y}_t \times \mathcal{Y}_t \to \mathbb{R}^+$ and a predictor $\boldsymbol{h} : \mathcal{X}_t \to \mathcal{Y}_t$, if

$P_{Y_s|X_s} = P_{Y_t|X_t}$, then the target risk

$$\begin{aligned} R_t(\boldsymbol{h}) &= \int_{\mathcal{X}_t \times \mathcal{Y}_t} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}) \mathrm{d}P_{X_t Y_t}(\mathbf{x}, \mathbf{y}) \\ &= \int_{\mathcal{X}_t \times \mathcal{Y}_t} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}) \mathrm{d}P_{Y_t|X_t}(\mathbf{y}|\mathbf{x}) \mathrm{d}P_{X_t}(\mathbf{x}) \\ &= \int_{\mathcal{X}_t \times \mathcal{Y}_s} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}) \mathrm{d}P_{Y_s|X_s}(\mathbf{y}|\mathbf{x}) \mathrm{d}P_{X_t}(\mathbf{x}) \\ &= \int_{\mathcal{X}_s \times \mathcal{Y}_s} \frac{P_{X_t}(\mathbf{x})}{P_{X_s}(\mathbf{x})} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}) \mathrm{d}P_{Y_s|X_s}(\mathbf{y}|\mathbf{x}) \mathrm{d}P_{X_s}(\mathbf{x}) \\ &= \int_{\mathcal{X}_s \times \mathcal{Y}_s} \frac{P_{X_t}(\mathbf{x})}{P_{X_s}(\mathbf{x})} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}) \mathrm{d}P_{X_s Y_s}(\mathbf{x}, \mathbf{y}). \end{aligned}$$

Hence, from the empirical risk minimization perspective [57], the objective function can be written as

$$\min_{\boldsymbol{h}} \frac{1}{n_s} \sum_{i=1}^{n_s} w^i \ell(\boldsymbol{h}(\mathbf{x}_s^i), \mathbf{y}_s^i) + \lambda \Omega(\boldsymbol{h}), \tag{2.1.1}$$

where w^i is the estimated weight for the i-th sample $\mathbf{x}_s^i$ and Ω is the regularization function. According the equation above the Eq. (2.1.1), the ideal value of w^i should be $P_{X_t}(\mathbf{x}_s^i)/P_{X_s}(\mathbf{x}_s^i)$.

Based on the above discussion, the key of instance weighting strategy is to estimate the density ratio $P_{X_t}(\mathbf{x}_s^i)/P_{X_s}(\mathbf{x}_s^i)$. To learn an accurate density ratio, many algorithms have been proposed [56, 58–61]. One of the well-known algorithms is Kernel Mean Matching (KMM) [56], which computes the weights by Maximum Mean Discrepancy (MMD) [62]: given a Reproducing Kernel Hilbert Space $\mathcal{H}_k$ (RKHS) with kernel feature map Φ, then

$$\begin{gathered} \min_{w^i \in [0,B],\ i=1,\ldots,n_s} \Big\| \frac{1}{n_s} \sum_{i=1}^{n_s} w^i \Phi(\mathbf{x}_s^i) - \frac{1}{n_u} \sum_{j=1}^{n_u} \Phi(\mathbf{x}_u^j) \Big\|_{\mathcal{H}_k}^2, \\ \text{s.t.} \ \Big| \frac{1}{n_s} \sum_{i=1}^{n_s} w^i - 1 \Big| \leq \epsilon, \end{gathered}$$

where B is the upper bound for weights and ϵ is a small parameter to avoid over-fitting. After the weight w^i has been estimated, then we can train the predictor by weighted source samples.

KMM estimates the weights by MMD distance. Many other algorithms estimate the weights by using different distribution distance. For example, Kanamori et al. [63] used the least-square distance to propose least-squares importance fitting (LSIF). Kullback-Leibler Importance Estimation Procedure (KLIEP) proposed by Sugiyama et al. [64] utilizes the Kullback-Leibler (KL) divergence.

Though instance weighting-based transfer learning algorithms have achieve great development, many unrealistic assumptions implied on most algorithms limits the application of those algorithms. For example, in KMM, LSIF and KLIEP, to ensure a promising performance, the assumption that $\mathcal{T}_s = \mathcal{T}_t$ and $\mathcal{X}_s = \mathcal{X}_t$ is indispensable. Hence, when the domain gap is substantially large, a large number of effective source samples will be down-weighted, resulting in the loss of effective information.

2.1.2.2 Feature Matching

Feature matching aims to reduce the domain gap by learning a new feature representation. Generally, feature matching-based algorithms aim to 1) transform samples into a new space, where distance measures are minimized, or 2) extract intermediate features to minimize domain gap by projecting samples onto subspaces or transforme a source subspace into a new subspace, which is related to the target subspace. We introduce several classical feature matching-based algorithms as follows:

• Transfer Component Analysis (TCA) [5] learns a new feature space to match marginal distributions (P_{X_s} and P_{X_t}) by employing the Maximum Mean Discrepancy (MMD) [62].

• Joint Distribution Adaptation (JDA) [65] improves TCA by jointly matching marginal distributions (P_{X_s} and P_{X_t}) and class-conditional distributions ($P_{X_s|Y_s}$ and $P_{X_t|Y_t}$).

• Scatter Component Analysis (SCA) [21] extends TCA and JDA, and considers the between and within class scatter.

• Geodesic Flow Kernel (GFK) [66] considers the intermediate information in geodesic curve between the source domain and target domain on

a Grassmann manifold.

• Subspace Alignment (SA) [67] maps a source PCA subspace into a new subspace which is well-aligned with the target subspace.

• Correlation Alignment (CORAL) [68] matches the covariance matrix of the source subspace and target subspace.

Recent advances show that deep networks can be successfully applied to domain adaptation tasks.

• Domain Adaptive Neural Networks (DaNN) [69] adds an adaptation layer in neural networks to reduce the domain gap.

• Deep Adaptation Networks (DAN) [8] considers three adaptation layers for matching domains and applies multiple kernels (MK-MMD) [70] for adapting deep representations.

• Wasserstein Distance Guided Representation Learning (WDGRL) [20] minimizes the domain gap by employing Wasserstein distance in neural networks.

However, these algorithms have a strong assumption that there is a unified transformation to map the source domain and target domain into a common space in which the feature representations are domain invariant. Hence, the feature matching-based algorithms could fail to match domains when the domain gap is substantially large.

2.1.3 Deep Learning in Transfer Learning

Due to the universal approximation ability of deep neural networks, deep learning has been applied in transfer learning. Generally, transfer learning with deep neural networks as the basic model is called deep transfer learning [71]. Based on survey [71], deep transfer learning can be roughly divided into three categories: network-based, mapping-based and adversarial-based.

Network-based deep transfer learning [9, 72–78] aims to use labeled source samples to pre-train a deep model, then use the labeled target samples to fine-tune the pre-trained deep model.

• Yosinski et al. [76] conducted dozens of experiments on different transfer

learning tasks to show the transferability of feature extracted by different hidden layers of deep neural networks.

• Kolesnikov et al. [9] pre-trained a deep model based large size datasets (i.e., JFT-300M [79], ImageNet-21k [80] and ILSVRC-2012 [81]), then fine-tuned the pre-trained model on smaller size datasets (i.e., CIFAR-10 [82], CIFAR-100 [82] and Pets [83]).

Mapping-based deep transfer learning [19, 20, 47, 84] aims to learn domain-invariant representation, which can reduce the domain gap.

• Long et al. [47] proposed a representative algorithm Joint Adaptation Networks (JAN), which learns a domain-invariant representation by matching the joint distributions in multiple hidden layers based on Maximum Mean Discrepancy (MMD) [62].

Adversarial-based transfer learning [19, 22, 85, 86] refers to introduce generative adversarial nets (GAN) [87] to find transferable features.

• One of the representative works for adversarial-based transfer learning is Domain-Adversarial Neural Network (DANN) [22], which is based on feed-forward neural networks by augmenting it with a simple new gradient reversal layer.

• Long et al. [19] proposed a famous algorithm Conditional Domain Adversarial Networks (CDANs), which learns the domain-invariant representation by generative adversarial strategy with the cross-covariance between feature representations and predictor as the adversarial features.

Though deep transfer learning has achieved great successes, some assumptions are still indispensable to ensure the effective of deep transfer learning algorithms [71].

1) Network-based deep transfer learning assumes that the front-layers of the network can be treated as a feature extractor, and the extracted features are versatile.

2) Mapping-based deep transfer learning assumes that source and target domains can be more similarly in an latent feature space.

3) Adversarial-based transfer learning assumes that an effective feature representation should be discriminative for the main learning task

and indiscriminate between the source domain and target domain.

2.2 Domain Adaptation

Domain adaptation, as one of the most important fields in transfer learning, has achieved dramatic successes in many real-world applications [12, 88]. Traditional domain adaptation [42] assumes that $\mathcal{T}_s = \mathcal{T}_t$, but $\mathcal{D}_s \neq \mathcal{D}_t$. However, as the development of domain adaptation, the concept of domain adaptation has been extended in a broader sense. In this thesis, we say a transfer learning problem is called domain adaptation, if the target task is related to the source task ($\mathcal{T}_s \sim \mathcal{T}_t$), but $\mathcal{D}_s \neq \mathcal{D}_t$. For example, open-set domain adaptation introduced in Definition 2 assumes that source and target domains and tasks are different ($\mathcal{D}_s \neq \mathcal{D}_t, \mathcal{T}_s \neq \mathcal{T}_t$). To ensure that open-set domain adaptation is solvable, the source and target tasks should be similar in the shared part ($P_{Y_s|X_s} \sim P_{Y_t|X_t, Y_t \in \mathcal{Y}_s}$).

Similar to transfer learning, domain adaptation can be categorized based on different criterion. Based the availability of labeled target samples, domain adaptation can be divided into three types [12]: supervised domain adaptation (i.e., massive target labels are available) [89], semi-supervised domain adaptation (i.e., few target labels are available) [90] and unsupervised domain adaptation (i.e., no any target labels are available) [48]. Of these three paradigms, unsupervised domain adaptation is, arguably, the most attractive, since unsupervised domain adaptation is the most challenging problem among the three paradigms.

When the feature spaces are different ($\mathcal{X}_s \neq \mathcal{X}_t$), domain adaptation is regarded as heterogeneous domain adaptation [45]; otherwise, domain adaptation is called homogeneous domain adaptation [15]. In addition, if we focus on the difference between source and target label sets, domain adaptation can be separated into different types: open-set domain adaptation (i.e., $\mathcal{Y}_s \subset \mathcal{Y}_t$) [28], partial domain adaptation (i.e., $\mathcal{Y}_t \subset \mathcal{Y}_s$) [91] and universal domain adaptation (i.e., $\mathcal{Y}_t - \mathcal{Y}_s \neq \emptyset$ and $\mathcal{Y}_s - \mathcal{Y}_t \neq \emptyset$) [92].

2.2.1 Unsupervised Homogeneous Domain Adaptation

As one of main questions in this thesis, we first recall the definition (given in Definition 1) of unsupervised homogeneous domain adaptation as follows:

Assume that feature spaces $\mathcal{X}_s$, $\mathcal{X}_t$ and label sets $\mathcal{Y}_s$, $\mathcal{Y}_t$ satisfy $\mathcal{X}_s = \mathcal{X}_t$, $\mathcal{Y}_s = \mathcal{Y}_t$. Given sets of samples called the labeled source and unlabeled target samples:

$$\mathcal{S} = \{(\mathbf{x}_s^i, \mathbf{y}_s^i)\}_{i=1}^{n_s} \sim P_{X_s Y_s} \ i.i.d.$$
$$\mathcal{T}_u = \{\mathbf{x}_u^i\}_{i=1}^{n_u} \sim P_{X_t} \ \ i.i.d.$$

*where $P_{X_s} \neq P_{X_t}$. The aim of **unsupervised homogeneous domain adaptation** (UHoDA) is to train a classifier $g : \mathcal{X}_t \to \mathcal{Y}_t$ such that g classifies the unlabeled target samples accurately by using labeled and unlabeled samples.*

From above definition, we can see that: 1) $\mathcal{X}_s = \mathcal{X}_t$ and $\mathcal{Y}_s = \mathcal{Y}_t$, because our focus is on the homogeneous situation; 2) there are no any target labels, because we aim to solve the unsupervised situation. When Y_t is continuous random variable, we target on the regression task. If Y_t is discrete random variable, we target on the classification task. In this thesis, we focus on UHoDA for classification task.

It is clear that without any connection between the source and target tasks, it is difficult to achieve an effective transfer [93], because we have not any true labels in target samples. To mitigate this issue and establish the connection between source and target tasks, Ben-David et al. [17] proposed the β-close assumption:

Assumption 1 (β-close)**.** *The source and target tasks are β-close under a hypothesis space $\mathcal{H}$ and loss ℓ, if the combined error is smaller than β, i.e., $\min_{\boldsymbol{h}\in\mathcal{H}} \big(R_s(\boldsymbol{h}) + R_t(\boldsymbol{h})\big) < \beta$, where $R_s(\boldsymbol{h})$, $R_t(\boldsymbol{h})$ are the source and target risks with respect to predictor $\boldsymbol{h}$ and loss ℓ.*

To address UHoDA problem under Assumption 1, there are four

main techniques: instance-based algorithms [54–56, 58–61], distribution-distance-based algorithms [5, 8, 20, 65, 94], adversarial-based algorithms [19, 22, 28, 95] and pseudo-labeling-based algorithms [96–98]. In most practical applications, the above techniques are used in combination to achieve better performance. We will introduce distribution-distance-based algorithms, adversarial-based algorithms and pseudo-labeling-based algorithms in subsections 2.2.1.1, 2.2.1.2 and 2.2.1.3. The instance-based algorithms are similar to the instance weighting-based algorithms introduced in subsection 2.1.2, hence, we omit it in this section.

2.2.1.1 Distribution-Distance-based Algorithms

2.2.1.1.1 Basic Ideas

1) The original idea of distribution-distance-based algorithms is to reduce the domain gap by learning new features, i.e., given transformation space $\mathcal{F} \subset \{\mathbf{F} : \mathcal{X}_t \to \mathcal{X}\}$, where $\mathcal{X}$ is the latent feature space,

$$\min_{\mathbf{F}\in\mathcal{F}} d(\widehat{P}_{\mathbf{F}(X_s)}, \widehat{P}_{\mathbf{F}(X_t)}),$$

where d is the distribution distance, and $\widehat{P}_{\mathbf{F}(X_s)}, \widehat{P}_{\mathbf{F}(X_t)}$ are the empirical source and target marginal distributions with new features after the transformation $\mathbf{F}$.

• Transfer Component Analysis (TCA) [5], as one of the most representative algorithms, introduces the Maximum Mean Discrepancy (MMD) [62] to UHoDA, i.e., given matrix $\mathbf{W}$ and kernel feature map Φ related to RKHS $\mathcal{H}$,

$$\min_{\mathbf{W}} \left\| \frac{1}{n_s} \sum_{i=1}^{n_s} \mathbf{W}^T \Phi(\mathbf{x}_s^i) - \frac{1}{n_u} \sum_{j=1}^{n_u} \mathbf{W}^T \Phi(\mathbf{x}_u^j) \right\|^2.$$

By optimizing the above problem using eigenvalue decomposition algorithms, TCA ensures the discrepancy between source and target domains is closer than before.

Considering the gap between source and target tasks (i.e., $P_{Y_s|X_s}$ may be different with $P_{Y_t|X_t}$), researchers [65, 96] have extended the original

idea by considering the class-condition distributions,

$$\min_{\mathbf{F}\in\mathcal{F}} d(\widehat{P}_{\mathbf{F}(X_s)}, \widehat{P}_{\mathbf{F}(X_t)}) + \mu \sum_{c\in\mathcal{Y}_t} d(\widehat{P}_{\mathbf{F}(X_s)|Y_s=c}, \widehat{P}_{\mathbf{F}(X_t)|Y_t=c}),$$

where $\widehat{P}_{\mathbf{F}(X_s)|Y_s=c}, \widehat{P}_{\mathbf{F}(X_t)|Y_t=c}$ are the empirical source and target class-condition distributions with new features after the transformation $\mathbf{F}$. Since we have no true labels for target samples, the target class-condition distribution $\widehat{P}_{\mathbf{F}(X_t)|Y_t=c}$ is computed by pseudo-labels.

• Joint Distribution Adaptation (JDA) [65] is the first work to introduce the class-condition distribution alignment technique in UHoDA. Given matrix $\mathbf{W}$ and kernel feature map Φ related to RKHS $\mathcal{H}$,

$$\begin{aligned}\min_{\mathbf{W}} \Big\| \frac{1}{n_s} \sum_{i=1}^{n_s} \mathbf{W}^T \Phi(\mathbf{x}_s^i) - \frac{1}{n_u} \sum_{j=1}^{n_u} \mathbf{W}^T \Phi(\mathbf{x}_u^j) \Big\|^2 \\ + \mu \sum_{c\in\mathcal{Y}} \Big\| \frac{1}{|\mathcal{S}_c|} \sum_{\mathbf{x}\in\mathcal{S}_c} \mathbf{W}^T \Phi(\mathbf{x}) - \frac{1}{|\mathcal{T}_c|} \sum_{\mathbf{x}\in\mathcal{T}_c} \mathbf{W}^T \Phi(\mathbf{x}) \Big\|^2,\end{aligned}$$

where $\mathcal{S}_c$ is the set of source samples whose true labels are c and $\mathcal{T}_c$ is the set of target whose pseudo labels are c. By optimizing the above problem using eigenvalue decomposition algorithms, JDA can achieve better performance than TCA.

2.2.1.1.2 Distribution Distance

To estimate the discrepancy between distributions, several well-known distribution distances have been proposed.

• $\mathcal{H}\Delta\mathcal{H}$-divergence is first proposed by [53] for proving a learning bound for homogeneous domain adaptation.

Definition 7 ($\mathcal{H}\Delta\mathcal{H}$-divergence [53])**.** *Let the hypothesis space $\mathcal{H}$ be a set of functions defined in a feature space $\mathcal{X}$, ℓ be a loss function and P_1, P_2 be distributions on space $\mathcal{X}$ and $\boldsymbol{h}$ be any element in $\mathcal{H}$. The $\mathcal{H}\Delta\mathcal{H}$-divergence $d_{\mathcal{H}}^{\ell}(P_1, P_2)$ between the distributions P_1 and P_2 over $\mathcal{X}$ is*

$$\sup_{\boldsymbol{h},\boldsymbol{h}^*\in\mathcal{H}} \left| \mathop{\mathbb{E}}_{\mathbf{x}\sim P_1} \ell(\boldsymbol{h}(\mathbf{x}), \boldsymbol{h}^*(\mathbf{x})) - \mathop{\mathbb{E}}_{\mathbf{x}\sim P_2} \ell(\boldsymbol{h}(\mathbf{x}), \boldsymbol{h}^*(\mathbf{x})) \right|.$$

Few algorithms have used the $\mathcal{H}\Delta\mathcal{H}$-divergence as the distance to estimate the gap between distributions, due to the difficulty to estimate the $\mathcal{H}\Delta\mathcal{H}$-divergence by finite samples. To obtain tighter bounds for domain adaptation, Zhang et al. [99] revised the $\mathcal{H}\Delta\mathcal{H}$-divergence and proposed the disparity discrepancy:

Definition 8 (Disparity Discrepancy [99]). *Let the hypothesis space $\mathcal{H}$ be a set of functions defined in a feature space $\mathcal{X}$, ℓ be a loss function and P_1, P_2 be distributions on space $\mathcal{X}$ and $\boldsymbol{h}$ be any element in $\mathcal{H}$. The disparity discrepancy $d^{\ell}_{\boldsymbol{h},\mathcal{H}}(P_1, P_2)$ between the distributions P_1 and P_2 over $\mathcal{X}$ is*

$$\sup_{\boldsymbol{h}^* \in \mathcal{H}} \big(\mathop{\mathbb{E}}_{\mathbf{x} \sim P_1} \ell(\boldsymbol{h}(\mathbf{x}), \boldsymbol{h}^*(\mathbf{x})) - \mathop{\mathbb{E}}_{\mathbf{x} \sim P_2} \ell(\boldsymbol{h}(\mathbf{x}), \boldsymbol{h}^*(\mathbf{x})) \big).$$

It is obvious that the disparity discrepancy is tighter than the $\mathcal{H}\Delta\mathcal{H}$-divergence. Furthermore, based on the disparity discrepancy, Zhang et al. [99] proposed a novel algorithm Margin Disparity Discrepancy (MDD) to solve UHoDA.

• The other distance is Maximum Mean Discrepancy (MMD).

Definition 9 (MMD [62]). *Given a feature space $\mathcal{X}$ and a class of function $\mathcal{F} \subset \{f : \mathcal{X} \to \mathbb{R}\}$, the maximum mean discrepancy between distributions P_1 and P_2 is*

$$D_{\mathcal{F}}(P_1, P_2) = \sup_{f \in \mathcal{F}} \left| \mathop{\mathbb{E}}_{\mathbf{x} \sim P_1} f(\mathbf{x}) - \mathop{\mathbb{E}}_{\mathbf{x} \sim P_2} f(\mathbf{x}) \right|.$$

Gretton et al. [62] proposed the unit ball in a *reproducing kernel Hilbert space* (RKHS) $\mathcal{H}_k$ [21] (the subscript k represents the reproducing kernel) as the MMD function class $\mathcal{F}$. As we have introduced before, TCA [5] and JDA [65] both use the MMD distance to align distributions.

The projected MMD [21, 94, 100] has been proposed to transform the MMD distance into a proper regularization term. Given a scoring function $\boldsymbol{h} = [h_1, ..., h_{|\mathcal{Y}_t|}] \in \mathcal{H}$, if $h_c \in \mathcal{H}_k, c = 1, ..., |\mathcal{Y}_t|$, the projected

MMD is defined as follows:

$$D_{\boldsymbol{h}}(P_1, P_2) = \left\| \mathop{\mathbb{E}}_{\mathbf{x}\sim P_1} \boldsymbol{h}(\mathbf{x}) - \mathop{\mathbb{E}}_{\mathbf{x}\sim P_2} \boldsymbol{h}(\mathbf{x}) \right\|_2, \tag{2.2.1}$$

where $\|\cdot\|_2$ is the ℓ_2 norm.

Long et al. [6] proposed the Adaptation Regularization Transfer Learning (ARTL) by using the projected MMD:

$$\min_{\boldsymbol{h}\in\mathcal{H}} \frac{1}{n_s} \sum_{i=1}^{n_s} \ell(\boldsymbol{h}(\mathbf{x}_s^i), \mathbf{y}_s^i) + \mu\big(D_{\boldsymbol{h}}(\widehat{P}_{X_s}, \widehat{P}_{X_t}) + \sum_{c\in\mathcal{Y}_t} D_{\boldsymbol{h}}(\widehat{P}_{X_s|Y_s=c}, \widehat{P}_{X_t|Y_t=c})\big).$$

Different from general distribution-distance-based algorithms (e.g., TCA, JDA), ARTL regards the output space $\mathbb{R}^{|\mathcal{Y}_t|}$ as the latent feature space.

• Wasserstein distance is one of the most famous distribution distances. Given metric ρ over feature space $\mathcal{X}$, the p-Wasserstein distance between P_1 and P_2 is defined as follows:

$$W_p(P_1, P_2) = \inf_{\mu} \big(\int_{\mathcal{X}\times\mathcal{X}} \rho(\mathbf{x}_1, \mathbf{x}_2)^p \mathrm{d}\mu(\mathbf{x}_1, \mathbf{x}_2)\big)^{1/p},$$

where μ is the joint distribution over $\mathcal{X}\times\mathcal{X}$ with P_1 and P_2 as marginal distributions. When $p = 1$ and the metric is the Earth-Mover metric, then the p-Wasserstein distance can be rewritten as a form of integral probability metric [20]

$$W_1(P_1, P_2) = \sup_{\|f\|_L\leq 1} \big(\int_{\mathcal{X}} f(\mathbf{x}) \mathrm{d}P_1(\mathbf{x}) - \int_{\mathcal{X}} f(\mathbf{x}) \mathrm{d}P_2(\mathbf{x})\big),$$

where $\|f\|_L = \sup_{\mathbf{x}_1,\mathbf{x}_2\in\mathcal{X}} \big(f(\mathbf{x}_1) - f(\mathbf{x}_2)\big)/\rho(\mathbf{x}_1, \mathbf{x}_2)$.

Shen et al. [20] proposed the Wasserstein Distance Guided Representation Learning (WDGRL), which minimizes the domain gap by employing 1-Wasserstein distance in neural networks. Furthermore, Lee et al. [101] constructed the sliced 1-Wasserstein discrepancy for addressing UHoDA problem.

2.2.1.2 Adversarial-based Algorithms

Adversarial-based algorithms aim to utilize the generative adversarial nets (GAN) [87]. For the adversarial-based algorithms, the pioneer works are Domain Adversarial Neural Network (DANN) [22], Conditional Domain Adversial Networks (CDAN) [19] and Adversarial Discriminative Domain Adaptation (ADDA) [102].

• As we know, DANN [22] is the first UHoDA algorithm based on GAN. Given feature transformation $\mathbf{F}$ and domain classifier D without $0, 1$, then DANN utilies the GAN technique to distinguish source and target samples:

$$\max_{\mathbf{F}} \min_{D} \frac{1}{n_s} \sum_{i=1}^{n_s} \ell(D \circ \mathbf{F}(\mathbf{x}_s^i), 0) + \frac{1}{n_u} \sum_{i=1}^{n_u} \ell(D \circ \mathbf{F}(\mathbf{x}_u^i), 1).$$

In above minmax problem, the domain classifier aims to distinguish whether transformed samples $\mathbf{F}(\mathbf{x})$ comes from the source domain or from the target domain, and the feature transformation aims to match source and target domain for confusing the domain classifier D.

• CDAN [19] develops DANN by introducing the tensor technique in DANN. Given feature transformation $\mathbf{F}$, domain classifier D without $0, 1$ and source and target predictors $\boldsymbol{h}_s, \boldsymbol{h}_t$, then CDAN considers following minmax problem:

$$\max_{\mathbf{F}} \min_{D} \frac{1}{n_s} \sum_{i=1}^{n_s} \ell(D \circ (\mathbf{F} \otimes \boldsymbol{h}_s)(\mathbf{x}_s^i), 0) + \frac{1}{n_u} \sum_{i=1}^{n_u} \ell(D \circ (\mathbf{F} \otimes \boldsymbol{h}_t)(\mathbf{x}_u^i), 1),$$

where $\otimes$ means tensor. Optimizing above minmax problem, CDAN not only matches the features, but also matches the source and target predictors.

2.2.1.3 Pseudo-labeling-based Algorithms

The pseudo-labeling technique is originally proposed for semi-supervised learning and gains popularity in other learning problems. The pseudo-labeling techniques, as ancillary and effective techniques, have been used

widely in UHoDA [6, 65, 96–98]. The basic idea of pseudo-labeling algorithms is to utilize the predicted target labels to help train the target predictor in the next iteration. JDA, as we known, is the first work to propose the pseudo-labeling strategy in domain adaptation. Using the more accurate labels predicted in the last iteration, JDA can alternatingly improve the labeling quality until convergence. Saito et al. [97] proposed an asymmetric tri-training algorithm for UHoDA, which assigns pseudo-labels to unlabeled target samples and trains neural networks as if they are true labels. Note that CDAN [19] introduced in subsection 2.2.1.2 also utilizes the pseudo-labeling technique, because the target predictor $\boldsymbol{h}_t$ is used during training process.

2.2.2 Unsupervised Open-Set Domain Adaptation and Open-Set Learning

As one of the main problems in this thesis, we recall the definition (given in Definition 2) of unsupervised domain adaptation as follows: *Assume that feature spaces $\mathcal{X}_s$, $\mathcal{X}_t$ and label sets $\mathcal{Y}_s$, $\mathcal{Y}_t$ satisfy $\mathcal{X}_s = \mathcal{X}_t$, $\mathcal{Y}_s \subset \mathcal{Y}_t$. Given sets of samples called the labeled source and unlabeled target samples:*

$$\mathcal{S} = \{(\mathbf{x}_s^i, \mathbf{y}_s^i)\}_{i=1}^{n_s} \sim P_{X_s Y_s} \ i.i.d.$$
$$\mathcal{T}_u = \{\mathbf{x}_u^i\}_{i=1}^{n_u} \sim P_{X_t} \ \ i.i.d.$$

*where $P_{X_s} \neq P_{X_t}$. The aim of **unsupervised open-set domain adaptation** (UOSDA) is to train a classifier $g : \mathcal{X}_t \to \mathcal{Y}_t$ such that g classifies 1) target samples, whose labels are from $\mathcal{Y}_s$, into the correct classes; 2) target samples, whose labels are from $\mathcal{Y}_t - \mathcal{Y}_s$, as unknown.*

From above definition, we can see that: 1) $\mathcal{X}_s = \mathcal{X}_t$, because our focus is on the homogeneous feature spaces; 2) there are no any target labels, because we aim to solve the unsupervised situation. In addition, we assume that Y_s and Y_t are discrete random variables, since we target on the classification task.

Similar to UHoDA, without any connection between the source and target tasks, it is difficult to achieve an effective transfer [93], because we have not any true labels in target samples. To mitigate this issue and establish the connection between source and target tasks, Fang et al. [31] proposed the β-close assumption for UOSDA:

Assumption 2 (β-close)**.** *The source and target tasks are β-close under a hypothesis space $\mathcal{H} \subset \{\boldsymbol{h} : \mathcal{X} \to \mathcal{Y}_t\}$ and loss function ℓ, if the open-set combined error is smaller than β, i.e.,*

$$\min_{\boldsymbol{h} \in \mathcal{H}} \big(R_s(\boldsymbol{h}) + R_t^*(\boldsymbol{h})\big) < \beta,$$

where $R_s(\boldsymbol{h})$ is the source risk with respect to predictor $\boldsymbol{h}$ and $R_t^*(\boldsymbol{h})$ is the target risk for known classes, i.e.,

$$R_t^*(\boldsymbol{h}) = \int_{\mathcal{X}_t \times \mathcal{Y}_s} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}) \mathrm{d} P_{X_t Y_t | Y_t \in \mathcal{Y}_s}(\mathbf{x}, \mathbf{y}).$$

Unsupervised open-set domain adaptation is deeply related to open-set learning. In fact, open-set domain adaptation can be regarded as a combination between open-set learning and unsupervised homogeneous domain adaptation.

Definition 10 (Open-Set Learning)**.** *Assume that feature spaces $\mathcal{X}_s$, $\mathcal{X}_t$ and label sets $\mathcal{Y}_s$, $\mathcal{Y}_t$ satisfy $\mathcal{X}_s = \mathcal{X}_t$, $\mathcal{Y}_s \subset \mathcal{Y}_t$. Given sets of samples called the labeled source samples (training samples):*

$$\mathcal{S} = \{(\mathbf{x}_s^i, \mathbf{y}_s^i)\}_{i=1}^{n_s} \sim P_{X_s Y_s} \ i.i.d.$$

Suppose that the joint distributions $P_{X_s Y_s}$ and $P_{X_t Y_t}$ satisfy that

$$P_{X_s Y_s} = P_{X_t Y_t | Y_t \in \mathcal{Y}_s}$$

*The aim of **open-set learning** (OSL) is to train a classifier g over $\mathcal{X}_t$ such that g classifies 1) target samples (test samples), whose labels are from $\mathcal{Y}_s$, into the correct classes; 2) target samples (test samples), whose*

labels are from $\mathcal{Y}_t - \mathcal{Y}_s$, as unknown.

From the definition of OSL, we can discover that OSL also can be regard as a task in transfer learning, because in open-set learning, $\mathcal{D}_s \neq \mathcal{D}_t$ and $\mathcal{T}_s \neq \mathcal{T}_t$, but $\mathcal{D}_s \sim \mathcal{D}_t$ and $\mathcal{T}_s \sim \mathcal{T}_t$. The main difference between OSL and UOSDA is that 1) in OSL, we assume that $P_{X_s Y_s} = P_{X_t Y_t | Y_t \in \mathcal{Y}_s}$, but UOSDA does not need the assumption; and 2) in OSL, we have not any target samples used for training models, however, UOSDA needs additional target samples during training process.

A common challenge for UOSDA and OSA is to discover the boundary to distinguish known target samples and unknown target samples (i.e., known target samples are target samples, whose labels are from $\mathcal{Y}_s$ and unknown target samples are target samples, whose labels are from $\mathcal{Y}_t - \mathcal{Y}_s$). However, due to the availability of target samples during training process, the strategies to address these two problems are different.

2.2.2.1 Algorithms for Unsupervised Open-Set Domain Adaptation

Here we introduce several representative UOSDA algorithms.

• The open-set domain adaptation problem is proposed by Assign-and-Transform-Iteratively (ATI) [27]. Using ℓ_2 distance between each target samples and the center of each source class, ATI constructs a constraint integer programming to recognize unknown target samples, then learns a linear transformation to match the source samples and predicted known target samples. However, as the first work of open-set domain adaptation, ATI requires the help of known unknown source samples, which are unavailable in our setting.

• The first deep learning-based algorithm, Open-Set Back Propagation (OSBP) is proposed by [28]. OSBP relies on an adversarial neural network and a binary cross entropy loss to learn the probability of the target samples. It then uses the estimated probability to separate samples of known and unknown classes in the target.

• Separate to Adapt (STA) [103] is the first algorithm to address UOSDA

by instance-weighting strategy. STA proposes a coarse-to-fine weighting mechanism to progressively separate the known target samples and unknown target samples, and simultaneously re-weight the target samples using estimated weights during feature matching process.

• Luo et al. [33] proposed Progressive Open-Set Unsupervised Domain Adaptation (POSUDA). This work is based on a Progressive Graph Learning (PGL) framework. Considering the shift between conditional distribution $P_{Y_s|X_s}$ and $P_{Y_t|X_t}$, POSUDA addresses the UOSDA from two perspectives: sample-level and manifold-level. As we know, POSUDA is the first work using graph neural networks to address UOSDA.

• Factorized Representations for OSDA (FROSDA) is proposed by Baktashmotlagh et al. [104]. FROSDA is a subspace-based algorithm, which assumes that the source samples and known target samples can be generated by a shared subspace, but the unknown target samples are from a different, private subspace. Hence, FROSDA factorizes samples into shared subspace and private subspace.

2.2.2.2 Background and Algorithms for Open-Set Learning

Background for Open-Set Learning. Supervised learning has achieved dramatic successes in many applications such as object detection [105], speech recognition [106] and natural language processing [107]. These successes are partly rooted in the closed-set assumption that training and test samples share the same label space. Under this assumption, the standard supervised learning is also regarded as closed-set learning (CSL) [108–110].

However, the closed-set assumption is not realistic during the testing phase (i.e., there are no labels in the samples), since it is not known whether the classes of test samples are from the label space of training samples. Test samples may come from some classes (unknown classes) that are not necessarily seen during training. These unknown classes can emerge unexpectedly and drastically weaken the performance of existing closed-set algorithms [111–113].

To solve supervised learning without closed-set assumption, [32] proposed a new problem setting, open-set learning (OSL), in which the test samples can come from any classes, even unknown classes. An open-set classifier should classify samples from known classes into correct known classes while recognizing samples from unknown classes into unknown classes.

Algorithms for Open-Set Learning. Remarkable advances have been achieved in open-set learning. The key challenge of OSL is to recognize the unknown test samples accurately. To address this challenge, different strategies have been proposed.

We can roughly separate OSL algorithms into two different categories: shadow algorithms (e.g., support vector machine (SVM)) and deep learning-based algorithms.

• By constructing open-space risk, Scheirer et al. [32] proposed a SVM-based OSL algorithms. Open-space risk, as one of important indexes to estimate the classification performance of known classes and unknown classes, has motivated many OSL algorithms [114, 115].

• Jain et al. [116], Rudd et al. [117] proposed OSL algorithms based on extreme value theory. Extreme value theory is a branch of statistics analyzing the distribution of samples of abnormally high or low values. The extreme value theory becomes one of the basic tools in OSL.

Recently, deep-based algorithms have been developed dramatically.

• OpenMax as the first deep-based algorithms is proposed by [114], to replace SoftMax in deep networks. Later, Ge et al. [115] combined the generative adversarial networks (GAN) with OpenMax and proposed G-OpenMax.

• Counterfactual Image Generation is proposed by [118]. It is the first OSL algorithm to use the data augmentation technique by generating the unknown classes so that the decision boundaries between unknown and known samples can be figured out.

• Neal et al. [119] used class conditioned auto-encoders to solve OSL problem, and modeled reconstruction errors using the extreme value the-

ory to find the threshold for identifying known/unknown samples.

2.2.3 Semi-supervised Heterogeneous Domain Adaptation

As one of the main problems that this thesis aims to address, we recall the definition (given in Definition 3) of semi-supervised heterogeneous domain adaptation as follows: *Assume that feature spaces $\mathcal{X}_s$, $\mathcal{X}_t$ and label sets $\mathcal{Y}_s$, $\mathcal{Y}_t$ satisfy $\mathcal{X}_s \neq \mathcal{X}_t$, $\mathcal{Y}_s = \mathcal{Y}_t$. Given sets of samples called the labeled source and unlabeled target samples:*

$$\mathcal{S} = \{(\mathbf{x}_s^i, \mathbf{y}_s^i)\}_{i=1}^{n_s} \sim P_{X_s Y_s} \ i.i.d.$$
$$\mathcal{T}_l = \{(\mathbf{x}_l^i, \mathbf{y}_l^i)\}_{i=1}^{n_l} \sim P_{X_t Y_t} \ i.i.d.$$
$$\mathcal{T}_u = \{\mathbf{x}_u^i\}_{i=1}^{n_u} \sim P_{X_t} \ \ i.i.d.$$

where $n_l \ll n_u$, $n_l \ll n_s$. The aim of ***semi-supervised heterogeneous domain adaptation*** *is to train a classifier $g : \mathcal{X}_t \to \mathcal{Y}_t$ such that g classifies the unlabeled target samples accurately by using labeled and unlabeled samples.*

From above definition, we can see that: 1) $\mathcal{X}_s \neq \mathcal{X}_t$, because our focus is on the heterogeneous feature spaces; 2) there are few target labels, because we aim to solve the semi-supervised situation. In addition, we assumes that Y_s and Y_t are discrete random variables, since we focus on the classification task.

In SsHeDA field, most existing algorithms align the source and target samples by constructing effective feature transformations [29]. Seven representative SsHeDA algorithms are reviewed as follows:

• Manifold Alignment. Domain Adaptation with Manifold Alignment (DAMA) [120] utilizes the manifold alignment technique [121] to learn the source and target linear transformations to achieve three goals: 1) the typologies of domains are preserved; 2) the samples from the same class are transformed to similar locations; 3) the samples from different classes are separated from each other. Hence, DAMA aligns the source

and target domains from the geometric view.

• Feature Augmentation. Sparse Heterogeneous Domain Adaptation (SHFA) [24] utilizes the augmented feature transformations to align the source and target domains. Augmented feature transformations are special linear projections, which map the source samples and target samples to a higher dimensional space. By incorporating the original features into the augmented features, SHFA enhances the similarities between samples from domains.

• Pseudo-label Strategy. Generalized Joint Distribution Adaptation (G-JDA) [122] extends the classical UHoDA algorithm JDA [65] to the heterogeneous scenarios. G-JDA constructs two feature transformations to project the source and target samples into a latent space, where the marginal and class-conditional distributions are matched. Lastly, Pseudo-label iteration technique is used to update target labels.

• Instance Weighting. Cross Domain Landmarks Selection (CDLS) [30] does not regard samples as being of equal importance during the domain matching process. CDLS learns two linear feature transformations and estimates the weights for the source and target samples at the same time. To estimate the discrepancy between transformed domains, CDLS utilizes the MMD [62].

• Covariance Matching. Domain Adaptation by Covariance Matching (DACoM) [123] learns the source and target subspaces with the following three aspects: 1) the transformed domains are matched with higher order moments; 2) the transformed domains preserve latent spectral structures; 3) the transformed domains contain consistent discriminate information.

• Transfer Neural Tree. Transfer Neural Trees (TNT) [124] jointly solves the source and target feature transformations, domain alignment, and classification in a neural network-based architecture. The structure consistency between target samples is preserved by introducing an embedding loss in the layer of the target feature transformation.

• Soft-label Iteration. Soft Transfer Network (STN) [23] considers the soft-label strategy of unlabeled target samples to address the SsHeDA

problem. STN introduces an adaptive coefficient to gradually increase the importance of the soft labels, which will become more accurate as the number of iterations increases.

Chapter 3

A Manifold-based Algorithm for Unsupervised Homogeneous Domain Adaptation

This chapter proposes a manifold-based algorithm to address the unsupervised homogeneous domain adaptation (UHoDA).

3.1 Introduction

A main problem of unsupervised homogeneous domain adaptation lies in reducing the discrepancy of distributions in two domains [4, 5, 21]. To handle distribution changes, many existing unsupervised homogeneous domain adaptation algorithms search for a new feature space, known as domain-invariant feature, to make the marginal distributions between two domains similar while utilizing the source classifier learnt from the domain-invariant feature to classify the target domain, see Fig. 3.1(b). However, the domain-invariant feature algorithms have an inherent assumption that the source classifier can be generalized to the target domain in the new feature space.

The assumption of domain-invariant feature algorithms is not realistic in an unsupervised setting (i.e., when there are no labels in the target domain): it is not known if the source classifier can perform well on the target domain in the new feature space. Examples have been constructed by [125] to make the assumption invalid and [125] has shown how, even in the case of perfectly aligned representations between the source and target domains, a small error source classifier performs poorly in the

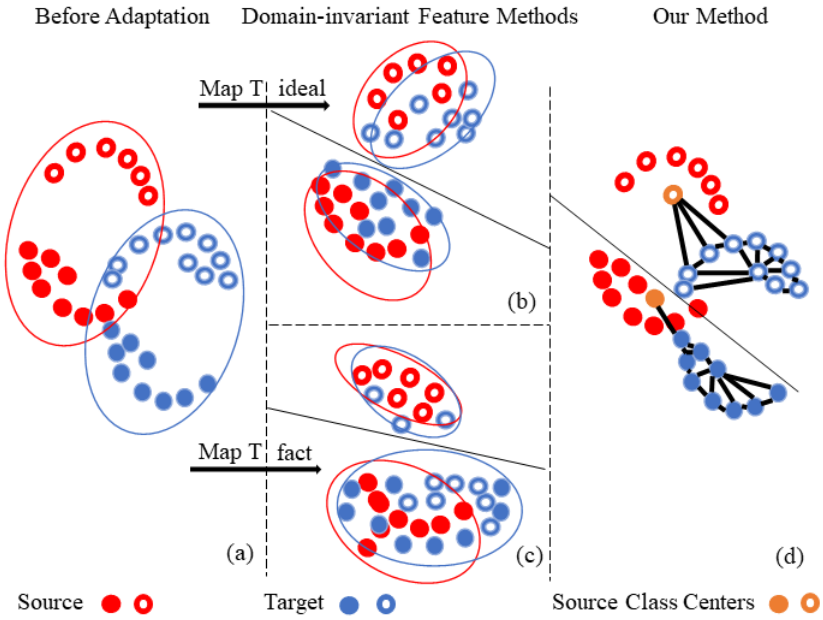


Figure 3.1: (a) Source domain and target domain are given. (b) The domain-invariant feature algorithms learn a map to seek a new feature space where domains are similar and assume the learned small error source classifier can be generalized to the target domain. (c) The domain-invariant feature algorithms match the marginal distributions but discrepancy between conditional distributions are enlarged. (d) Our algorithm (MFC) learns the geometric structure among target samples and source class centers while utilizing the fuzzy c-mean clustering to help classification rather than seeking an unknown feature space.

target domain.

A main reason for the failure of these algorithms is that, although the learnt new feature representation can align the marginal distributions well, the discrepancy between conditional distribution may be enlarged (Fig. 3.1(c)). To overcome the problem of domain-invariant feature algorithms, many algorithms [94, 96] have attempted to match conditional distributions. However, those techniques still suffer from the same problem because no any truth target labels can be used. They are restricted to using only pseudo labels to predict the target conditional distribution.

A simple algorithm of avoiding the problem of the domain-invariant feature algorithms is to abandon seeking such a feature and turn to mine the intrinsic transfer information while combining the domain structures, such as those of manifold and cluster, to learn target classifier. Motivated by this idea, we design a new UHoDA algorithm, referred to as Manifold

Fuzzy Clustering (MFC), which primarily integrates manifold learning [121] and fuzzy clustering [126] to guide the transferring process.

Two observations support our work. To begin with, the concept of "discrepancy between domains" itself is very fuzzy because using different algorithms and aspects to measure the discrepancy between domains leads to different evaluations and results. Consequently, it is unreasonable to apply "hard" classification to represent and solve a problem with fuzzy intrinsic characteristics. Fuzzy theory provides an effective solution. A second observation is that the visual data used in this work has both the manifold and the cluster structures [121, 127]. These characteristics will be strengthened through the feature extraction process, in which the CNN structure could facilitate the representation of both structures.

Looking more closely, MFC employs the fuzzy c-means clustering method to cluster the target domain while learning the manifold information to extract the underlying structure between domains. To bridge the source and target domains, the source class centers are applied to replace the unknown target class centers in clustering process. Finally, the clustered target samples acquire the membership values to recognize the importance of every class.

The main characteristics of MFC are:

• MFC explores the intrinsic manifold structure of target samples to improve the performance of fuzzy c-means clustering, further enhancing classification accuracy in target domain.

• MFC mainly utilizes fuzzy c-means clustering to classify target samples. The ablation study has showed the significant and indispensable role of fuzzy c-means clustering in MFC.

• MFC is easily tuned. There exist only two hyperparameters in the proposed MFC and experiments show that a wide range of parameter values can be selected to obtain a satisfactory performance, which is extremely beneficial for real-world applications.

• MFC only leverages the source class centers, a very robust algo-

rithm, even when labeled source samples are insufficient. As such, this algorithm also offers a promising solution to source label insufficiency.

• MFC has been compared with some competitive unsupervised domain algorithms on 48 real-world UHoDA tasks and a synthetic data task. Extensive experiments demonstrate the effectiveness of our algorithm.

3.2 Proposed Algorithm

In this section, we introduce a novel UHoDA algorithm, called Manifold Fuzzy Clustering (MFC). MFC uses the manifold regularization to learn the underling information of samples while using fuzzy c-means clustering [126] to group the target samples. Note that a clustering algorithm could not provide label information for these clusters. So, we transfer labels of source centers to the target domain to recognize the clusters and complete the classification task. We will start with a review of fuzzy c-means (FCM) [126], before beginning to describe our algorithm by introducing manifold regularization [121].

3.2.1 Fuzzy c-means Clustering

Fuzzy c-means (FCM) [126] is applicable to a wide variety of data analysis problems. A basic assumption for FCM is that samples have clustering structure.

Let $\{\mathbf{x}^i\}_{i=1}^n \subset \mathbb{R}^d$ be samples, where n is the number of samples and d is the dimension of samples. Suppose $\{\mathbf{x}^i\}_{i=1}^n$ are clustered to K clusters, then the membership matrix $\mathbf{U} = [u_{ij}]_{K\times n}$ is the matrix representation of the membership values for different clusters and u_{ij} satisfies following conditions:

$$\begin{aligned} &u_{ij} = u_i(\mathbf{x}^j) \in [0,1], \\ &\Sigma_{i=1}^K u_{ij} = 1, \\ &\Sigma_{j=1}^n u_{ij} > 0, \end{aligned} \tag{3.2.1}$$

where u_{ij} represents membership of samples $\mathbf{x}^j$ belonging to ith cluster.

To compute the matrix for membership values, the FCM algorithm

minimizes the following optimization problem:

$$\mathcal{F}(\mathbf{U}, \mathbf{V}) = \Sigma_j^n \Sigma_i^K \ u_{ij}^m \text{dist}(\mathbf{x}^j, \mathbf{v}^i)^2, \tag{3.2.2}$$

where $\mathbf{V} = \{\mathbf{v}^i\}_{i=1}^K$ is cluster centers, m is the weighting exponent, and $\text{dist}(\cdot, \cdot)$ is the distance function defined in $\mathbb{R}^d \times \mathbb{R}^d$. From Eq. (3.2.2), we observe that, if $\mathbf{V}$ is fixed, the membership matrix $\mathbf{U}$ is intimately bound up with the geometric distance between $\{\mathbf{x}^i\}_{i=1}^n$ and $\mathbf{V}$.

Although the FCM algorithm mainly exploits the geometric distance between samples $\mathbf{X}$ and centers $\mathbf{V}$, it omits the intrinsic geometric structure among samples. By considering the inner geometric relationship of samples, we introduce manifold regularization.

3.2.2 Manifold Regularization

In manifold learning, samples have special manifold structure, which will influence the membership values. To be more specific, if two points $\mathbf{x}^1, \mathbf{x}^2$ are close in the underlying manifold structure, then their membership values $u_i(\mathbf{x}^1), u_i(\mathbf{x}^2)$ should also be similar.

Extracting such geometry-based correlation and integrating it with membership function learning process could enhance the ability to explore samples' intrinsic structures and relations. On this basis, we add a manifold regularization term to learn the intrinsic geometric information of samples when learning the membership function.

Before introducing the manifold regularization, we define the pairwise affinity matrix as follows:

$$\mathbf{W}_{ij} = \begin{cases} \cos(\mathbf{x}^i, \mathbf{x}^j), & \mathbf{x}^i \in \mathcal{N}_p(\mathbf{x}^j) \text{ or } \mathbf{x}^j \in \mathcal{N}_p(\mathbf{x}^i) \\ 0, & \text{otherwise} \end{cases} \tag{3.2.3}$$

where $\mathcal{N}_p(\mathbf{x}^i)$ denotes the set of p-nearest neighbors to point $\mathbf{x}^i$ and p is a free parameter.

Based on the affinity matrix $[\mathbf{W}_{ij}]$, the manifold regularization can then be formulated as follows:

$$\begin{aligned}\mathcal{M}[\{\mathbf{x}^i\}_{i=1}^n](\mathbf{U}) &= \sum_{l=1}^{K}\sum_{i,j=1}^{n}(u_l(\mathbf{x}^i)-u_l(\mathbf{x}^j))^2\mathbf{W}_{ij} \\ &= \sum_{l=1}^{K}\sum_{i,j=1}^{n}u_l(\mathbf{x}^i)\mathbf{N}_{ij}u_l(\mathbf{x}^j) \\ &= \text{trace}(\mathbf{UNU}^T),\end{aligned} \tag{3.2.4}$$

where $\mathbf{U}$ is the membership matrix, and $\mathbf{N}$ is the Laplacian matrix, which can be written as $\mathbf{D}-\mathbf{W}$, here $\mathbf{D}$ is a diagonal matrix and $\mathbf{D}_{ii} = \sum_{j=1}^{n}\mathbf{W}_{ij}$.

3.2.3 Manifold Fuzzy Clustering

Manifold Fuzzy Clustering is proposed by combining FCM and manifold regularization to reflect the membership values of clusters while containing the samples' geometric structure. The membership matrix is reformulated by adding manifold regularization as follows:

$$\begin{aligned}\mathbf{U} &= \underset{\mathbf{U}=[u_{ij}]_{K\times n_u}}{\arg\min}\ \mathcal{L}(\mathbf{U},\mathbf{V}) \\ &= \underset{\mathbf{U}=[u_{ij}]_{K\times n_u}}{\arg\min}\ \mathcal{M}[\mathcal{T}_u\cup\mathbf{V}](\mathbf{U})+\lambda\mathcal{F}(\mathbf{U},\mathbf{V}) \\ &= \underset{\mathbf{U}=[u_{ij}]_{K\times n_u}}{\arg\min}\ \text{trace}(\mathbf{UNU}^T)+\lambda\Sigma_j^n\Sigma_i^K\ u_{ij}^m\text{dist}(\mathbf{x}^j,\mathbf{v}^i)^2,\end{aligned} \tag{3.2.5}$$

subject to

$$\begin{aligned}0 \le u_{ij} \le 1, \\ \Sigma_{i=1}^K u_{ij} = 1,\end{aligned}$$

where $\mathcal{T}_u = \{\mathbf{x}_u^i\}_{i=1}^{n_u}$ are target samples, $\mathbf{V}$ are the cluster centers, $\mathcal{M}[\mathcal{T}_u\cup\mathbf{V}](\mathbf{U})$ is defined by Eq. (3.2.4), $\mathcal{F}(\mathbf{U},\mathbf{V})$ is defined by Eq. (3.2.2), $\mathbf{N}$ is the Laplacian matrix given in Eq. (3.2.4) and $\lambda \ge 0$ is MFC's parameter that impacts the level of fuzziness.

By solving problem (3.2.5), the target samples could be clustered and we obtain the memberships to each cluster, which determine the belongings of samples to clusters. But our final classification task is yet to be completed, since no labels are specified to these clusters under the problem setting, unsupervised domain adaptation.

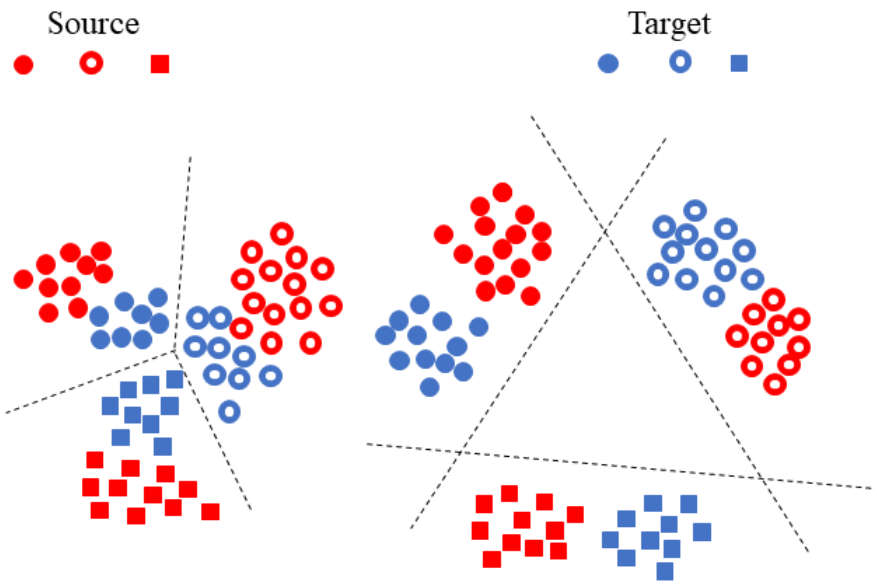


Figure 3.2: Two types of distribution discrepancy between source domain and target domain.

So, how to leverage the knowledge of source domain and label the target clusters becomes the key problem. We propose a center-based transfer method, which is prompted by two aspects:

1) Distribution discrepancy minimization is the most commonly used method in transfer learning, but in some scenarios, distribution discrepancy between source and target domains is not a good indicator. For example, Fig. 3.2 shows two typical scenarios where target and source domains vary. Comparing the left scenario with the right reveals a larger magnitude domain shift which maintains linear separability. However, modelling the domain discrepancy and minimizing it is unsatisfactory for the right scenario.

2) We apply CNN to extract the features which confirm that the samples of the same class across domains are sufficiently close and that different classes have sufficiently large margins. This implies that target samples are very likely to be classified correctly without minimizing a distributions gap in two domains. It also provides a promising method to leverage the discriminative information from the target domain, even when the target labels are not available.

Inspired by 1) and 2), our method only exploits the class mean to

Algorithm 1 Manifold Fuzzy Clustering (MFC)

Input: Data $\mathcal{S}, \mathcal{T}_u$; parameters λ; #neighbor p.

1. $\mathbf{V}_s \leftarrow \mathcal{S}$;% Source class centers;
2. Compute distance matrix $\text{dist}(\mathcal{T}_u, \mathbf{V}_s)^2$;
3. Compute manifold term $\mathcal{M}[\mathcal{T}_u \cup \mathbf{V}_s]$ by Eq. (3.2.4);
4. Compute membership values $\mathbf{U}$ by QP solver according to Eq. (3.2.5);
5. $\hat{\mathbf{V}}_t \leftarrow \mathcal{T}_u, \mathbf{U}$;% Compute pseudo target class centers by Eq. (3.2.6);
6. Compute distance matrix $\text{dist}(\mathcal{T}_u, \hat{\mathbf{V}}_t)^2$;
7. Compute manifold term $\mathcal{M}[\mathcal{T}_u \cup \hat{\mathbf{V}}_t]$ by Eq. (3.2.4);
8. Compute membership values $\mathbf{U}$ by QP solver according to Eq. (3.2.5);
9. $\widetilde{\mathbf{Y}}_t(j) = \arg\max_{i=1,...,K}[u_{1j}, u_{2j}, ..., u_{Kj}]$;% Predict target label for the samples $\mathbf{x}_u^j$.

Output: Predicted target labels $\widetilde{\mathbf{Y}}_t$; Membership matrix $\mathbf{U}$.

characterize class-level difference and transfer it from source domain to target domain. Specifically, we replace the target cluster centers with source cluster centers when solving problem (3.2.5). More importantly, all source samples are labeled, as this process also labels the clusters in the target domain with labeled source cluster centers.

Initially we set the labeled source centers as target cluster centers. Then, the pseudo class centers of target samples can be computed by the membership matrix in problem (3.2.5). Similar to using FCM, we compute the pseudo class centers of target samples as follows:

$$\hat{\mathbf{v}}_t^i = \frac{\sum_{j=1}^{n_u} u_{ij}^m \mathbf{x}_u^j}{\sum_{j=1}^{n_u} u_{ij}^m}, \tag{3.2.6}$$

where $\mathbf{x}_u^j \in \mathcal{T}_u$ is the target samples and $\hat{\mathbf{v}}_t^i$ is the ith pseudo class center of the target domain.

For the solving of problem (3.2.5), the manifold regularization is a quadratic equation and the FCM clustering term is an m-order equation, where m is the weighting exponent of MFC. If we set $m = 1$, our optimization problem becomes the **quadratically constrained quadratic programming**, which can be resolved easily by any QP solver. The logical choice in this chapter is clearly $m = 1$. The implementation details of Manifold Fuzzy Clustering are demonstrated in Algorithm 1.

3.2.4 Influence of Parameter λ

λ is an important parameter in problem (3.2.5) and influences the level of fuzziness. We note that in FCM, the parameter m impacts on the level of fuzziness. Following theorem shows that when λ approaches $+\infty$, our MFC degenerates to a "hard" algorithm. Theorem 1 verifies our conjecture.

Theorem 1. *If* $\mathbf{V} = \{\mathbf{v}^i\}_{i=1}^K$ *are fixed,* $\mathbf{v}^i \neq \mathbf{v}^j$*, when* $i \neq j$*, and* $\text{dist}(\mathbf{x}^k, \mathbf{v}^i) \neq \text{dist}(\mathbf{x}^l, \mathbf{v}^j)$*, when* $(k, i) \neq (l, j)$*, then the solution* $[u_{ij}]$ *of the optimization problem*

$$\begin{aligned} &\min_{\mathbf{U}} \mathcal{L}(\mathbf{U}, \mathbf{V}) \\ \text{subject to } \ &\Sigma_{i=1}^K u_{ij} = 1, \quad 0 \leq u_{ij} \leq 1; \end{aligned} \tag{3.2.7}$$

is

$$\lim_{\lambda \to +\infty} u_{ij} = \begin{cases} 1, & \text{dist}(\mathbf{x}^j, \mathbf{v}^i) = \min\limits_{l \in \{1,..,K\}} \text{dist}(\mathbf{x}^j, \mathbf{v}^l); \\ 0, & \text{otherwise.} \end{cases} \tag{3.2.8}$$

Proof. Let M be the upper bound of the manifold term $\mathcal{M}[\mathcal{T}_u \cup \mathbf{V}](\mathbf{U})$:

$$0 \leq \mathcal{M}[\mathcal{T}_u \cup \mathbf{V}](\mathbf{U}) \leq M,$$

for any $\mathbf{U}$ satisfies the conditions $\Sigma_{i=1}^K u_{ij} = 1, \quad 0 \leq u_{ij} \leq 1$.

If $\text{dist}(\mathbf{x}^j, \mathbf{v}^i) = \min_{l \in \{1,..,K\}} \text{dist}(\mathbf{x}^j, \mathbf{v}^l)$, we denote that

$$\delta = \min_{l \neq i} \text{dist}(\mathbf{x}^j, \mathbf{v}^l)^2 - \text{dist}(\mathbf{x}^j, \mathbf{v}^i)^2.$$

According to the condition $\text{dist}(\mathbf{x}^k, \mathbf{v}^i) \neq \text{dist}(\mathbf{x}^l, \mathbf{v}^j)$, when $(k, i) \neq (l, j)$, we know that $\delta > 0$.

Given any $\tilde{\mathbf{U}} = [\tilde{u}_{kl}]$, then we define a new membership matrix $\hat{\mathbf{U}}$ whose elements $\hat{u}_{ij} = 1$, $\hat{u}_{lj} = 0$ $(l \neq i)$ and other elements $\hat{u}_{lk} = \tilde{u}_{lk}$. Then we

compute

$$\begin{aligned}
&\mathcal{L}(\hat{\mathbf{U}}, \mathbf{V}) - \mathcal{L}(\tilde{\mathbf{U}}, \mathbf{V}) \\
=&\mathcal{M}[\mathcal{T}_u \cup \mathbf{V}](\hat{\mathbf{U}}) - \mathcal{M}[\mathcal{T}_u \cup \mathbf{V}](\tilde{\mathbf{U}}) \\
+&\lambda \Sigma_l^K \ \hat{u}_{lj} \text{dist}(\mathbf{x}^j, \mathbf{v}^l)^2 - \lambda \Sigma_l^K \ \tilde{u}_{lj} \text{dist}(\mathbf{x}^j, \mathbf{v}^l)^2 \\
\leq& M - \lambda\delta(1 - \tilde{u}_{ij}).
\end{aligned} \tag{3.2.9}$$

Now let $\tilde{\mathbf{U}} = [\tilde{u}_{kl}]$ be the optimizer for (3.2.7) with parameter λ. According to inequality (3.2.9), we note that $0 \leq \mathcal{L}(\hat{\mathbf{U}}, \mathbf{V}) - \mathcal{L}(\tilde{\mathbf{U}}, \mathbf{V}) \leq M - \lambda\delta(1 - \tilde{u}_{ij})$. Hence,

$$\tilde{u}_{ij} \geq 1 - \frac{M}{\lambda\delta}, \tag{3.2.10}$$

which implies that:

$$1 \geq \lim_{\lambda \to +\infty} \tilde{u}_{ij} \geq \lim_{\lambda \to +\infty} 1 - \frac{M}{\lambda\delta} = 1. \tag{3.2.11}$$

Combining the condition $\Sigma_{l=1}^K \widetilde{u}_{lj} = 1$, we obtain the result we want to prove. □

Theorem 1 implies that, with a larger λ, the "harder" our fuzzy model is. It also indicates that the manifold factor may contribute to the fuzziness of the constructed model. So, we design experiments in Section 3.3 to validate this.

3.3 Experimental Results

This section presents quantitative and qualitative results of the proposed MCF. We begin by conducting experiments on a synthetic dataset to demonstrate the critical problem suffered by the domain-invariant feature algorithms and to indicate how our MFC algorithm works. Five real-world datasets are used to verify the effectiveness of MFC by comparing it with existing UHoDA algorithms. Further experiments are designed to analyze the parameters in MFC algorithm.

3.3.1 Baseline Algorithms

Several classical algorithms are used as baselines, the basic ideas of these algorithms are shown as follows.

No Adaptation

• 1NN. 1-nearest neighbor, which is implemented without transfer learning.

Geodesic Flow Kernel Algorithm

• GFK [66], which connects two domains with geodesic curves on a Grassmann manifold.

Correlation Alignment Algorithm

• CORAL [68], which performs second-order subspace alignment.

Domain-invariant Feature Algorithms

• TCA [5], which seeks linear subspace and uses MMD [62] to match distributions.

• DANN [22], a deep algorithm which uses adversarial neural networks to learn the domain-invariant feature.

• DAN [8], which is a deep algorithm that learns the domain-invariant feature by MK-MMD.

Domain-invariant Feature Algorithms with Conditional Distribution Matching

• JDA [65], which jointly matches the marginal distributions and conditional distributions.

• JGSA [35], which not only considers the distribution discrepancy but also matches the geometric shift.

We use the parameters recommended by the original papers for all the baselines.

3.3.2 Synthetic Data

A dataset is synthesized to demonstrate the one issue that the domain-invariant feature algorithms cannot avoid and to illustrate how MFC overcome this obstacle.

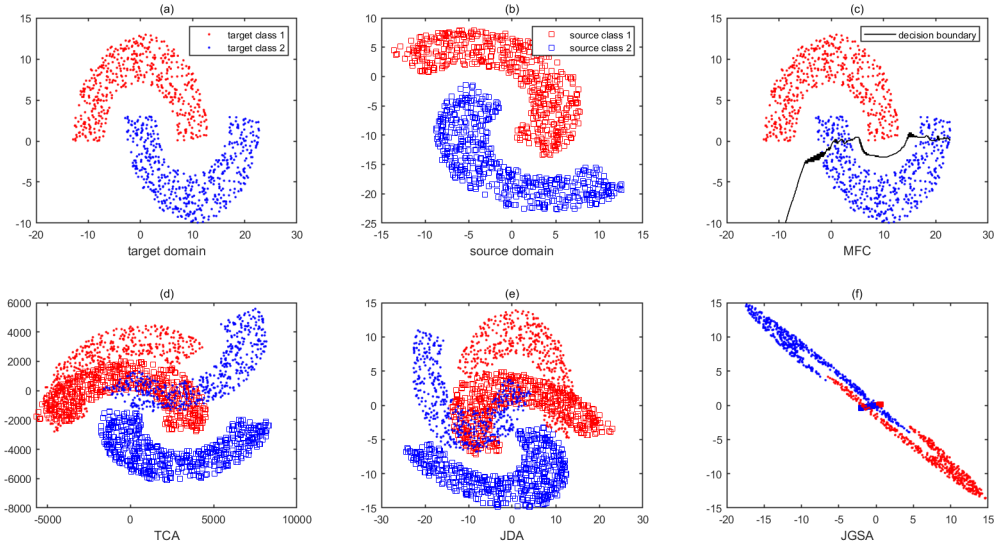


Figure 3.3: Comparisons of baselines and the proposed MFC algorithm on the synthetic samples

Fig. 3.3 shows the synthesized datasets and the corresponding transfer learning results of different algorithms on this dataset. The first two figures illustrate synthesized source sample. Source domain and target domain are shown in Fig. 3(a) and Fig. 3(b). Target domain is a two-moons dataset, as shown in Fig. 3(a). This dataset includes two classes, each containing 1000 samples. Source domain is constructed by rotating and translating the target domain sample, as shown in Fig. 3(b). Since the synthetic samples are generated randomly, we conduct our experiments 1000 times and report the mean accuracy and standard deviation in Table 3.1.

From the figures for TCA, JDA, and JGSA, we can see that the discrepancy between marginal distributions is reduced (in that the center of sample is aligned). Although target class 1 and source class 1 are aligned well, target class 2 and source class 1 are mixed, which implies that the discrepancy of conditional distributions between domains is enlarged. Therefore, it is impossible for algorithms TCA, JDA and JGSA to classify the target class 2 well.

The results in Table 3.1 also verify our analysis, where the performance of MFC is superior to the other six algorithms. There are three reasons for the superiority of the MFC algorithm: 1) MFC exploits the

Table 3.1: Accuracy (%) on Synthetic datasets.

Algorithms	1NN	GFK	CORAL	TCA	JDA	JGSA	MFC
Avg	51.2	68.3	52.9	52.1	53.0	43.6	**89.0**
Sd	0.36	2.00	0.95	0.61	1.03	18.58	1.43

geometric structure of target sample. By learning the manifold term, MFC can use the similarity of sample to make sure that if two sample $\mathbf{x}_1, \mathbf{x}_2$ are close in the intrinsic geometry, then the membership values $u_i(\mathbf{x}_1)$ and $u_i(\mathbf{x}_2)$ are similar; 2) the FCM term helps MFC to learn relationship between the source and the target domains so that MFC can exploit the labeled source sample to classify the target sample; 3) without trying to seek the domain-invariant feature, the divergence between conditional distributions is preserved.

3.3.3 Real World Datasets

We evaluate our algorithm on five famous UHoDA datasets: **Office-31** [128], **Office-Home** [129] , **Office-Caltech** [128,130], **Image-CLEF DA** [47] and **Amazon Review** [131]. Table 3.2 lists statistics of these datasets.

Table 3.2: Introduction of datasets.

Dataset	#data	#Feature	#Class	Domain
Office	1,410	4,096	10	A,W,D
Caltech	1,123	4,096	10	C
Office-31	4,110	4,096	31	A,W,D
Image-CLEF DA	1,800	2,048	12	C,I,P
Amazon Review	7,996	400	2	B,E,D,K
Office-Home	15,500	2,048	65	Ar,Cl,Pr,Rw

Office+Caltech10 consists of 2,533 images in 10 categories, forming four domains: AMAZON (**A**), WEBCAM (**W**), DSLR (**D**), and CALTECH (**C**). By considering one as source domain and one as target domain, twelve transfer learning tasks can be constructed: **A** $\rightarrow$ **D**, **A** $\rightarrow$ **W**, **D** $\rightarrow$ **A**,..., **W** $\rightarrow$ **C**. We use features extracted from datasets in DeCAF6.

Office-31 has 4,652 images with 31 common categories, and consists

of three real-world object domains: AMAZON (**A**), DSLR (**D**) and WEBCAM (**W**). Similarly, six transfer learning tasks are implemented: **A** $\rightarrow$ **D**, **A** $\rightarrow$ **W**, **D** $\rightarrow$ **A**, **W** $\rightarrow$ **A**, **D** $\rightarrow$ **W**, **W** $\rightarrow$ **D**. Following the standard protocol and for a fair comparison with the other algorithms, we extracted feature vectors from ResNet-50.

ImageCLEF-DA is a benchmark dataset for ImageCLEF 2014 transfer learning challenge, which aims to classify 12 categories shared in three datasets: Caltech-256 (**C**), ImageNet ILSVRC 2012 (**I**), and PascalVOC 2012 (**P**). By considering each as a domain, we build six transfer tasks: **C** $\rightarrow$ **I**, **C** $\rightarrow$ **P**, **I** $\rightarrow$ **P**, **I** $\rightarrow$ **C**, **P** $\rightarrow$ **I**, **P** $\rightarrow$ **C**. Following the standard protocol and for a fair comparison with the other algorithms, we extracted feature vectors from ResNet-50.

Amazon Review is a cross-domain sentiment analysis dataset that contains positive and negative reviews of four kinds of products: Kitchen appliance (**K**), DVDs (**D**), Electronics (**E**), and Books (**B**). We use all domain combinations and build twelve transfer tasks: **K** $\rightarrow$ **D**, **K** $\rightarrow$ **E**,..., **B** $\rightarrow$ **D**, **B** $\rightarrow$ **E**, **B** $\rightarrow$ **K**.

Office-Home consists of 4 different domains: Artistic (**Ar**), Clipart (**Cl**), Product (**Pr**) and Real-World (**Rw**). Each domain contains images from 65 object classes. We constructed 12 transfer learning tasks: **Ar** $\rightarrow$ **Cl**, **Ar** $\rightarrow$ **Pr**,..., **Rw** $\rightarrow$ **Ar**. Following the standard protocol and for fair comparison with the other algorithms, we also extracted feature vectors from ResNet-50.

3.3.4 Hyper-parameter Settings

Before reporting the detailed evaluation results, it is important to explain how MFC hyper-parameters are tuned. There are two hyper-parameters for MFC: 1) the number of neighbours p; 2) λ. Empirically, we set $p = 10$. And for λ, we need to interpret how to further tune λ. When λ becomes smaller, the manifold term becomes more important. This means more geometric information of target samples could be extracted. When λ becomes larger, MCF is more similar to FCM, allowing more information

between target samples and the cluster centers $\mathbf{V}$ to be learned.

We designed experiments that choose the parameter λ from $0, 0.01, 0.05, 0.1, 0.5, 1, 5, 10$, and the results verified that fixed λ could obtain promising results on different tasks. Empirically, we set the parameter λ to be set at 10 for tasks **Office+Caltech10**, **Office-Home**, **Amazon Review**, and be set to 1 for tasks **ImageCLEF-DA** and **Office-31**. Further, we provide parameter sensitivity analysis for λ and p, which will verify that MFC can achieve stable performance for a wide range of hyper-parameter settings.

3.3.5 Experimental Results

The classification accuracy of our proposed MFC algorithm and other compared algorithms on five datasets are shown on Tables 3.3-3.7. From observing and analyzing the results in these tables, we could summarize the following conclusions.

1) The performance of unsupervised transfer learning algorithms is always better than the algorithms without adaptation, 1NN and ResNet50. This is because samples at the source and the target are from different domains.

2) MFC algorithm achieves much better performance than the baseline algorithms on most (32 out of 48) tasks. The average classification accuracy of MFC on five datasets is 80.3%, showing a performance improvement of 3.7% when compared with the best baseline JGSA.

3) We report results of **Office+Caltech10**, **Office-31** and **Office-Home** in Tables 3.3-3.5. We observe that MFC outperforms all comparison algorithms on most tasks. Specifically, large margins are obtained on the tasks $\mathbf{A} \rightarrow \mathbf{C}$ (2.8%), $\mathbf{D} \rightarrow \mathbf{C}$ (2.0%) and $\mathbf{W} \rightarrow \mathbf{C}$ (2.8%) on **Office+Caltech10**, tasks $\mathbf{A} \rightarrow \mathbf{D}$ (3.9%) and $\mathbf{W} \rightarrow \mathbf{A}$ (1.4%) on **Office-31** and almost all tasks (avg 4.0%) on **Office-Home**, when comparing MFC with other domain-invariant feature algorithms. The reason for this is that those tasks all have relatively large domain discrepancies and imbalanced domain scales.

4) On **ImageCLEF-DA** and **Amazon Revoew** datasets, MFC exceeds baseline algorithms on most tasks. MFC is superior than other algorithms averagely by 0.9% and 3.5% on **ImageCLEF-DA** and **Amazon Revoew** respectively. Note that the reason for the minor boost for **ImageCLEF-DA** datasets is that its distribution discrepancy for **ImageCLEF-DA** datasets is small.

In conclusion, the performance of algorithms TCA, JDA, JGSA, DANN and DAN are generally worse than that of MFC. These results verify that, although those compared algorithms could reduce the marginal distribution gap between domains in the latent space, the difference between conditional distributions of two domains is increased. This may create a classifier learned from source samples that cannot be used to predict target samples. Note that, although JDA and JGSA try to match the conditional distribution, they still suffer the same issue of domain-invariant feature algorithms because the use of pseudo labels might misdirect this process. MFC could avoid this issue by sufficiently exploiting the geometric information between domains and applying the FCM technique instead of seeking an unknown space that contains much uncertainty.

3.3.6 Parameter Analysis

We analyze the sensitivity of two important parameters in MFC, λ and p, in this section. Our first step is to generate experiments designed to validate the premise that a wide range of parameter values can be chosen to obtain satisfactory performance on different types of datasets. Next, the variants of MFC are evaluated by preforming an ablation study. Our final step is to investigate how the level of fuzziness is influenced by parameter λ.

Parameter Sensitivity. The values of λ and p vary across a large range, and four experiments, **A** $\rightarrow$ **D** (**Office+Caltech10**), **A** $\rightarrow$ **D** (**Office-31**), **P** $\rightarrow$ **C** (**ImageCLEF-DA**) and **B** $\rightarrow$ **E** (**Amazon Review**), are used for illustration in Figs. 3.4 and 3.5. The solid line is the accuracy of MFC using different parameters, and the dashed line indicates the results

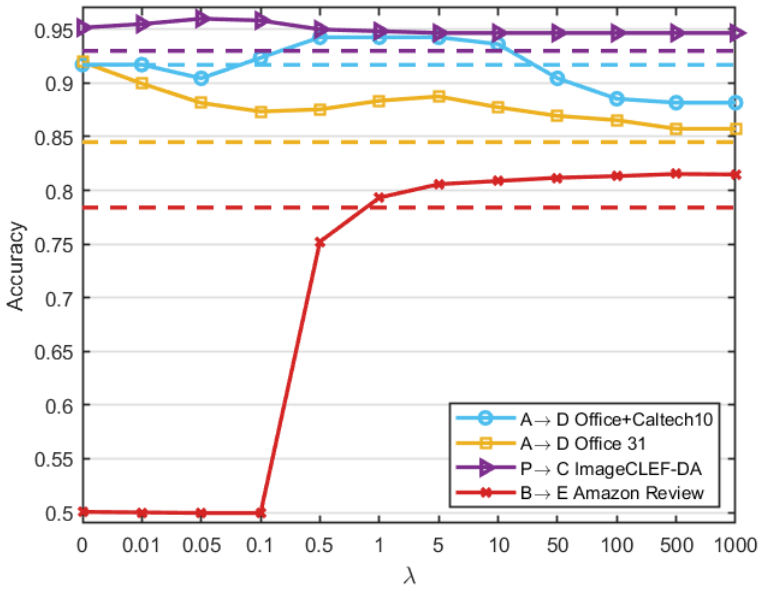


Figure 3.4: Parameter λ study of MFC

obtained by the best baseline algorithm on each dataset.

λ is the trade-off parameter to evaluate the importance of the manifold term and FCM. If $\lambda \to +\infty$, the fuzzy structure between target samples and the source centers is learned. If $\lambda \to 0$, the geometric information among target samples is extracted. Fig. 3.4 shows that λ can be selected from $[1, 10]$.

p is the other critical parameter, representing the number of Nearest Neighbors for calculating the affinity matrix. We run MFC with varying values of p. If $p \to +\infty$ and two samples which are not at all similar are connected. If $p \to 0$, limited similarity information between samples is captured. For optimum working, p should not be too large or too small. Fig. 3.5 shows that p can be selected from $[2, 32]$.

Ablation Study. We go deeper into the efficacy of our algorithm by performing an ablation study that evaluates variants of MFC. $\lambda = 0$ and $\lambda = +\infty$ are two extreme cases, indicating that MFC only considers the manifold term and MFC degenerates a special case of FCM respectively. We conduct experiments on datasets **Office+Caltech10** and **Amazon Review** with various values of $\lambda \in \{0, 10, +\infty\}$, and results are shown in Table 3.8.

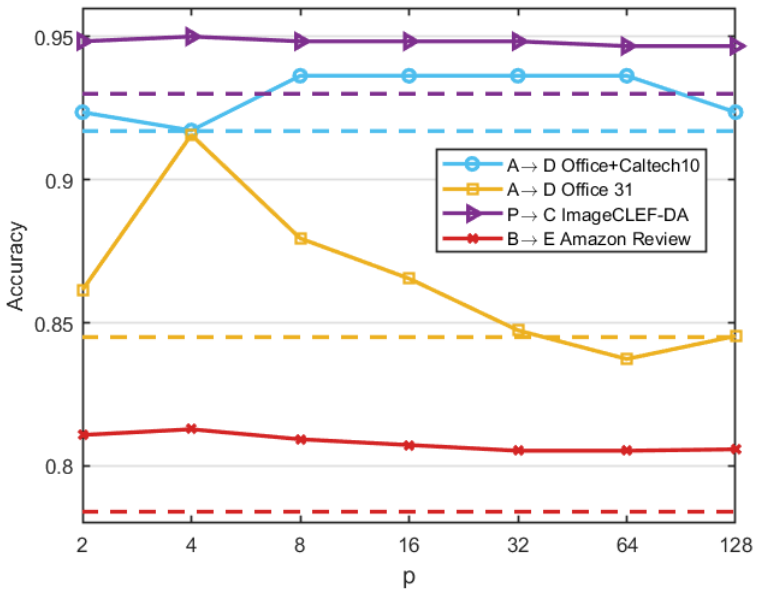


Figure 3.5: Parameter p study of MFC

For dataset **Office+Caltech10**, experiments show that the performance of $\lambda = 0$ is slightly better (0.3%) than the performance of $\lambda = 10$. Plus, the performance of $\lambda = 0$ or 10 is much better than the performance of $\lambda = +\infty$ (4.3%. 4.0%). However, for dataset **Amazon Review**, the performance of $\lambda = 0$ is very poor (52.3%), while $\lambda = +\infty$ and $\lambda = 10$ show solid and similar achievements (79.8%). Integrating the results mentioned above, we obtain the following outcomes:

a) For different datasets, the variation between the performances of $\lambda = 0$ and $\lambda = +\infty$ may be large. That is determined by the structure of datasets. If the target domain has a manifold structure, the performance of $\lambda = 0$ may be better than that of $\lambda = +\infty$. If the cluster centers of the target domain is close to those of the source domain, the performance of $\lambda = +\infty$ may be better than that of $\lambda = 0$.

b) Our experiments also show that when $\lambda = 10$, we can obtain a good performance. Combining with parameter sensitivity, λ can be selected from $[1, 10]$.

In general, the manifold term and fuzzy term are both important and necessary for our algorithm MFC.

Fuzzifier. To investigate the influence of parameter λ to the level

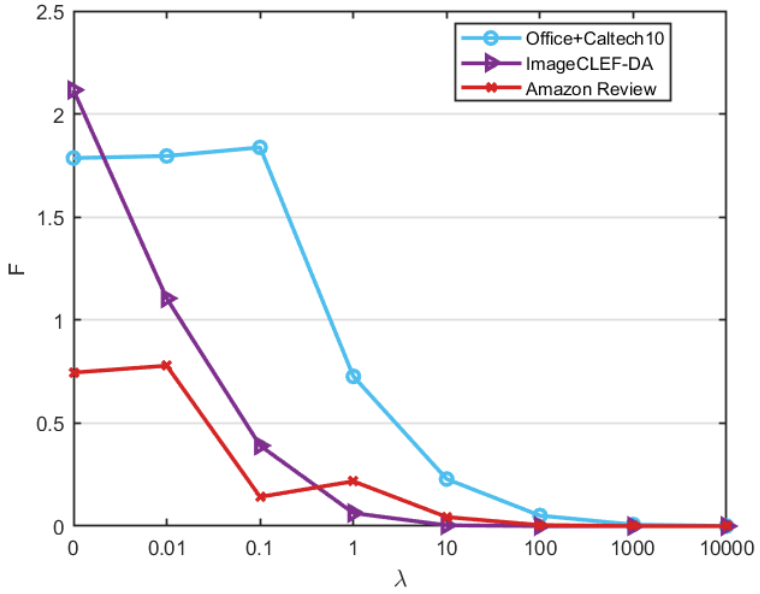


Figure 3.6: Analysis for the level of fuzziness.

of fuzziness, we analyze empirically the level of fuzziness by varying the value of λ based on the following index: $F = \frac{c}{n}[\sum_{j=1}^{n}\sum_{i=1}^{c}(u_{ij} - 1)^2\mathbb{1}_{u_{ij}>0.5} + \sum_{j=1}^{n}\sum_{i=1}^{c}(u_{ij})^2\mathbb{1}_{u_{ij}\leq 0.5}]$.

Remark 3. *For the obtained membership vector for each samples, we compare the greatest membership value with 1 and others with 0. F is the average of the comparison of all data. The smaller the value of F is, the harder the fuzziness becomes. When $F = 0$, the membership value $u_{ij} = 0$ or 1 and vice versa.*

We conduct the experiments on datasets **Office+Caltech10**, **ImageCLEF-DA** and **Amazon Review**, and report F for different λ, see Fig. 3.6, from which we observe that λ deeply influences the level of fuzziness. In general, as λ increases, the value of F decreases. When λ is larger than 1000, F becomes 0, which implies that MFC becomes a hard model. We also note that when the $\lambda = 0$, the value of F reaches its largest value. This implies that the manifold regularization is the crucial term to provide fuzzy information.

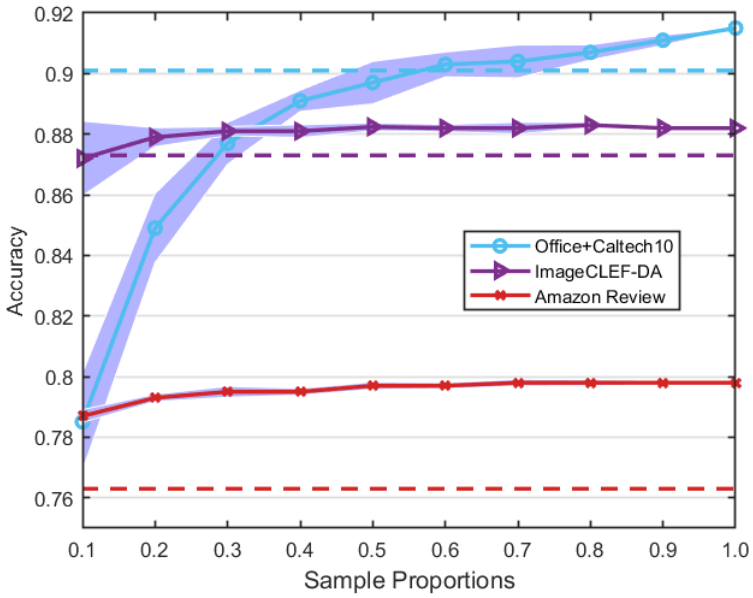


Figure 3.7: Accuracy and standard deviation on different data proportions.

3.3.7 Insufficient Source Samples

There is another advantage in our MFC algorithm in that only source centers are used, encouraging us to propose that MFC may provide a solution in the case of insufficient source samples. In transfer learning, there is an assumption that the source domain always has enough labeled samples. Existing transfer learning algorithms rely heavily on this assumption, especially the domain-invariant feature algorithms which need to train discriminate source classifiers. However, in real-world applications, it is not easy to find a suitable and label sufficient source domain for a given target domain. Again, our algorithm provides a promising solution to such source label insufficiency cases, and the application of transfer learning will be greatly expanded.

To verify the robustness of our algorithm MFC with a different number of source samples, we conduct experiments on three different **Amazon Review**, **Office+Caltech10** and **ImageCLEF-DA** datasets with varying numbers of labeled source samples. Labeled source samples are selected randomly according to sample proportions varing from 0.1 to 0.9.

To show that MFC is robust to a change in the number of the labeled source samples, we use the same parameters for all proportion values. Given the limited space here, we only report the average accuracy and standard deviation in Fig. 3.7. The dash line is the performance of the best baseline algorithm with all source samples.

On datasets **Office+Caltech10**, when the data proposition is from $[0.1, 0.5]$, the performance is much worse than the baseline. This is because the source data scale for the datasets is very small. When we randomly select these based on a small data proposition, it is possible that the selected source samples cannot cover all classes, which leads to the failure of MFC to recognize samples from missing classes. In addition, we note that the performance dramatically increases and draws level with the baseline as the data proposition increases. When the data proposition is larger than 0.5, the performance is better than the baseline.

On datasets **ImageCLEF-DA**, when the data proposition is 0.1, the mean performance is slightly poorer than the baseline with a large standard deviation. As we increase the data proposition, the performance dramatically improves and the standard deviation dramatically decreases. When the data proposition is larger than 0.2, MFC performs steadily with small standard deviations and achieves better outcomes than the baseline algorithm. The performance of MFC on datasets **Amazon Review** is stable, especially when the data proposition is larger than 0.3, and the standard deviations are very small, corresponding to all data propositions.

In general, MFC on **ImageCLEF-DA** and **Amazon Review** can achieve a robust performance even when the data propositions are smaller than 0.5. MFC on datasets **Office+Caltech10** can achieve good results with small standard deviations when the data proposition is larger than 0.5. Therefore, we conclude that our algorithm MFC may achieve satisfactory performance compared with the best baseline algorithm which uses all source samples, even when the data proposition is 0.5.

3.4 Summary

This work develops a Manifold Fuzzy Clustering algorithm, which integrates manifold structure to fuzzy clustering and transfers the class-level information between source and target domains. The manifold learning could explore the intrinsic geometric structure of domains, which supplements the FCM algorithm by providing the ability to discover relations among samples. Additionally, high level information of domains is transferred instead of specific data level information, such as distribution gap minimization. The proposed algorithm is evaluated and compared with some competitive unsupervised homogeneous domain adaptation algorithms on 48 real-world UHoDA tasks and a synthetic dataset. The experimental results demonstrate that our algorithm outperforms others in most of these datasets. To strengthen our work, experiments are designed to analyze the parameters in model and verify our algorithm's robustness.

Table 3.3: Accuracy (%) on Office+Caltech10 datasets using DeCAF6 features.

Dataset	C→A	C→W	C→D	A→C	A→W	A→D	D→A	D→C	D→W	W→C	W→A	W→D	Avg
1NN	87.1	72.2	80.9	78.5	77.3	80.3	75.9	70.1	98.0	68.2	73.1	**100.0**	80.1
TCA	88.7	77.3	86.6	76.9	68.1	73.9	84.9	72.9	98.6	71.8	76.3	**100.0**	81.3
GFK	82.3	75.9	83.4	80.3	77.0	80.9	75.8	69.1	98.6	67.8	74.3	**100.0**	80.9
CORAL	92.0	80.0	84.7	83.2	74.6	84.1	85.5	76.8	98.4	75.5	81.2	**100.0**	84.7
JDA	89.7	83.7	86.6	82.2	78.6	80.2	91.7	80.1	98.9	80.5	85.1	**100.0**	86.7
JGSA	91.4	86.8	**93.6**	84.9	81.0	88.5	92.0	86.2	**99.7**	85.0	90.7	**100.0**	90.0
AlexNet	91.9	83.7	87.1	83.0	79.5	87.4	87.1	79.0	97.7	73.0	83.8	**100.0**	86.1
DAN	92.0	**90.6**	89.3	84.1	**91.8**	91.7	90.0	80.3	98.5	81.2	92.1	**100.0**	90.1
MFC	**92.4**	89.8	92.4	**87.8**	91.5	**92.4**	**92.6**	**88.2**	96.6	**87.8**	**92.2**	94.9	**91.5**

Table 3.4: Accuracy (%) on Office-31 datasets using ResNet50 features.

Dataset	A→D	A→W	D→A	D→W	W→A	W→D	Avg
1NN	79.9	77.0	63.2	97.0	63.2	99.4	79.9
TCA	78.9	73.1	66.8	97.2	65.4	99.2	80.1
GFK	78.7	77.6	64.1	96.5	63.8	**99.8**	80.1
CORAL	80.1	77.2	63.3	96.9	63.3	99.4	80.0
JDA	84.5	81.4	68.9	97.6	70.7	99.2	83.7
JGSA	81.3	**82.1**	72.0	**97.7**	71.8	**99.8**	84.1
ResNet50	68.9	68.4	62.5	96.7	60.7	99.3	76.1
DANN	79.7	82.0	68.2	96.9	67.4	99.1	82.2
DAN	78.6	80.5	63.6	97.1	62.8	99.6	80.4
MFC	**88.4**	81.9	**72.9**	93.5	**73.2**	95.8	**84.3**

Table 3.5: Accuracy (%) on Office-Home datasets using ResNet50 features.

Dataset	Ar→Cl	Ar→Pr	Ar→Rw	Cl→Ar	Cl→Pr	Cl→Rw	Pr→Ar	Pr→Cl	Pr→Rw	Rw→Ar	Rw→Pr	Rw→Cl	Avg
1NN	42.4	59.5	65.8	45.1	56.3	58.7	48.7	40.9	69.6	60.1	73.4	47.2	55.6
TCA	42.4	62.6	66.4	41.7	56.6	59.1	44.2	39.5	68.0	55.9	73.4	44.3	54.5
GFK	42.5	62.1	66.2	44.9	58.0	59.8	47.1	40.2	69.4	58.8	73.3	46.6	55.7
CORAL	42.2	59.1	64.9	46.4	56.3	58.3	45.4	41.2	68.5	60.1	73.1	48.2	55.3
JDA	46.1	64.7	68.5	45.0	60.8	62.2	48.4	42.4	69.0	57.6	76.2	49.4	57.5
JGSA	46.4	69.9	73.5	46.1	67.0	66.7	49.2	44.4	69.1	56.7	76.2	48.3	59.5
ResNet50	34.9	50.0	58.0	37.4	41.9	46.2	38.5	31.2	60.4	53.9	59.9	41.2	46.1
DANN	45.6	59.3	70.1	47.0	58.5	60.9	46.1	43.7	68.5	**63.2**	76.8	**51.8**	57.6
DAN	43.6	57.0	67.9	45.8	56.5	60.4	44.0	43.6	67.7	53.1	74.3	51.5	56.3
MFC	**50.5**	**73.1**	**76.2**	**56.0**	**67.5**	**67.9**	**55.6**	**46.8**	**77.4**	63.1	**77.7**	50.2	**63.5**

Table 3.6: Accuracy (%) on ImageCLEF-DA using ResNet50 features.

Dataset	I→P	P→I	I→C	C→I	C→P	P→C	Avg
1NN	72.8	71.2	90.7	85.2	71.3	75.8	77.8
TCA	76.3	78.8	91.5	84.8	71.2	85.2	81.3
GFK	75.7	75.0	92.3	86.2	73.0	81.7	80.6
CORAL	74.0	71.3	91.2	80.7	67.0	73.0	76.2
JDA	77.0	77.5	93.7	**91.5**	74.8	85.8	83.4
JGSA	**79.0**	88.8	94.8	91.3	**76.5**	93.0	87.3
ResNet50	74.8	83.9	91.5	78.0	65.5	91.2	80.7
DANN	75.0	86.0	96.2	87.0	74.3	91.5	85.0
DAN	74.5	82.2	92.8	86.3	69.2	89.8	82.5
MFC	78.3	**92.3**	**96.3**	91.3	76.3	**94.8**	**88.2**

Table 3.7: Accuracy (%) on Amazon Review datasets.

Dataset	B→E	B→K	B→D	E→B	E→K	E→D	K→B	K→E	K→D	D→B	D→K	D→E	Avg
1NN	58.2	60.8	60.6	59.2	63.9	57.9	59.5	64.3	60.1	60.4	57.5	57.6	60.0
TCA	64.1	66.4	65.4	63.7	71.0	65.3	66.9	70.4	65.5	67.4	66.1	64.8	66.4
GFK	63.8	66.6	68.4	63.7	70.6	63.8	65.1	70.2	64.1	65.1	65.7	62.8	65.8
CORAL	58.7	60.8	60.3	59.4	64.8	58.7	60.5	64.8	59.8	60.1	56.7	57.3	60.1
JDA	64.7	67.5	65.3	63.2	69.8	64.3	66.1	72.9	62.5	68.0	68.5	67.0	66.7
JGSA	72.4	73.4	72.3	71.0	75.1	69.4	68.7	75.1	65.5	69.8	70.9	70.2	71.2
DANN	78.4	73.3	77.9	72.3	75.4	**78.3**	71.3	73.8	**85.4**	70.9	74.0	**84.3**	76.3
MFC	**80.9**	**82.8**	**80.2**	**74.6**	**84.2**	76.3	**76.0**	**81.7**	77.6	**78.3**	**83.7**	81.6	**79.8**

Table 3.8: Accuracy (%) on Office+Caltech10 and Amazon Review datasets with $\lambda = 0, 10, +\infty$.

Dataset	C→A	C→W	C→D	A→C	A→W	A→D	D→A	D→C	D→W	W→C	W→A	W→D	Avg
$\lambda = 10$	**92.4**	89.8	**92.4**	**87.8**	91.5	**92.4**	**92.6**	88.2	96.6	87.8	92.2	**94.9**	91.5
$\lambda = 0$	**92.4**	**94.2**	91.1	87.6	**91.9**	91.7	92.5	**88.4**	**97.0**	**88.2**	**92.4**	94.3	**91.8**
$\lambda = +\infty$	91.5	84.4	84.1	84.7	86.1	88.5	90.7	78.7	96.5	77.7	88.7	**98.7**	87.5
Dataset	B→E	B→K	B→D	E→B	E→K	E→D	K→B	K→E	K→D	D→B	D→K	D→E	Avg
$\lambda = 10$	80.9	**82.8**	**80.2**	74.6	**84.2**	76.3	76.0	81.7	**77.6**	78.3	**83.7**	81.6	**79.8**
$\lambda = 0$	50.1	69.1	50.8	50.0	50.0	50.0	56.1	50.8	50.8	50.0	50.0	50.1	52.3
$\lambda = +\infty$	**81.4**	82.1	80.0	**75.0**	83.3	**76.8**	**76.4**	**82.0**	76.9	**79.0**	82.9	**81.9**	**79.8**

Chapter 4

Source Domain Reconstruction for Unsupervised Homogeneous Domain Adaptation

This chapter discusses how to construct a new source domain using the original source domain and target domain to achieve better transfer performance.

4.1 Introduction

The successes of UHoDA derive in large part from the availability of the high-quality source domain [132–134]. Hence, it is crucial to select a suitable source domain for a given target domain. To solve the source domain selection problem, two important settings have been considered. The first setting is called Source Domain Selection (SDS) [132, 135, 136], which aims to select the most suitable source domain from given multiple candidate source domains. The second setting is called Multi-source Domain Adaptation (MsDA) [137–142], which aims to utilize the given multiple candidate source domains to improve transfer performance instead of recognizing the most suitable source domain.

No matter the SDS problem or the MsDA problem, they both have an implicit assumption that the multiple transferable source domains are available. However, in real world application, collecting multiple transferable source domains are very expensive and prohibitive, especially when the budget is limited [143, 144]. Even if we have collected multiple source domains, the computing complexity and computing time of SDS or

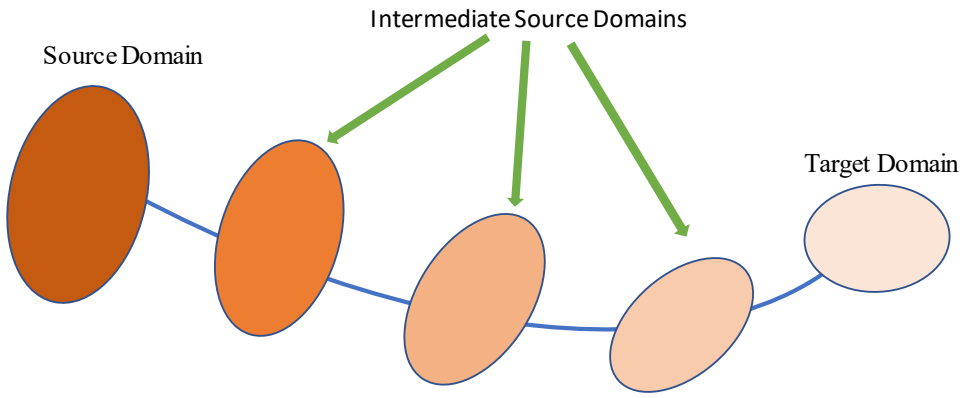


Figure 4.1: To connect source domain and target domain, we construct a source domain sequence, where domains varies continuously. Then, we select an intermediate source domain as the reconstructed source domain.

MsDA algorithm may be very high, because we need to process massive labeled samples from multiple domains. This circumstance may hinder the promotion of SDS, MsDA and UHoDA to more areas.

To mitigate the problems mentioned above, we propose a novel problem setting: Source Domain Reconstruction (SDR), which aims to construct the pseudo source domain by the existing source domain and target domain. Compared to the original source domain, we hope that the generated source domain i) has a stronger transfer ability corresponding to the given target domain, ii) can replace the original source domain to build more accurate target classifier and iii) can also be used as the candidate source domain in the SDS and MsDA problems.

To explore the feasibility of SDR, we study it from a theoretical perspective. Based on the discovery that SDR algorithm is a functional of UHoDA algorithms, we establish a strictly mathematical definition for SDR. Further, we discuss a problem that whether there exists a SDR algorithm can work for all effective UHoDA algorithms. To answer this problem, we prove a corresponding "No Free Lunch Theorem" for SDR [57], which provides a negative answer. This implies that we may only construct SDR algorithm for some special UHoDA algorithm spaces.

To further study those special UHoDA algorithm spaces, we develop a novel concept named as unsaturated algorithm space. Based on the novel concept, we prove an interesting and important theorem by Axiom of Choice, i.e., the necessary and sufficient condition for the existence of a SDR algorithm corresponding to an algorithm space is that the algorithm space is unsaturated. With this result, we establish a solid theoretical guarantee for SDR, providing a positive answer for the feasibility of SDR.

Following our theoretical analysis, we design a novel SDR algorithm for a specific UHoDA algorithm space consisting of nine famous UHoDA algorithms. The presented SDR algorithm is referred to as *Domain MixUp* (DMU), which aims to construct a source domain sequence (see Fig. 4.1) with index varying from 0 to 1. When the index is 0, the domain is the original source domain. When the index is 1, the domain mainly contains the information related to the target domain. Lastly, we select an intermediate domain as the reconstructed source domain. Note that we describe our algorithm using the word "MixUp", since DMU mainly utilizes the basic idea from MixUp technique [145].

To validate the efficacy of the DMU algorithm over selected UHoDA algorithm space, we conduct extensive experiments on several standard benchmark datasets containing 66 transfer tasks. Extensive experiments have shown that DMU does construct a new source domain with stronger transfer ability. Furthermore, due to the nature of SDR algorithm, we can use reconstructed source domain with the target domain and selected UHoDA algorithm to train a more accurate target classifier. Hence, combining with DMU algorithm, we have strengthened nine UHoDA algorithms from the selected algorithm space.

The contributions of this chapter are summarized as follows.

• We propose a new UHoDA problem: Source Domain Reconstruction, which can not only construct new source domain, but also enhance the performance for existing UHoDA algorithms by the reconstructed source domain.

• We have provided a strictly mathematical definition for source do-

main reconstruction. The feasibility analysis for source domain reconstruction has also been studied from the theoretical perspective.

• We develop a source domain reconstruction algorithm, Domain MixUp (DMU). The algorithm is simple, which can be implemented easily and enables us to construct a new source domain with stronger transfer ability corresponding to a give target domain.

• We have conducted 66 transfer learning tasks for evaluating DMU. Extensive experiments demonstrate that DMU does solve the source domain reconstruction problem for selected UHoDA algorithms.

4.2 Problem Setting and Theoretical Analysis

In this section, we elaborate the source domain reconstruction problem, then we discuss the problem from a theoretically perspective.

4.2.1 Problem Setting

The main aim of unsupervised homogeneous domain adaptation is to build an algorithm which generates target classifier by labeled source samples and unlabeled target samples:

Definition 11 (Unsupervised Homogeneous Domain Adaptation (UHoDA) Algorithm)**.** *Given any labeled source samples* $\mathcal{S} = \{(\mathbf{x}_s^i, y_s^i)\}_{i=1}^{n_s}$ *and unlabeled target samples* $\mathcal{T}_u = \{\mathbf{x}_t^i\}_{i=1}^{n_t}$ *defined in Definition 1, an unsupervised domain adaptation algorithm is*

$$\begin{aligned} A : \{\mathcal{X} \times \mathcal{Y}\}^{n_s} \times \mathcal{X}^{n_t} &\to \mathcal{Y}^{\mathcal{X}} \\ \mathcal{S} \times \mathcal{T}_u \ &\to \mathcal{A}(\mathcal{S}, \mathcal{T}_u). \end{aligned} \tag{4.2.1}$$

We define $\mathcal{A}$ *as the unsupervised domain adaptation algorithm space, which is a subspace of the space containing all unsupervised domain adaptation algorithms.*

Note that in this chapter, we use y to present the label, but not one-hot vector $\mathbf{y}$. The effectiveness of UHoDA heavily depends on the quality of source domain. Instead of extracting knowledge from multiple sources, which is the key technique for multi-source domain adaptation and domain selection, we define a source domain reconstruction algorithm, which could effectively utilize the current single source domain to enhance transferring performance.

Generally, source domain reconstruction algorithm A' is a functional on source samples $\mathcal{S}$ and target samples $\mathcal{T}_u$: $A'(\mathcal{S}, \mathcal{T}_u) \in \{\mathcal{X} \times \mathcal{Y}\}^{n'_s}$, where n'_s is the size of constructed source samples. The output of reconstruction algorithm is a reconstructed source domain. A crucial problem is how to estimate the transfer ability for reconstructed source domain. A feasible method is to utilize UHoDA algorithm A: we say the new source domain $A'(\mathcal{S}, \mathcal{T}_u)$ has a stronger transfer ability than $\mathcal{S}$ corresponding to a given target domain $\mathcal{T}_u$, if

$$\epsilon_t(A(A'(\mathcal{S}, \mathcal{T}_u), \mathcal{T}_u)) < \epsilon_t(A(\mathcal{S}, \mathcal{T}_u)), \tag{4.2.2}$$

where ϵ_t is the target error.

However, this method has a serious defect that one can easily construct different UHoDA algorithms A_1 and A_2 such that

$$\begin{aligned} &\epsilon_t(A_1(A'(\mathcal{S}, \mathcal{T}_u), \mathcal{T}_u)) < \epsilon_t(A_1(\mathcal{S}, \mathcal{T}_u)), \\ &\epsilon_t(A_2(A'(\mathcal{S}, \mathcal{T}_u), \mathcal{T}_u)) > \epsilon_t(A_2(\mathcal{S}, \mathcal{T}_u)). \end{aligned} \tag{4.2.3}$$

From the perspective of A_1, new source domain has a stronger transfer ability, but original source samples $\mathcal{S}$ is better from the view of A_2. Hence, it is hard to determine which source samples is better.

To overcome the problem, we also require the source domain reconstruction algorithm A' is a functional on UHoDA algorithms: $A'(\mathcal{S}, \mathcal{T}_u, A) \in \{\mathcal{X} \times \mathcal{Y}\}^{n'_s}$. In this case, the conflict in (4.2.3) can be mitigated. That is because that $A'(\mathcal{S}, \mathcal{T}_u, A_1)$ may be different with $A'(\mathcal{S}, \mathcal{T}_u, A_2)$, if $A_1 \neq A_2$.

Utilizing above discussion, we propose the definition of source domain reconstruction algorithm as follows:

Definition 12 (Source Domain Reconstruction (SDR)). *Given labeled source samples* $\mathcal{S} = \{(\mathbf{x}_s^i, y_s^i)\}_{i=1}^{n_s}$ *and unlabeled target samples* $\mathcal{T}_u = \{\mathbf{x}_t^i\}_{i=1}^{n_t}$ *and any unsupervised domain adaptation algorithm* $A \in \mathcal{A}$*, an algorithm called source domain reconstruction algorithm* A' *is a functional* $A'(\mathcal{S}, \mathcal{T}_u, A) \in \{\mathcal{X} \times \mathcal{Y}\}^{n_s'}$*, such that*

$$\epsilon_t(A(A'(\mathcal{S}, \mathcal{T}_u, A), \mathcal{T}_u)) < \epsilon_t(A(\mathcal{S}, \mathcal{T}_u)),$$

and

$$A'(\emptyset, \mathcal{T}_u, A) = \emptyset,$$

where ϵ_t *is the target error and* n_s' *is the size of constructed source domain.*

4.2.2 Proposed Theoretical Problems

Here, to study the feasibility of the SDR, we list three main problems.

Problem 1. *For all unsupervised domain adaptation algorithms* $A \in \mathcal{A}$*, is there a source domain reconstruction algorithm* A' *?*

Problem 2. *Given an unsupervised domain adaptation algorithm* A*, if* A *has a corresponding source domain reconstruction algorithm* A'*, consider the following unsupervised domain adaptation algorithm sequence* $\{A_i\}_{i=0}^{+\infty}$*:*

$$A_i(\mathcal{S}, \mathcal{T}_u) = A_{i-1}(A'(\mathcal{S}, \mathcal{T}_u, A_{i-1}), \mathcal{T}_u) \text{ and } A_0 = A.$$

is it possible, for any $i < j$*,* $i, j \in \mathbb{Z}_+$ *such that*

$$\epsilon_t(A_i(\mathcal{S}, \mathcal{T}_u)) < \epsilon_t(A_j(\mathcal{S}, \mathcal{T}_u)) \text{ ?}$$

Problem 3. *What are the necessary and sufficient conditions that the*

unsupervised domain adaptation algorithm space $\mathcal{A}$ has a corresponding source domain reconstruction algorithm A'?

The first and third problems mainly argue the existence of source domain reconstruction algorithm for a given algorithm space $\mathcal{A}$. The second problem mainly discusses the limitation of the source domain reconstruction algorithm. If the problem 2 can be proven, one may achieve a better performance by iterating continuously.

It is clear that the problem 1 is impossible to be achieved. One can construct the unsupervised cluster algorithm A, which satisfies $A(\mathcal{S}', \mathcal{T}_u) = A(\emptyset, \mathcal{T}_u)$, for any source domain $\mathcal{S}'$. In this scenario, we have

$$\epsilon_t(A(A'(\mathcal{S}, \mathcal{T}_u, A), \mathcal{T}_u)) = \epsilon_t(A(\mathcal{S}, \mathcal{T}_u)) = \epsilon_t(A(\emptyset, \mathcal{T}_u)).$$

Above example tells us that the algorithm space $\mathcal{A}$ should be limited in a suitable range. We mainly focus on those algorithms satisfying the negative transfer condition [146]:

$$\epsilon_t(A(\mathcal{S}, \mathcal{T}_u)) > \epsilon_t(A(\emptyset, \mathcal{T}_u)). \tag{4.2.4}$$

Hence, we revise the problem 1 as follows:

Assume $\mathcal{A}$ collects all unsupervised domain adaptation algorithms satisfying the negative transfer condition (4.2.4), is there a source domain reconstruction algorithm A' ?

4.2.3 Theoretical Analysis for SDR

By constructing a counterexample, Theorem 2 provides a negative answer for revised problem 1:

Theorem 2 (No Free Lunch Theorem for SDR)**.** *Given a source domain $\mathcal{S}$, a target domain $\mathcal{T}_u$ and the algorithm space $\mathcal{A}$ containing all unsupervised domain adaptation algorithms satisfying the negative transfer condition (4.2.4), there is no source domain reconstruction algorithm*

A' such that

$$\epsilon_t(A(A'(\mathcal{S}, \mathcal{T}_u, A), \mathcal{T}_u)) < \epsilon_t(A(\mathcal{S}, \mathcal{T}_u)),$$

for all $A \in \mathcal{A}$.

Proof. We assume there is a source domain reconstruction algorithm A' such that

$$\epsilon_t(A(A'(\mathcal{S}, \mathcal{T}_u, A), \mathcal{T}_u)) < \epsilon_t(A(\mathcal{S}, \mathcal{T}_u)),$$

for all $A \in \mathcal{A}$.

We need to select a special algorithm A from $\mathcal{A}$. We set the output of A belonging to a finite hypothesis space $\mathcal{H}$ [57] (one can construct such algorithm A easily). Using A, we construct unsupervised domain adaptation algorithm sequence $\{A_i\}_{i=0}^{+\infty}$ as follows:

$$A_i(\mathcal{S}, \mathcal{T}_u) = A_{i-1}(A'(\mathcal{S}, \mathcal{T}_u, A_{i-1}), \mathcal{T}_u) \tag{4.2.5}$$

and $A_0 = A$. Hence,

$$\epsilon_t(A_i(\mathcal{S}, \mathcal{T}_u)) < \epsilon_t(A_j(\mathcal{S}, \mathcal{T}_u)), \quad \text{any } j < i. \tag{4.2.6}$$

It is clear that $A_i \in \mathcal{A}$ and the output of A_i belonging to $\mathcal{H}$. Note that the space $\mathcal{H}$ is finite, thus, there exists k_1, k_2 satisfying $k_1 \neq k_2, |k_1 - k_2| \leq |\mathcal{H}|$ such that

$$A_{k_1}(\mathcal{S}, \mathcal{T}_u) = A_{k_2}(\mathcal{S}, \mathcal{T}_u), \tag{4.2.7}$$

implying that

$$\epsilon_t(A_{k_1}(\mathcal{S}, \mathcal{T}_u)) = \epsilon_t(A_{k_2}(\mathcal{S}, \mathcal{T}_u)),$$

which is conflict with Eq. (4.2.6). □

This theorem states that there does not exist a SDR algorithm A' for all unsupervised domain adaptation algorithms satisfying the negative transfer condition (4.2.6). Note that the proof process of Theorem 2 implies the answer of problem 2. In Theorem 2, we have constructed unsupervised domain adaptation algorithm sequence $\{A_i\}_{i=0}^{+\infty}$ and shown

that there exist different indexes k_1, k_2 such that

$$\epsilon_t(A_{k_1}(\mathcal{S}, \mathcal{T}_u)) = \epsilon_t(A_{k_2}(\mathcal{S}, \mathcal{T}_u)). \tag{4.2.8}$$

Though a counterexample has been constructed to deny the problem 2, we are still curious whether there exists example satisfying the requirement of problem 2? Following example provides us a scenarios.

Example 1. *We set the source marginal distribution be the uniform distribution in interval* $(0.5, 1.5)$ *and the label is* 1 *if samples is in* $(0.5, 1.0)$*, the the label is* 0 *if sample is in* $[1.0, 1.5)$*. We set the target marginal distribution be the uniform distribution in* $(0, 2.0)$ *and the label is* 1 *if sample is in* $(0, 1.0)$*, the the label is* 0 *if samples is in* $[1.0, 2.0)$*. We also define the hypothesis space* $\mathcal{H}$ *is* $\{1\} \cup \{h_{\mathbf{x}} : \mathbf{x} \in (0.5, 1.5)\}$*, where* $h_{\mathbf{x}}(\mathbf{z}) = 1$ if $\mathbf{z} < \mathbf{x}$; otherwise $h_{\mathbf{x}}(\mathbf{z}) = 0$*. Hence we define the unsupervised domain adaptation algorithm* A *as follows:*

$$A(\mathcal{S}, \mathcal{T}_u) = h_{\mathbf{x}} \in \mathcal{H}, \tag{4.2.9}$$

where $\mathbf{x} = \min_{\{(\mathbf{z},y)\in\mathcal{S}\}} \mathbf{z}$*. And* $A(\emptyset, \mathcal{T}_u) = 1$*.*
We construct the source domain reconstruction algorithm $A'(\mathcal{S}, \mathcal{T}_u, A)$ *as follows:*

$$\{(\mathbf{x}, y) : \mathbf{x} = \mathbf{z} + 0.5(1.0 - \mathbf{z})\mathbf{1}_{\mathbf{z}<1.0}, \text{ for } (\mathbf{z}, y) \in \mathcal{S}\}, \tag{4.2.10}$$

and $A'(\emptyset, \mathcal{T}_u, A) = \emptyset$*, where* $\mathbf{1}_{\mathbf{z}<1.0}$ *is the indicator function over* $\{\mathbf{z} < 1.0\}$*. We can easily prove that for any* $i < j$*,* $i, j \in \mathbb{Z}_+$ *such that*

$$\epsilon_t(A_i(\mathcal{S}, \mathcal{T}_u)) < \epsilon_t(A_j(\mathcal{S}, \mathcal{T}_u)),$$

and

$$\lim_{i\to+\infty} \epsilon_t(A_i(\mathcal{S}, \mathcal{T}_u)) = 0.$$

To address problem 3, we define several novel concepts.

Definition 13. *An algorithm A is unsaturated for samples $\mathcal{S}$ and $\mathcal{T}_u$, if there exists samples $\mathcal{S}'$ such that $\epsilon_t(A(\mathcal{S}', \mathcal{T}_u)) < \epsilon_t(A(\mathcal{S}, \mathcal{T}_u))$.*

Definition 14. *An algorithm space $\mathcal{A}$ is unsaturated for samples $\mathcal{S}$ and $\mathcal{T}_u$, if for any $A \in \mathcal{A}$, A is unsaturated for samples $\mathcal{S}$ and $\mathcal{T}_u$.*

Then, the following theorem tells us that for unsaturated algorithm space $\mathcal{A}$, there exists a source domain reconstruction algorithm.

Theorem 3. *Given an algorithm space $\mathcal{A}$, there is a source domain reconstruction algorithm A' for $\mathcal{A}$ if and only if $\mathcal{A}$ is unsaturated.*

Proof. First, we prove that if $\mathcal{A}$ is unsaturated, then there is a source domain reconstruction algorithm A' for $\mathcal{A}$.

For any $A \in \mathcal{A}$,

$$S(A) = \{S' \ : \ \epsilon_t(A(\mathcal{S}', \mathcal{T}_u)) < \epsilon_t(A(\mathcal{S}, \mathcal{T}_u))\}.$$

Because $\mathcal{A}$ is unsaturated, it is clear that $S(A) \neq \emptyset$. By the Axiom of Choice, there exists a functional

$$f : \mathcal{A} \to \cup_{A \in \mathcal{A}} S(A)$$

such that $f(A) \in S(A)$. Then, we set

$$A'(\mathcal{S}, \mathcal{T}_u, A) = f(A).$$

Second, we proof that if there is a source domain reconstruction algorithm A' for $\mathcal{A}$, then $\mathcal{A}$ is unsaturated.

For any $A \in \mathcal{A}$, we set $S' = A'(\mathcal{S}, \mathcal{T}_u, A)$. Then, it is clear that $\epsilon_t(A(\mathcal{S}', \mathcal{T}_u)) < \epsilon_t(A(\mathcal{S}, \mathcal{T}_u))$. Hence, A is unsaturated, which implies that $\mathcal{A}$ is unsaturated. □

Theorem 3 provides a necessary and sufficient condition for the existence of a source domain reconstruction algorithm A' corresponding to an algorithm space $\mathcal{A}$. The theorem provides a positive answer for problem 3. Note that we only consider a pair $(\mathcal{S}, \mathcal{T}_u)$. Given an algorithm

space $\mathcal{A}$ and a set of samples $\{(\mathcal{S}^i, \mathcal{T}_u^i)\}_{i\in\mathcal{I}}$, where $\mathcal{I}$ is an index set, is there a domain reconstruction algorithm A' such that for any $A \in \mathcal{A}$ and any $(\mathcal{S}^i, \mathcal{T}_u^i)$, $i \in \mathcal{I}$,

$$\epsilon_t(A(A'(\mathcal{S}^i, \mathcal{T}_u^i, A), \mathcal{T}_u^i)) < \epsilon_t(A(\mathcal{S}^i, \mathcal{T}_u^i)) \ ?$$

To address above problem, we extend definitions 13 and 14 to a general version.

Definition 15. *An algorithm A is unsaturated for $\{(\mathcal{S}^i, \mathcal{T}_u^i)\}_{i\in\mathcal{I}}$, if for any $i \in \mathcal{I}$, there exists samples $\tilde{\mathcal{S}}^i$ such that $\epsilon_t(A(\tilde{\mathcal{S}}^i, \mathcal{T}_u^i)) < \epsilon_t(A(\mathcal{S}^i, \mathcal{T}_u^i))$.*

Definition 16. *An algorithm space $\mathcal{A}$ is unsaturated for $\{(\mathcal{S}^i, \mathcal{T}_u^i)\}_{i\in\mathcal{I}}$, if for any $A \in \mathcal{A}$, A is unsaturated for $\{(\mathcal{S}^i, \mathcal{T}_u^i)\}_{i\in\mathcal{I}}$.*

Theorem 4. *Given an algorithm space $\mathcal{A}$ and a set of samples $\{(\mathcal{S}^i, \mathcal{T}_u^i)\}_{i\in\mathcal{I}}$, there is a source domain reconstruction algorithm A' for $\mathcal{A}$ over $\{(\mathcal{S}^i, \mathcal{T}_u^i)\}_{i\in\mathcal{I}}$ if and only if $\mathcal{A}$ is unsaturated for $\{(\mathcal{S}^i, \mathcal{T}_u^i)\}_{i\in\mathcal{I}}$.*

Proof. The Axiom of Choice is the main lemma to complete the proof. We omit the proof because the proof of Theorem 4 is similar to the proof of Theorem 3. □

Summarizing the discussions about problem 1, problem 2 and problem 3, we have the following results:

1. Not any algorithm space $\mathcal{A}$ has a source domain reconstruction algorithm A';

2. It is impossible for all algorithms to satisfy the requirement of problem 2;

3. An example has shown that there exists algorithm A and corresponding algorithm A' satisfying the requirement of problem 2.

4. The necessary and sufficient condition for the existence of a SDR algorithm corresponding to an algorithm space $\mathcal{A}$ is that the algorithm space $\mathcal{A}$ is unsaturated.

4.2.4 Selected Algorithm Spaces

Here we mainly focus on the following problem: given famous unsupervised domain adaptation algorithms, is there a source domain reconstruction algorithm A'?

In this chapter, we only consider shallow unsupervised domain adaptation algorithms. Most of the existing UHoDA algorithms for classification tasks satisfy the negative transfer condition [146], and we select nine algorithms with more than 50 citation each to form our UHoDA algorithm space. The nine UHoDA algorithms are summarized as follows:

No Transfer

• 1NN. 1-nearest neighbor, which is implemented without transfer learning.

• PCA+1NN.

Subspace Transfer

• GFK [66], which connects two domains with geodesic curves on a Grassmann manifold.

• CORAL+1NN [68], which performs second-order subspace alignment.

Domain Invariant Feature

• TCA [5], which seeks linear subspace and uses MMD [62] to match distributions.

• JDA [65], which jointly mathches the marginal distributions and conditional distributions.

• JGSA [35], which not only considers the distribution discrepancy but also matches the geometric shift.

Geometric Transfer

• EasyTL [147], which is able to learn both non-parametric transfer features and classifiers by exploiting intra-domain structures.

Adaptation Based Transfer

• MEDA [96], which performs dynamic distribution alignment by considering the different importance of marginal and conditional distributions.

4.3 Proposed Algorithm

The basic motivation is from continuous method used in partial differential equation. Assume we have a source domain sequence $\{\mathcal{S}_\alpha\}_{\alpha\in[0,1]}$. When $\alpha = 0$, $\mathcal{S}_0 = \mathcal{S}$ and when $\alpha = 1$, $\mathcal{S}_1$ mainly contains the information related to the target samples $\mathcal{T}_u$. Then, we can use an intermediate domain $\mathcal{S}_{\alpha_0}$ as reconstructed source domain to train the target classifier.

There are many different strategies to construct the domain sequence. In our proposed method, a feasible strategy is adopted to extend the MixUp technique [145] to domain adaptation scenarios.

4.3.1 MixUp in Domain Adaptation

MixUp is proposed by Zhang et al. [145] and can be understood as a form of data augmentation encouraging the classifier h to behave linearly in-between training samples. Given samples $\{(\mathbf{x}^i, y^i)\}_{i=1}^n$, MixUp generates new samples as follows:

$$\begin{aligned} \tilde{\mathbf{x}} &= \alpha\mathbf{x}^i + (1-\alpha)\mathbf{x}^j, \\ \tilde{y} &= \alpha y^i + (1-\alpha)y^j, \end{aligned} \tag{4.3.1}$$

where $(\mathbf{x}^i, y^i)$ and $(\mathbf{x}^j, y^j)$ are two different samples drawn at random from training samples and $\alpha \in [0, 1]$.

Observing Eq. (4.3.1), two conditions are required:

1. all samples should have labels;
2. the implementing algorithm can use soft labels as input.

The two conditions are the main obstacles for us to use MixUp in unsupervised domain adaptation scenarios. Because 1) target samples have not any labels, thus, we cannot use the target samples to generate labeled samples by MixUp. However, generated source samples $\mathcal{S}_\alpha$ should contain target information. 2) Not all unsupervised domain adaptation can use the soft labels as input, such as JDA and JGSA.

To solve the first obstacle, target pseudo labels generated by UHoDA

Algorithm 2 Domain MixUp

Input: Data $\mathcal{S}, \mathcal{T}_u$; UHoDA algorithm A; parameters α_0, T
1. $t \leftarrow 0$ and $\mathcal{S}_0 \leftarrow \mathcal{S}$
While $t < T$ 2. Estimate classifier $h \leftarrow A(\mathcal{S}_{t\alpha_0/T}, \mathcal{T}_u)$;
3. Predict pseudo class centers $\{(\mathbf{c}_t^y, y)\}_{y=1}^K$
4. Construct $\mathcal{S}_{(t+1)\alpha_0/T}$ by $\{(\mathbf{c}_t^y, y)\}_{y=1}^K$, Eq. (4.3.2) with $\alpha = (t+1)\alpha_0/T$
5. $t \leftarrow t+1$
Output: Classifier $A(\mathcal{S}_{\alpha_0}, \mathcal{T}_u)$.

algorithm $A(\mathcal{S}, \mathcal{T}_u)$ may be an allowed strategy. However, when the pseudo labels are not accurate, there exist bias for the generated samples leading to a poor performance. According to [148], the pseudo class centers are more robust and may maintain stably, even if the pseudo labels are not accurate. Hence, we use the pseudo target class centers $\{(\mathbf{c}_t^y, y)\}_{y=1}^K$ and source samples $\{(\mathbf{x}_s^i, y_s^i)\}_{i=1}^{n_s}$ as the training samples in MixUp process.

To solve the second obstacle, we simply use the samples from same class to implement MixUp. We can construct source samples as follows:

$$\begin{aligned} \mathbf{x}_\alpha^i &= \alpha \mathbf{x}_s^i + (1-\alpha)\mathbf{c}_{y_s^i}^t, \\ y_\alpha^i &= \alpha y_s^i + (1-\alpha) y_s^i = y_s^i. \end{aligned} \tag{4.3.2}$$

Hence, n_s samples $\{(\mathbf{x}_\alpha^i, y_\alpha^i)\}_{i=1}^{n_s}$ are generated. Note that when $\alpha = 0$, the constructed source domain is $\mathcal{S}$ and when $\alpha = 1$, the constructed source domain is $\{(\mathbf{c}_t^y, y)\}_{y=1}^K$, which mainly contains the target information.

4.3.2 Intermediate Domains

We construct the source domain sequence based on the above Mixup technique. We select a parameter α_0 ($0 < \alpha_0 < 1$) to fix the intermediate domain $\mathcal{S}_{\alpha_0}$ used to train transfer classifier and set T as the number of intermediate domains to connect $\mathcal{S}_0$ and $\mathcal{S}_{\alpha_0}$.

Given the $t < T$, we assume that $\mathcal{S}_{t\alpha_0/T}$ as the generated source domain. Then we introduce how to utilize $\mathcal{S}_{t\alpha_0/T}$ and UHoDA algorithm A to construct $\mathcal{S}_{(t+1)\alpha_0/T}$. We use $A(\mathcal{S}_{t\alpha_0/T}, \mathcal{T}_u)$ to estimate the pseudo labels Y, then use Y to predict the pseudo class centers $\{(\mathbf{c}_t^y, y)\}_{y=1}^K$.

Lastly, we use Eq. (4.3.2) with $\alpha = (t+1)\alpha_0/T$ to generate the source domain $\mathcal{S}_{(t+1)\alpha_0/T}$. Finally, we use $A(\mathcal{S}_{\alpha_0}, \mathcal{T}_u)$ as the final classifier.

Algorithm 2 provides a complete summary of DMU.

4.4 Experiments

The proposed algorithm is tested on several datasets and nine shallow UHoDA algorithms to verify the effectiveness of the proposed Domain MixUp algorithm (DMU).

Table 4.1: Introduction of datasets.

Dataset	Feature	#Sample	#Feature	#Class	Domain
Office-31	DeCAF6	4,110	4,096	31	A,W,D
Office-31	ResNet50	4,110	4,096	31	A,W,D
Offie-Home	ResNet50	15,500	4,096	65	Ar,Cl,Pr,Rw
ImageCLEF-DA	ResNet50	1,800	2,048	12	C,I,P
Office+Caltech256	SURF	2,533	2,048	10	A,W,D,C
Office+Caltech256	DeCAF6	2,533	2,048	10	A,W,D,C
Amazon Review	SURF	7,996	400	2	B,E,D,K

4.4.1 Datasets

We evaluate our algorithm on following UHoDA datasets: **Office-31** (ResNet50) [128], **Office-31** (DeCAF6) [128], **Offie-Home** (ResNet50) [129], **ImageCLEF-DA** (ResNet50) [47], **Office+Caltech10** (SURF) [149], **Office+Caltech256** (DeCAF6) [130,149], and **Amazon Review** [131]. Table 4.1 lists statistics of these datasets.

Office-31 has 4,652 images with 31 common categories, and consists of three real-world object domains: AMAZON (**A**), DSLR (**D**) and WEBCAM (**W**). Similarly, six transfer learning tasks are implemented: $\mathbf{A} \to \mathbf{D}$, $\mathbf{A} \to \mathbf{W}$, $\mathbf{D} \to \mathbf{A}$, $\mathbf{W} \to \mathbf{A}$, $\mathbf{D} \to \mathbf{W}$, $\mathbf{W} \to \mathbf{D}$. Following the standard protocol, we extracted feature vectors from ResNet50 and DeCAF6.

ImageCLEF-DA is a benchmark dataset for ImageCLEF 2014 transfer learning challenge, which aims to classify 12 categories shared in three

Table 4.2: Accuracy (%) on Amazon Review datasets.

Dataset	B→E	B→K	B→D	E→B	E→K	E→D	K→B	K→E	K→D	D→B	D→K	D→E	Avg
1NN	66.3(58.0)↑	68.6(59.8)↑	67.1(60.3)↑	64.4(60.8)↑	71.3(65.2)↑	63.7(60.3)↑	64.1(62.4)↑	70.3(64.2)↑	63.3(60.8)↑	65.6(62.4)↑	65.6(57.7)↑	65.6(58.6)↑	66.3(60.9)↑
PCA+1NN	69.7(65.3)↑	70.9(67.4)↑	68.2(65.5)↑	66.2(61.4)↑	73.6(70.0)↑	65.4(63.5)↑	66.7(65.6)↑	74.8(70.5)↑	66.8(61.7)↑	68.9(65.1)↑	69.4(65.5)↑	66.4(62.1)↑	68.9(65.3)↑
TCA	69.1(64.0)↑	68.8(66.0)↑	72.3(65.3)↑	68.8(62.1)↑	77.7(68.2)↑	70.7(63.3)↑	68.7(64.4)↑	75.1(69.9)↑	70.1(61.3)↑	68.3(66.5)↑	75.4(64.4)↑	72.2(62.4)↑	71.4(64.8)↑
CORAL+1NN	66.3(57.9)↑	68.6(59.3)↑	67.8(60.1)↑	64.1(60.3)↑	71.0(65.4)↑	64.4(60.6)↑	64.3(62.2)↑	71.3(64.9)↑	62.4(60.7)↑	66.8(62.2)↑	65.7(57.7)↑	65.6(57.9)↑	66.5(60.8)↑
EasyTL	80.4(77.5)↑	82.1(79.2)↑	78.5(78.4)↑	75.8(73.0)↑	84.0(84.6)↓	75.3(73.1)↑	76.6(75.2)↑	81.3(82.0)↑	76.5(73.8)↑	79.6(79.5)↑	82.8(80.4)↑	81.3(77.4)↑	79.5(77.8)↑
GFK	77.5(64.8)↑	78.5(65.5)↑	75.8(67.0)↑	71.8(64.7)↑	80.2(70.6)↑	73.9(67.1)↑	73.1(65.7)↑	79.6(72.7)↑	73.3(66.2)↑	75.3(66.9)↑	79.9(67.8)↑	77.0(65.7)↑	76.3(67.1)↑
JDA	63.5(61.7)↑	66.6(66.3)↑	62.0(62.6)↓	61.5(61.0)↑	67.4(68.5)↓	60.7(60.6)↑	66.3(65.7)↑	68.4(69.3)↓	60.8(61.4)↓	63.5(65.0)↓	62.6(62.1)↑	63.3(62.3)↑	63.9(63.9)
JGSA	75.0(68.3)↑	75.7(67.3)↑	73.2(69.1)↑	72.2(67.0)↑	76.6(72.1)↑	69.3(65.4)↑	70.2(67.8)↑	76.3(72.9)↑	68.3(66.0)↑	71.0(68.3)↑	76.4(67.5)↑	71.3(64.4)↑	73.0(68.0)↑
MEDA	79.7(77.1)↑	78.9(77.7)↑	77.4(76.2)↑	75.2(73.0)↑	81.0(82.0)↓	74.8(74.1)↑	74.3(74.1)↑	80.7(80.6)↑	73.9(72.5)↑	77.4(76.6)↑	81.2(79.6)↑	79.9(76.7)↑	77.8(76.7)↑

Table 4.3: Accuracy (%) on Office-Home datasets using ResNet50 features.

Dataset	Ar→Cl	Ar→Pr	Ar→Rw	Cl→Ar	Cl→Pr	Cl→Rr	Pr→Ar	Pr→Cl	Pr→Rw	Rw→Ar	Rw→Pr	Rw→Cl	Avg
1NN	54.0(47.7)↑	75.0(63.9)↑	77.2(69.0)↑	53.5(48.8)↑	70.0(58.7)↑	70.4(62.3)↑	56.7(52.0)↑	49.0(45.6)↑	76.7(71.3)↑	64.9(62.7)↑	81.2(77.1)↑	54.1(51.2)↑	65.2(59.2)↑
PCA+1NN	53.4(48.0)↑	75.4(64.8)↑	77.6(69.6)↑	55.6(48.6)↑	71.5(59.4)↑	71.5(62.9)↑	57.8(51.9)↑	49.2(44.9)↑	77.4(71.6)↑	65.5(62.6)↑	81.5(77.2)↑	54.7(50.8)↑	65.9(59.4)↑
TCA	47.1(41.9)↑	72.1(61.3)↑	73.0(65.5)↑	50.4(45.0)↑	65.1(56.2)↑	65.4(60.2)↑	50.1(47.6)↑	43.4(41.5)↑	72.7(68.3)↑	57.4(57.1)↑	76.5(73.1)↑	47.4(46.5)↑	60.1(55.3)↑
CORAL+1NN	54.0(47.9)↑	75.1(63.9)↑	77.2(69.0)↑	53.8(49.2)↑	70.0(58.7)↑	70.3(62.3)↑	56.4(51.9)↑	49.0(45.6)↑	76.7(71.3)↑	65.0(62.6)↑	81.3(77.1)↑	54.1(51.2)↑	65.2(59.2)↑
EsayTL	53.7(50.9)↑	75.2(68.6)↑	76.8(73.9)↑	56.8(53.4)↑	67.0(62.7)↑	69.1(65.6)↑	56.8(52.2)↑	47.3(46.3)↑	77.3(73.9)↑	64.0(62.8)↑	78.4(76.3)↑	52.6(50.8)↑	64.6(61.4)↑
GFK	45.6(39.2)↑	70.2(58.4)↑	71.4(61.9)↑	48.2(40.7)↑	65.6(54.6)↑	65.1(56.4)↑	49.0(43.4)↑	42.5(38.3)↑	72.4(65.5)↑	55.0(52.8)↑	74.9(70.6)↑	47.0(43.6)↑	58.9(52.1)↑
JDA	49.9(48.8)↑	71.7(67.0)↑	77.8(69.7)↑	51.2(47.3)↑	68.8(64.2)↑	67.8(63.5)↑	53.1(51.1)↑	42.0(43.8)↓	74.2(71.6)↑	59.1(60.6)↓	77.9(77.4)↑	49.5(50.7)↓	61.9(59.6)↑
JGSA	49.6(46.7)↑	70.5(64.8)↑	75.9(70.3)↑	50.3(47.1)↑	69.6(63.5)↑	68.7(62.9)↑	53.2(52.0)↑	44.3(43.5)↑	74.8(72.8)↑	59.1(59.0)↑	77.3(74.7)↑	50.5(48.4)↑	62.0(58.8)↑
MEDA	55.7(54.0)↑	77.1(74.6)↑	79.9(76.2)↑	59.0(58.3)↑	74.1(72.1)↑	73.1(70.6)↑	60.3(58.4)↑	52.7(51.7)↑	79.5(77.5)↑	68.6(68.3)↑	82.0(81.0)↑	57.0(56.4)↑	68.3(66.6)↑

Table 4.4: Accuracy (%) on Office+Caltech256 datasets using DeCAF6 features.

Dataset	C→A	C→A	C→D	A→C	A→W	A→D	W→C	W→A	W→D	D→C	D→A	D→W	Avg
1NN	91.2(88.3)↑	85.4(74.9)↑	86.0(82.8)↑	84.8(80.8)↑	80.7(72.2)↑	84.1(80.9)↑	82.1(72.0)↑	89.8(77.0)↑	100.0(100.0)	84.9(76.0)↑	91.1(81.7)↑	100.0(99.7)↑	88.3(82.2)↑
PCA+1NN	92.5(88.5)↑	92.2(78.0)↑	93.6(86.0)↑	87.7(81.5)↑	82.7(70.8)↑	80.9(79.0)↑	85.9(73.5)↑	91.3(76.7)↑	100.0(100.0)	87.7(76.0)↑	92.6(80.6)↑	99.3(99.0)↑	90.5(82.5)↑
TCA	92.9(90.2)↑	90.2(76.9)↑	92.4(85.4)↑	86.1(82.7)↑	83.7(74.6)↑	85.4(80.3)↑	86.6(79.9)↑	92.3(84.4)↑	100.0(100.0)	86.4(82.5)↑	91.2(88.2)↑	99.7(99.7)	90.6(85.4)↑
CORAL+1NN	91.2(88.2)↑	85.4(74.9)↑	86.0(82.8)↑	84.9(80.8)↑	80.7(71.9)↑	84.1(80.9)↑	81.9(71.6)↑	89.8(76.6)↑	100.0(100.0)	85.0(76.2)↑	91.1(81.6)↑	100.0(99.7)↑	88.3(82.1)↑
EsayTL	92.0(90.2)↑	84.1(76.9)↑	86.0(81.5)↑	86.0(81.7)↑	87.8(74.2)↑	89.2(84.7)↑	82.7(66.3)↑	91.3(73.6)↑	98.7(98.1)↑	82.0(69.1)↑	92.1(76.3)↑	97.6(93.9)↑	89.1(80.5)↑
GFK	92.4(86.8)↑	95.6(83.1)↑	97.5(89.2)↑	87.2(77.6)↑	85.8(73.2)↑	89.8(78.3)↑	86.6(71.6)↑	91.2(77.7)↑	100.0(100.0)	83.6(71.0)↑	93.0(77.6)↑	98.0(98.6)↑	91.7(82.1)↑
JDA	91.6(88.6)↑	94.6(80.3)↑	97.5(87.3)↑	85.0(77.6)↑	83.4(74.6)↑	88.5(77.7)↑	82.3(76.4)↑	88.6(83.4)↑	100.0(100.0)	78.9(75.2)↑	91.2(88.3)↑	98.6(98.6)	90.0(84.0)↑
JGSA	92.1(92.1)	89.5(86.4)↑	93.0(92.4)↑	87.2(85.1)↑	78.0(79.0)↓	80.9(79.6)↑	86.0(84.9)↑	91.3(90.3)↑	100.0(100.0)	85.5(85.0)↑	91.6(91.9)↓	99.7(99.7)	89.6(88.9)↑
MEDA	93.5(93.3)↑	95.6(93.9)↑	95.5(91.7)↑	88.7(87.8)↑	91.6(91.5)↑	92.4(89.8)↑	88.6(87.2)↑	94.1(92.7)↑	99.4(100.0)↓	87.7(86.2)↑	93.6(92.4)↑	98.8(98.6)↑	93.2(92.1)↑

Table 4.5: Accuracy (%) on Office+Caltech256 datasets using SURF features.

Dataset	C→A	C→A	C→D	A→C	A→W	A→D	W→C	W→A	W→D	D→C	D→A	D→W	Avg
1NN	40.3(23.7)↑	35.3(25.8)↑	31.8(25.5)↑	36.4(26.0)↑	35.9(29.8)↑	29.3(25.5)↑	26.6(19.9)↑	33.8(23.0)↑	72.6(59.2)↑	32.7(26.3)↑	38.5(28.5)↑	84.4(63.4)↑	41.5(31.4)↑
PCA+1NN	51.7(42.1)↑	41.7(33.6)↑	40.8(39.5)↑	42.0(38.4)↑	38.6(36.9)↑	36.3(31.2)↑	31.9(29.4)↑	40.6(33.7)↑	86.0(84.1)↑	33.1(31.1)↑	36.0(30.4)↑	91.5(87.1)↑	47.5(43.1)↑
TCA	54.1(46.1)↑	49.2(38.0)↑	47.1(45.9)↑	40.2(40.6)↓	42.7(40.0)↑	35.0(31.8)↑	30.6(31.3)↓	32.0(31.9)↑	90.4(89.2)↑	33.7(33.4)↑	36.2(31.2)↑	91.2(87.5)↑	48.6(45.6)↑
CORAL+1NN	40.6(23.6)↑	34.9(23.7)↑	38.9(26.8)↑	35.4(25.4)↑	32.9(26.8)↑	27.4(26.8)↑	29.0(22.7)↑	33.3(26.2)↑	86.0(84.1)↑	31.4(30.0)↑	35.9(28.8)↑	88.5(84.4)↑	42.8(35.8)↑
EsayTL	50.0(50.1)↓	53.2(49.5)↑	49.0(48.4)↑	39.0(43.0)↓	44.1(40.7)↑	38.9(38.9)	32.7(29.7)↑	36.8(35.2)↑	77.1(77.1)	33.0(31.3)↑	39.7(31.9)↑	74.6(69.5)↑	47.3(45.4)↑
GFK	57.0(41.0)↑	53.9(40.3)↑	46.5(41.4)↑	43.1(40.2)↑	58.3(40.0)↑	48.4(36.3)↑	32.3(30.8)↑	41.0(31.7)↑	91.1(88.5)↑	36.5(30.1)↑	40.1(31.8)↑	88.5(84.4)↑	53.1(44.7)↑
JDA	55.1(42.9)↑	53.9(39.0)↑	42.0(40.8)↑	39.5(39.4)↑	45.1(40.3)↑	29.3(27.4)↑	27.7(30.9)↑	26.7(30.9)↓	87.9(86.0)↑	30.4(31.2)↓	35.7(30.2)↑	90.5(86.1)↑	47.0(43.7)↑
JGSA	53.2(51.5)↑	47.8(45.4)↑	49.7(45.9)↑	40.8(41.5)↓	50.8(45.8)↑	49.0(47.1)↑	31.0(33.2)↓	42.3(39.9)↑	89.2(90.4)↓	32.5(29.9)↑	40.2(38.0)↑	92.2(91.9)↑	51.6(50.0)↑
MEDA	57.6(55.8)↑	59.3(56.6)↑	56.7(59.2)↓	40.9(44.1)↓	52.9(49.2)↑	51.6(46.5)↑	32.3(34.3)↓	42.0(40.8)↑	86.0(87.9)↓	34.5(36.0)↓	41.9(38.6)↑	89.5(87.1)↑	53.8(53.0)↑

Table 4.6: Accuracy (%) on ImageCLEF-DA datasets using ResNet50 features.

Dataset	I→P	P→I	I→C	C→I	C→P	P→C	Avg
1NN	77.7(74.8)↑	91.8(71.5)↑	95.5(89.0)↑	91.8(83.5)↑	75.7(71.3)↑	94.0(75.7)↑	87.8(77.6)↑
PCA+1NN	78.7(75.3)↑	91.0(77.0)↑	95.3(91.3)↑	92.0(87.5)↑	76.8(73.5)↑	95.2(82.2)↑	88.2(81.1)↑
TCA	77.7(77.7)	90.8(79.3)↑	96.5(92.7)↑	92.8(89.3)↑	77.5(76.3)↑	96.3(84.7)↑	88.6(83.3)↑
CORAL+1NN	78.2(75.0)↑	88.8(72.3)↑	95.0(89.8)↑	90.0(79.0)↑	74.0(68.2)↑	92.8(73.2)↑	86.5(76.3)↑
EsayTL	78.7(78.5)↑	90.5(89.5)↑	95.3(93.3)↑	90.5(85.5)↑	74.8(72.0)↑	94.8(91.0)↑	87.4(85.0)↑
GFK	78.5(75.5)↑	91.2(76.3)↑	97.0(93.0)↑	92.5(86.3)↑	76.8(73.3)↑	95.3(81.8)↑	88.6(81.1)↑
JDA	77.7(75.7)↑	90.0(79.5)↑	96.7(93.2)↑	92.2(91.8)↑	75.7(75.7)	93.2(84.5)↑	87.6(83.4)↑
JGSA	78.8(77.5)↑	92.0(83.8)↑	97.3(95.0)↑	92.8(91.2)↑	77.5(76.8)↑	95.3(85.8)↑	89.0(85.0)↑
MEDA	79.8(79.5)↑	92.0(92.0)	96.3(95.2)↑	93.0(91.8)↑	78.7(77.7)↑	96.7(94.5)↑	89.4(88.4)↑

Table 4.7: Accuracy (%) on Office31 datasets using ResNet50 features.

Dataset	A→D	A→W	D→A	D→W	W→A	W→D	Avg
1NN	86.7(79.3)↑	85.5(77.5)↑	70.5(63.8)↑	98.6(97.5)↑	70.1(63.5)↑	99.8(99.6)↑	85.2(80.2)↑
PCA+1NN	86.9(79.5)↑	85.7(76.5)↑	71.2(64.4)↑	98.5(97.7)↑	71.4(64.0)↑	100.0(99.4)↑	85.6(80.3)↑
TCA	84.9(79.3)↑	79.9(74.7)↑	68.4(64.2)↑	94.8(96.6)↓	69.2(62.9)↑	98.2(99.4)↓	82.6(79.5)↑
CORAL+1NN	85.9(79.3)↑	85.5(77.6)↑	70.4(63.8)↑	98.6(97.6)↑	70.2(63.5)↑	99.8(99.4)↑	85.1(80.2)↑
EsayTL	86.3(81.7)↑	82.5(75.6)↑	70.1(64.9)↑	94.1(93.2)↑	72.2(64.9)↑	97.0(96.6)↑	83.7(79.5)↑
GFK	85.1(78.7)↑	83.6(74.8)↑	69.0(63.1)↑	95.0(95.6)↓	70.4(63.3)↑	97.8(99.6)↓	83.5(79.2)↑
JDA	87.6(82.1)↑	87.5(82.5)↑	69.0(68.8)↑	96.5(97.6)↓	72.2(70.1)↑	99.2(99.6)↓	85.3(83.5)↑
JGSA	92.4(89.2)↑	86.9(85.3)↑	70.0(69.6)↑	96.4(97.6)↓	72.8(71.4)↑	98.2(99.6)↓	86.1(85.4)↑
MEDA	88.2(85.1)↑	87.7(84.0)↑	73.6(72.8)↑	97.0(97.5)↓	74.4(72.8)↑	99.0(98.8)↑	86.6(85.2)↑

Table 4.8: Accuracy (%) on Office31 datasets using DeCAF6 features.

Dataset	A→D	A→W	D→A	D→W	W→D	W→A	Avg
1NN	69.9(52.8)↑	63.8(50.4)↑	47.2(33.8)↑	95.2(91.9)↑	98.8(98.0)↑	43.6(33.4)↑	69.8(60.1)↑
PCA+1NN	69.5(56.4)↑	65.8(51.2)↑	50.3(41.1)↑	93.8(92.5)↑	98.8(98.2)↑	46.3(40.8)↑	70.8(63.4)↑
TCA	62.7(48.8)↑	54.1(45.5)↑	40.5(37.7)↑	90.0(89.9)↑	93.6(93.4)↑	37.2(35.6)↑	62.9(58.5)↑
CORAL+1NN	65.3(50.4)↑	66.5(45.9)↑	40.8(33.9)↑	94.8(91.7)↑	97.6(97.4)↑	40.1(31.0)↑	67.5(58.4)↑
EsayTL	57.4(49.4)↑	51.2(45.3)↑	39.9(38.7)↑	85.9(85.3)↑	89.8(89.2)↑	41.6(37.8)↑	61.0(57.6)↑
GFK	61.0(49.6)↑	57.0(47.0)↑	42.9(35.0)↑	90.0(89.8)↑	95.8(96.0)↓	40.6(34.0)↑	64.5(58.6)↑
JDA	65.1(55.4)↑	61.3(55.0)↑	44.0(44.0)	95.3(93.5)↑	97.6(97.6)	41.3(41.3)	67.4(64.5)↑
JGSA	68.5(63.5)↑	64.3(55.8)↑	49.9(50.6)↓	96.6(96.4)↑	97.9(97.8)↑	48.4(47.9)↑	70.8(68.6)↑
MEDA	70.2(65.8)↑	71.1(64.1)↑	54.5(51.1)↑	95.0(93.8)↑	96.8(97.4)↓	54.0(51.6)↑	73.6(70.6)↑

datasets: Caltech-256 (**C**), ImageNet ILSVRC 2012 (**I**), and PascalVOC 2012 (**P**). By considering each as a domain, we build six transfer tasks: **C** $\rightarrow$ **I**, **C** $\rightarrow$ **P**, **I** $\rightarrow$ **P**, **I** $\rightarrow$ **C**, **P** $\rightarrow$ **I**, **P** $\rightarrow$ **C**. Following the standard protocol, we extracted feature vectors from ResNet50.

Office+Caltech256 consists of 2,533 images in 10 categories (8 to 151 images per category per domain), forming four domains: AMAZON (**A**) (images from amazon.com), WEBCAM (**W**) (low-resolution images captured by webcams), DSLR (**D**) (high-resolution images captured by a digital SLR camera), and CALTECH (**C**) (images from the Caltech-256 dataset). Twelve transfer learning tasks are implemented: **A** $\rightarrow$ **D**, **A** $\rightarrow$ **W**, **A** $\rightarrow$ **C**, ..., **W** $\rightarrow$ **C**. Following the standard protocol and for a fair comparison with the other algorithms, two types of features are used: SURF and Decaf6.

Office-Home consists of 4 different domains: Artistic (**Ar**), Clipart (**Cl**), Product (**Pr**) and Real-World (**Rw**). Each domain contains images from 65 object classes. We constructed 12 transfer learning tasks: **Ar** $\rightarrow$ **Cl**, **Ar** $\rightarrow$ **Pr**,..., **Rw** $\rightarrow$ **Ar**. Following the standard protocol and for fair comparison with the other methods, we also extracted feature vectors from ResNet50.

Amazon Review is a cross-domain sentiment analysis dataset that contains positive and negative reviews of four kinds of products: Kitchen appliance (**K**), DVDs (**D**), Electronics (**E**), and Books (**B**). We use all domain combinations and build twelve transfer tasks: **K** $\rightarrow$ **D**, **K** $\rightarrow$ **E**,..., **B** $\rightarrow$ **D**, **B** $\rightarrow$ **E**, **B** $\rightarrow$ **K**.

4.4.2 Experimental Setup

Before reporting the detailed evaluation results, it is important to explain how to tune hyper-parameters for DMU and selected UHoDA algorithms.
• DMU. DMU has two parameters T and ρ_0. For all tasks, we set $T = 10$. Next, we discuss how to select ρ_0. First, ρ_0 should not be too small, because if ρ_0 is too small, few target information is contained by $\mathcal{S}_{\rho_0}$. Second, ρ_0 should not be too large, because if ρ_0 is too large, few source

information is contained by $\mathcal{S}_{\rho_0}$, thus if the pseudo class centers $\{\mathbf{c}_t^y, y\}_{y=1}^K$ are not accurate, we may obtain a poor transfer performance. Hence, we select $\rho = 0.6$ for all tasks.

• 1NN, CORAL and EasyTL. The three algorithms are parameter free.

• PCA+1NN. Let the reduced dimension for PCA be 100 for all tasks.

• GFK [66]. GFK dimension d is the unique parameter for GFK. We set $d = 20$ for all tasks.

• TCA [5]. There are at least three parameters: the kernel function, the TCA dimension d and the regularization parameter λ. We set the linear kernel as the kernel function, $d = 100$ and $\lambda = 1$ for all tasks.

• JDA [65]. There are at least three parameters: the kernel function, the JDA dimension d and the regularization parameter λ. We set the gaussian kernel with kernel radius 1 as the kernel function, $d = 100$ and $\lambda = 1$ for all tasks.

• JGSA [35]. There are at least four parameters: the kernel function, the JGSA dimension d, subspace divergence parameter λ, the target variance parameter μ and the source discrimination parameter β. We set the gaussian kernel with kernel radius 2 as the kernel function, $d = 30$, $\lambda = 1$, $\mu = 1$ and $\beta = 0.1$ for all tasks.

• MEDA [147]. There are at least four parameters: the distribution alignment parameter λ, the manifold regularization parameter ρ, the number of neighbours p and regularization parameter η. We set $\lambda = 1$, $\rho = 1$, $p = 10$ and $\eta = 0.1$.

Note that JDA, JGSA and MEDA are iterative algorithms. For the baseline, we implement JDA, JGSA and MEDA with 10 iterations. For JDA+DMU, JGSA+DMU and MEDA+DMU, we set the number of iterations as 1 for JDA, JGSA and MEDA. That is because DMU itself is an iterative algorithm, and it is unnecessary to implement JDA, JGSA and MEDA iteratively within every iteration of DMU. In addition, JDA, JGSA and MEDA needs the pseudo labels within every iteration. For the first iteration, we use the pseudo labels predicted by 1NN. Later, we use the pseudo labels predicted by last iteration.

For fair comparison, the algorithm parameters are set with the same values for all tasks. Hence, the performance for some tasks may be not same with the performance reported by original papers.

4.4.3 Experimental Results

The classification accuracy for all tasks is shown in Tables 4.2-4.8. In these tables, the accuracy in brackets is the performance without using DMU, the green arrow means that the performance with using DMU is better than the performance without using DMU and the red arrow means that the performance with using DMU is worse than the performance without using DMU. From observing and analyzing the results in these tables, we could summarize the following results.

• 1NN, PCA+1NN and CORAL+1NN.

The performance of 1NN+DMU, PCA+1NN+DMU, CORAL+1NN+DMU is much better than 1NN, PCA+1NN and CORAL+1NN in most tasks (65/66, 65/66, 65/66), respectively. There is only one task (W→D in Office+Caltech256 DeCAF6) that the performance (100.0%) of 1NN+DMU, PCA+1NN+DMU and CORAL+1NN+DMU is equal to that (100.0%) of 1NN, PCA+1NN and CORAL+1NN. Compared to 1NN, PCA+1NN and CORAL+1NN, the Avgrage performance improvements are 7.4%, 5.9% and 6.7%, which shows DMU does construct more suitable source domain for algorithms 1NN, PCA+1NN and CORAL+1NN.

• TCA.

TCA+DMU has achieved better performance than TCA in most tasks (60/66). There are four tasks (**A→C**, **W→C** in **Office+Caltech256** SURF and **D→W**, **W→D** in **Office31** ResNet50) that the performance (40.2%, 30.6%, 94.8%, 98.2%) of TCA+DMU is worse than that (40.6%, 31.3%, 96.6%, 99.4%) of TCA. TCA+DMU has achieved the Average performance improvement about 4.7%.

• EasyTL.

EasyTL+DMU has achieved better performance than EasyTL in most tasks (62/66). There are three tasks (**E→K** in **Amazon Review**, **C→A**,

A→C in **Office+Caltech256** SURF) that the performance (84.0%, 50.0%, 39.0%) of EasyTL+DMU is worse than that (84.6%, 50.1%, 43.0%) of EasyTL. DMU helps EasyTL to improve the Average performance about 3.7%.

• GFK.

GFK+DMU has achieved better performance than EasyTL in most tasks (63/66). There are three transfer tasks (**D→W**, **W→D** in **Office31** ResNet50 and **W→D** in **Office31** DeCAF6) that the performance (95.0%, 97.8%, 95.8%) of GFK+DMU is worse than that (95.6%, 99.6%, 96.0%) of GFK. DMU does help GFK to improve the Average performance about 7.8%.

• JDA.

Compared to JDA, JDA+DMU improves the performance in 53 tasks. There are 10 tasks that JDA+DMU has worse performance than JDA. DMU can help JDA to improve about 2.9% mean accuracy.

• JGSA.

Compared to JGSA, JGSA+DMU achieves better performance in 55 tasks. There are 8 tasks that JGSA+DMU has worse performance than JGSA. JGSA+DMU has achieved about 2.5% mean improvement.

• MEDA.

MEDA+DMU achieves better performance in 55 tasks than MEDA. There are 9 tasks that MEDA+DMU has worse performance than MEDA. DMU can help MEDA to improve about 1.3% mean accuracy.

According to above observations, we summarize the following conclusions:

1. DMU does help selected algorithms to improve the performance in most tasks and improve the mean performance at least 1.3% (at most 7.8%). Hence, we have empirically proven that for selected algorithms, there exists at least a source domain reconstruction algorithm in most transfer learning tasks.

2. We discover that a better algorithm may have a lower performance improvement. For example, MEDA as the best algorithm in selected al-

gorithms, has been improved 1.3% mean performance by DMU. However, 1NN has been improved 7.4% mean performance by DMU.

3. Every UHoDA algorithm uses the same parameters for all tasks for fair comparison. Hence, it is possible that the performances of some algorithms reported in this chapter are poorer than the performance reported in original paper, where the parameters are finely tuned for different datasets to get the best performance. Nevertheless, DMU still could help those algorithms to achieve better performance than their best performance reported in original papers. For examples, the mean performance of MEDA+DMU for all tasks is 76.0%, and the used MEDA have not been finely tuned and are the same for all datasets. However, the mean performance of MEDA reported in [150] for all tasks is 75.2% that is lower than our method. Hence, DMU not only improves the performance, but also improves the parameter robustness for selected algorithms.

4.4.4 Parameter Sensitivity

These studies are conducted on different types of datasets and UHoDA algorithms to demonstrate that a wide range of parameter values can be chosen to obtain satisfactory performance. We evaluate DMU's parameters ρ_0 and T, reporting the results on different algorithms (1NN, GFK, EasyTL, JGSA and MEDA) and different transfer learning tasks: **Amazon Review** (**B**→**K**), **ImageClef-DA** (**P**→**C**), **Office-Home** (**Ar**→**Pr**) and **Office31** ResNet50 (**A**→**D**). The dashed line denotes the results of the selected algorithms with each task.

Parameter ρ_0. The parameter ρ_0 is the index of intermediate domain $\mathcal{S}_{\rho_0}$. We run DMU with varying values of ρ_0. Fig. 4.2 plots the classification accuracy w.r.t. to different values of ρ_0 from $[0.1, 1]$. From Fig. 4.2, we can see that: 1) When $\rho_0 = 0.1$, the performance is worst for most tasks (18/20). 2) After the increasing of the ρ_0 from 0.1 to 1, the performance gradually increases for most tasks (16/20). 3) DMU can help selected algorithms to improve accuracy for all $\rho_0 \in [0.1, 1]$.

Parameter T. T is the number of intermediate domains to connect $\mathcal{S}_0$ and $\mathcal{S}_{\rho_0}$. In these experiments, we run DMU with varying values of T from 1 to 20. Larger values of T implies more intermediate domains are designed for constructing $\mathcal{S}_{\rho_0}$. From Fig. 4.3, we can see that: 1) DMU helps selected algorithms to achieve steady and consistently good performance when $\rho \in [9, 20]$. 2) For different algorithms, the behaviours of accuracy on changing T are different. For example, on task Office-Home (**Ar**→**Pr**), the performance of algorithms 1NN+DMU, GFK+DMU and EasyTL+DMU gradually increase from 1 to 13 and slightly decrease when T varies from $[14, 20]$. However, for all T from $[2, 20]$, JGSA+DMU and MEDA+DMU can achieve a steady performance with small fluctuations. Overall, DMU can help selected algorithms to improve accuracy for all $T \in [9, 20]$.

4.5 Summary

This chapter presents a new problem setting for the domain adaptation field, called source domain reconstruction (SDR), which constructs a new source domain by the given source domain, target domain and UHoDA algorithm. To explore the feasibility of SDR, we analyze it theoretically. Further, we propose a novel Domain MixUp algorithm with utilizing MixUp technique. Nine famous UHoDA algorithms are selected to evaluate the ability of SDR algorithm in reconstructing source sequence and enhance the effectiveness of unsupervised homogeneous domain adaptation. Experiments conducted on 66 transfer tasks and nine UHoDA algorithms confirm that DMU solves the SDR problem on selected UHoDA algorithms.

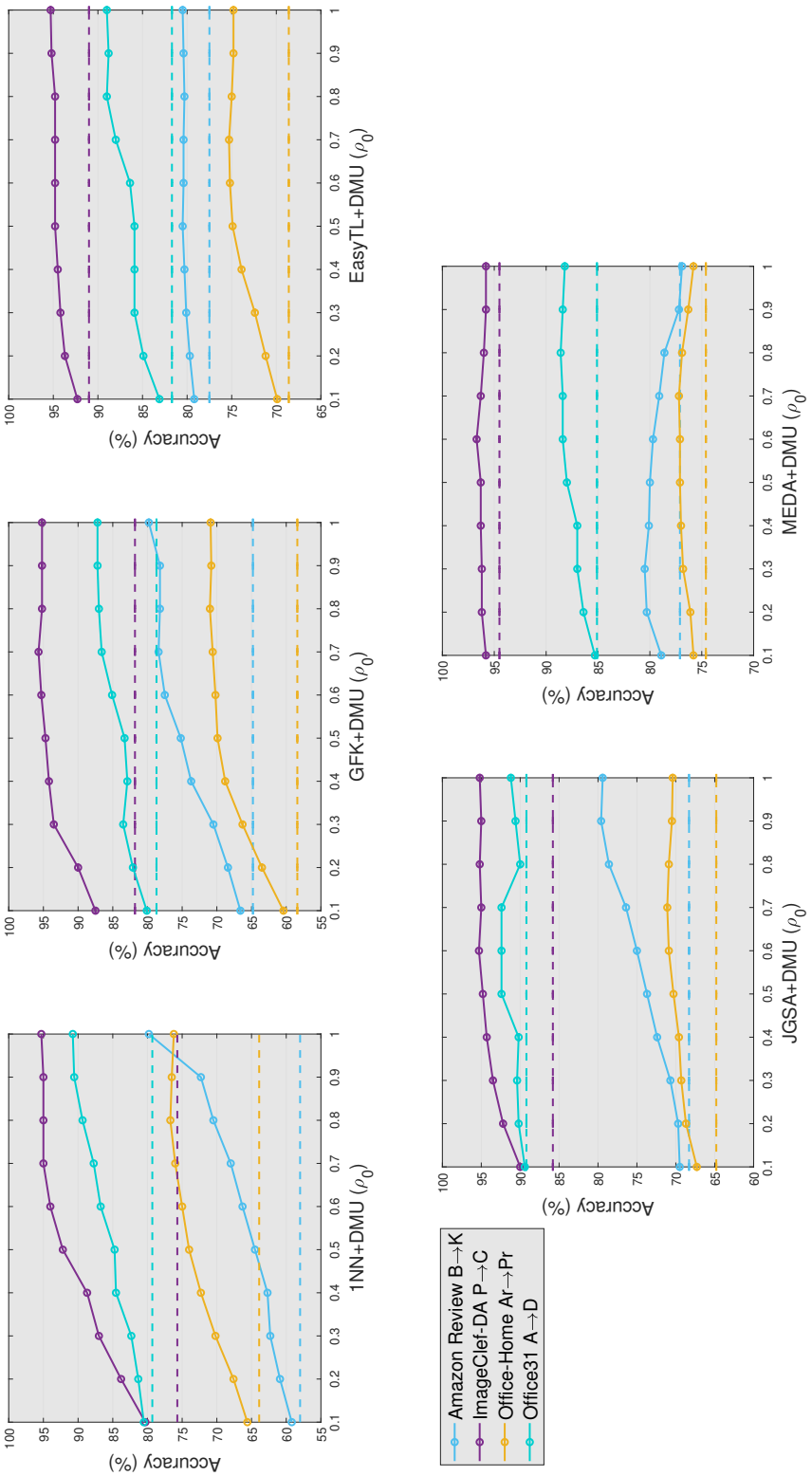


Figure 4.2: Parameter analysis for proposed algorithm DMU on ρ_0.

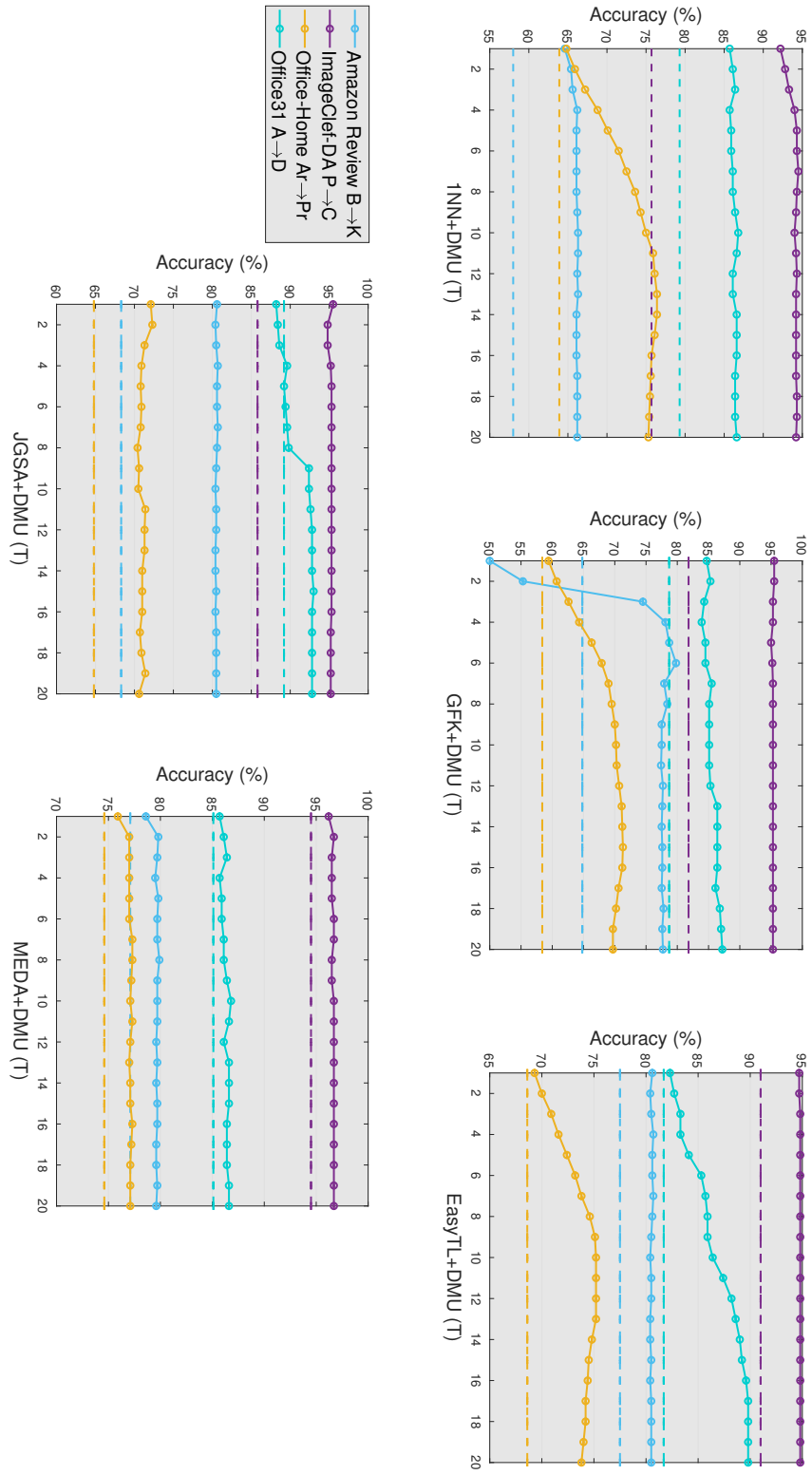


Figure 4.3: The parameter analysis for proposed algorithm DMU on T.

Chapter 5

Unsupervised Open-Set Domain Adaptation: Theory and Algorithm

This chapter proposes a theoretical foundation for open-set domain adaptation and uses the theoretical observation to address the UOSDA problem.

The material of this chapter is based on the publication:
Zhen Fang, Jie Lu, Feng Liu, Junyu Xuan, and Guangquan Zhang, Open Set Domain Adaptation Theoretical Bound and Algorithm, IEEE Transactionson Neural Networks and Learning Systems (TNNLS), 2020.

5.1 Introduction

The aim of unsupervised homogeneous domain adaptation (UHoDA) is to minimize the discrepancy between the distributions of two domains. Existing works on UHoDA mainly focus on domain-invariant features, where the marginal distributions or conditional distributions from the two domains are similar [5, 65, 94]. There is an implicit assumption in most existing UHoDA algorithms [15,151–154] that the source and target domains share the same label set. Under this assumption, UHoDA is also regarded as unsupervised closed-set domain adaptation (UCSDA) [27].

However, this assumption in UCSDA algorithms is not realistic in an unsupervised setting (i.e., there are no labels in the target domain), since it is not known whether the classes of target samples are from the label set of the source domain. It may be that the target domain contains additional classes (*unknown classes*) that do not exist in the label set

of the source domain [28]. For example, in the Syn2Real task [155], there may be more classes for the real-world objects in the target domain than the synthetic objects contained in the source domain. Therefore, if existing UCSDA algorithms are used to solve the UDA problem without the assumption in UCSDA, the potential mismatches between unknown and known classes would likely result in negative transfer [156] (see Fig. 5.1(b)).

To address UDA problem without the assumption, Busto et al. [27] and Saito et al. [28] recently proposed a new problem setting, unsupervised open-set domain adaptation (UOSDA), in which the unlabeled target domain contains unknown classes that do not belong to the label set of the source domain (see Fig. 1). There are two key challenges [28] in addressing the UOSDA problem. The first challenge is that there is not enough knowledge in the target domain to classify the unknown samples. So how should these samples be labeled? The solution is to mine deeper information in the target domain to delineate a boundary between the known and unknown classes. The second challenge in UOSDA is the difference in distributions. The unknown target samples should not be matched when the overall distribution is matched, otherwise negative transfer may occur.

Several algorithms have been proposed to solve the UOSDA problem [27, 28, 157–159]. The first proposed UOSDA algorithm is Assign-and-Transform-Iteratively (ATI) [27], which recognizes unknown target samples using constraint integer programming. It then learns a linear map to match the source domain with the target domain by excluding the predicted unknown target samples. However, ATI carries the assumption that the source domain contains unknown classes that are not in the target domain. Hence, the first proposed deep UOSDA algorithm, Open-Set Back Propagation (OSBP) [28] is developed to address the UOSDA problem without this assumption. It rejects unknown samples by training a binary cross entropy loss.

Although ATI and OSBP are designed to solve the UOSDA problem,

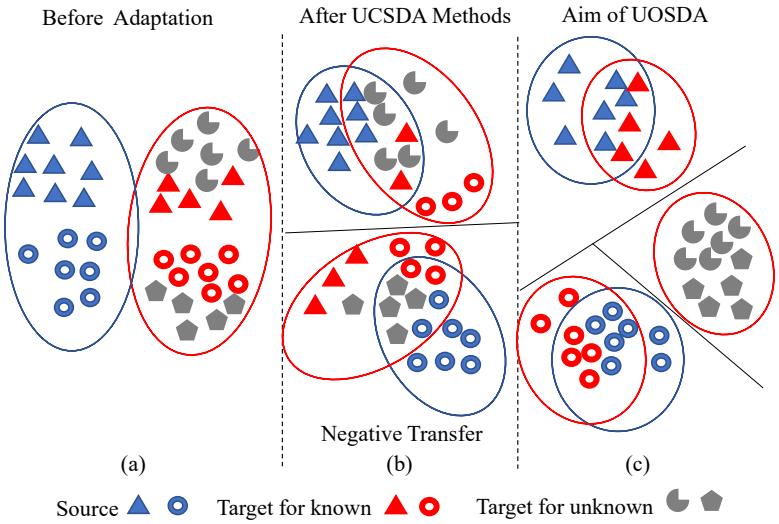


Figure 5.1: Aim of UOSDA. (a) The original source and target samples are given. (b) UCSDA algorithm matches the source and target samples, leading to negative transfer. Because the unknown target samples interfere with distribution matching. (c) UOSDA algorithm classifies known target samples into the correct known classes and recognizes the unknown target samples as unknown.

neither is based on a theoretical analysis of UOSDA. Moreover, no work has yet given a learning bound for open-set domain adaptation problems. To fill this gap, this chapter presents a theoretical exploration of UOSDA. In studying the risk of the target classifier on unknown classes, we discovered the risk is closely related to a special term called *open-set difference* which can be estimated from the unlabeled samples. Minimizing the open-set difference helps us to classify unknown target samples, addressing the first challenge.

Following our theory, we design a principle-guided UOSDA algorithm referred to as Distribution Alignment with Open Difference (DAOD). This algorithm can accurately classify unknown target samples while minimizing the discrepancy between the two domains for known classes. DAOD learns the target classifier by simultaneously optimizing the structural risk function [160], the joint distribution alignment, the manifold regularization [121], and open-set difference. The reason DAOD is able to avoid negative transfer lies in its ability to minimize the open-set difference, which enables the unknown target samples to be classified

accurately as unknown. By excluding these recognized unknown target samples, the source and target domains can be precisely aligned, addressing the second challenge.

As mentioned, there is no theoretical work in the literature for open-set domain adaptation. The closest theoretical work is by Ben-David et al. [17], who gives VC-dimension-based generalization bounds. Unfortunately, this work has several restrictions: 1) the theoretical analysis only covers closed-settings; and 2) the work only solves binary classification tasks, rather than the multi-class problems common to open settings. A significant contribution of this chapter is that the theoretical work gives a learning bound for open-set domain adaptation.

The contributions of this chapter are summarized as follows:

• We provide the theoretical bound for open-set domain adaptation. The closed-set domain adaptation theory [17] is a special case of our theoretical results. To the best of our knowledge, this is the first work on open-set domain adaptation theory.

• We develop an unsupervised novel open-set domain adaptation algorithm, Distribution Alignment with Open Difference (DAOD), which is based on the open-set learning bound proposed. The algorithm enables the unknown target samples to be separated from samples using open-set difference.

• We conduct 38 real-world UOSDA tasks (including 20 face recognition tasks and 18 object recognition tasks) for evaluating DAOD and existing UOSDA algorithms. Extensive experiments demonstrate that DAOD outperforms the UOSDA algorithms ATI and OSBP.

5.2 Theoretical Analysis

In this section, the definition of unsupervised open-set domain adaptation is recalled. Then, necessary concepts are introduced. Lastly, theoretical analysis is presented.

5.2.1 Problem Setting and Necessary Concepts

Assume that feature spaces $\mathcal{X}_s$, $\mathcal{X}_t$ and label sets $\mathcal{Y}_s$, $\mathcal{Y}_t$ satisfy $\mathcal{X}_s = \mathcal{X}_t$, $\mathcal{Y}_s \subset \mathcal{Y}_t$. Given sets of samples called the *labeled source* and *unlabeled target* samples:

$$\mathcal{S} = \{(\mathbf{x}_s^i, \mathbf{y}_s^i)\}_{i=1}^{n_s} \sim P_{X_s Y_s} \ i.i.d.$$
$$\mathcal{T}_u = \{\mathbf{x}_u^i\}_{i=1}^{n_u} \sim P_{X_t} \ \ i.i.d.$$

where $P_{X_s} \neq P_{X_t}$. The aim of **unsupervised open-set domain adaptation** (UOSDA) is to train a classifier $g : \mathcal{X}_t \to \mathcal{Y}_t$ such that g classifies 1) target samples, whose labels are from $\mathcal{Y}_s$, into the correct classes; 2) target samples, whose labels are from $\mathcal{Y}_t - \mathcal{Y}_s$, as unknown.

For simplicity, we assume that the feature space is $\mathcal{X}$ ($\mathcal{X} = \mathcal{X}_s = \mathcal{X}_t$). In addition, the UOSDA algorithm only needs to classify unknown samples as unknown and classify known target samples into the correct known classes. Classifying unknown target samples into correct unknown classes is not necessary. Hence, we consider all unknown target samples are allocated to one big unknown class. For the sake of simplicity, we assume that $\mathcal{Y}_s = \{\mathbf{y}_c\}_{c=1}^{K}$, $\mathcal{Y}_t = \{\mathbf{y}_c\}_{c=1}^{K+1}$, where the label $\mathbf{y}_{K+1}$ represents the unknown target classes and the label $\mathbf{y}_c \in \mathbb{R}^{K+1}$ is a one-hot vector, whose c-th coordinate is 1 and other coordinates are 0. The label $\mathbf{y}_c$ represents the c-th class.

5.2.2 Concepts and Notations

Before introducing our main results, we need to introduce the following necessary concepts and notations in this field. Unless otherwise specified, all the following notations are used consistently throughout the chapter without further explanations.

5.2.2.1 Notations for distributions

For the sake of simplicity, $P_{X_s|\mathbf{y}_c}$ and $P_{X_t|\mathbf{y}_c}$ represent the conditional distributions for the c-th class $P_{X_s|Y_s=\mathbf{y}_c}$ and $P_{X_t|Y_t=\mathbf{y}_c}$, while π_t^c represents

the target class-prior probability for c-th class $P_{Y_t=\mathbf{y}_c}$. Hence, $\pi_t^{K+1} = P_{Y_t=\mathbf{y}_{K+1}}$ is the class-prior probability for the unknown target classes.

Lastly, $P_{X_t|\mathcal{Y}_s}$ represents the target conditional distribution for the known classes $P_{X_t|Y_t\in\mathcal{Y}_s}$, which can be evaluated by

$$\frac{P_{X_tY_t\in\mathcal{Y}_s}}{P_{Y_t\in\mathcal{Y}_s}} = \frac{\sum_{c=1}^{K} P_{X_t|Y_t=\mathbf{y}_c}\pi_t^c}{1-\pi_t^{K+1}}.$$

The notation $\widehat{P}$ denotes the corresponding empirical distribution to any distribution P. For example, $\widehat{P}_{X_sY_s}$ represents the empirical distribution corresponding to $P_{X_sY_s}$.

5.2.2.2 Risks and Partial Risks

Risks and partial risks are two important concepts in learning theory, which are briefly explained in the following and later used in our theorems.

Following the notations in [99], we consider a multi-class classification task with a hypothesis space $\mathcal{H}$ of the scoring functions

$$\begin{aligned} \boldsymbol{h}:\ \mathcal{X} &\to \mathbb{R}^{|\mathcal{Y}_t|} = \mathbb{R}^{K+1} \\ \mathbf{x} &\to [h_1(\mathbf{x}), ..., h_{K+1}(\mathbf{x})]^T, \end{aligned} \tag{5.2.1}$$

where the output $h_c(\mathbf{x})$ indicates the confidence in the prediction of the label $\mathbf{y}_c$. Let $\ell : \mathbb{R}^{K+1} \times \mathbb{R}^{K+1} \to \mathbb{R}_+$ be a symmetric loss function. Then the risks of $\boldsymbol{h} \in \mathcal{H}$ w.r.t. ℓ under $P_{X_sY_s}$ and $P_{X_tY_t}$ are given by

$$\begin{aligned} R_s(\boldsymbol{h}) &= \mathop{\mathbb{E}}_{(\mathbf{x},\mathbf{y})\sim P_{X_sY_s}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}) = \mathbb{E}\ \ell(\boldsymbol{h}(X_s), Y_s)), \\ R_t(\boldsymbol{h}) &= \mathop{\mathbb{E}}_{(\mathbf{x},\mathbf{y})\sim P_{X_tY_t}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}) = \mathbb{E}\ \ell(\boldsymbol{h}(X_t), Y_t)). \end{aligned} \tag{5.2.2}$$

The partial risk of $\boldsymbol{h} \in \mathcal{H}$ for the known target classes is

$$R_t^*(\boldsymbol{h}) = \frac{1}{1-\pi_t^{K+1}} \int_{\mathcal{X}\times\mathcal{Y}_s} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}) \mathrm{d}P_{X_tY_t}(\mathbf{x}, \mathbf{y}) \tag{5.2.3}$$

and the partial risk of $\boldsymbol{h} \in \mathcal{H}$ for the unknown target classes is

$$\begin{aligned} R_t^{K+1}(\boldsymbol{h}) &= \mathop{\mathbb{E}}_{\mathbf{x}\sim P_{X_t|\mathbf{y}_{K+1}}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}) \\ &= \int_{\mathcal{X}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}) \mathrm{d}P_{X_t|\mathbf{y}_{K+1}}(\mathbf{x}). \end{aligned} \tag{5.2.4}$$

According to Eq. (5.2.2), Eq. (5.2.3) and Eq. (5.2.4), we have

$$R_t(\boldsymbol{h}) = \pi_t^{K+1} R_t^{K+1}(\boldsymbol{h}) + (1 - \pi_t^{K+1}) R_t^*(\boldsymbol{h}). \tag{5.2.5}$$

Lastly, we denote

$$\begin{aligned} R_s^{u,K+1}(\boldsymbol{h}) &= \mathop{\mathbb{E}}_{\mathbf{x}\sim P_{X_s}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}) = \mathbb{E}\ \ell(\boldsymbol{h}(X_s), \mathbf{y}_{K+1}), \\ R_t^{u,K+1}(\boldsymbol{h}) &= \mathop{\mathbb{E}}_{\mathbf{x}\sim P_{X_t}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}) = \mathbb{E}\ \ell(\boldsymbol{h}(X_t), \mathbf{y}_{K+1}) \end{aligned} \tag{5.2.6}$$

as the risks that the samples are regarded as the unknown classes.

Given a risk $R(\boldsymbol{h})$, it is convenient to use notation $\widehat{R}(\boldsymbol{h})$ as the empirical risk that corresponds to $R(\boldsymbol{h})$. Hence, notations $\widehat{R}_s(\boldsymbol{h})$, $\widehat{R}_s^{u,K+1}(\boldsymbol{h})$ and $\widehat{R}_t^{u,K+1}(\boldsymbol{h})$ represent the empirical risks corresponding to the risks $R_s(\boldsymbol{h})$, $R_s^{u,K+1}(\boldsymbol{h})$ and $R_t^{u,K+1}(\boldsymbol{h})$, respectively.

5.2.2.3 Main Theoretical Result and Open-Set Difference

Before introducing the main theorem, we firstly define open-set difference, one of the main contributions of this work.

Definition 17 (Open-Set Difference)**.** *Given risks $R_s^{u,K+1}(\boldsymbol{h})$, $R_t^{u,K+1}(\boldsymbol{h})$ defined in (5.2.6), the open-set difference is*

$$\Delta_o = \frac{R_t^{u,K+1}(\boldsymbol{h})}{1 - \pi_t^{K+1}} - R_s^{u,K+1}(\boldsymbol{h}),$$

where π_t^{K+1} is the class-prior probability for the unknown target classes.

The following theorem provides an open-set domain adaptation bound according to discrepancy distance and open-set difference.

Theorem 5. *Given a symmetric loss function ℓ satisfying triangle inequality and a hypothesis $\mathcal{H}$ with a mild condition that the constant vector value function $\boldsymbol{g} = \mathbf{y}_{K+1} \in \mathcal{H}$, then for any $\boldsymbol{h} \in \mathcal{H}$, we have*

$$\begin{aligned}\frac{R_t(\boldsymbol{h})}{1-\pi_t^{K+1}} \leq & \overbrace{R_s(\boldsymbol{h})}^{\textit{Source Risk}} + 2\overbrace{d_{\mathcal{H}}^{\ell}(P_{X_t|\mathcal{Y}_s}, P_{X_s})}^{\mathcal{H}\Delta\mathcal{H}\textit{-divergence}} + \Lambda \\ & + \underbrace{\frac{R_t^{u,K+1}(\boldsymbol{h})}{1-\pi_t^{K+1}} - R_s^{u,K+1}(\boldsymbol{h})}_{\textit{Open-Set Difference } \Delta_o},\end{aligned}$$

where $R_s(\boldsymbol{h})$ and $R_t(\boldsymbol{h})$ are the risks defined in Eq. (5.2.2), $R_s^{u,K+1}(\boldsymbol{h})$ and $R_t^{u,K+1}(\boldsymbol{h})$ are the risks defined in Eq. (5.2.6), $R_t^*(\boldsymbol{h})$ is the partial risk defined in Eq. (5.2.3), and $\Lambda = \min_{\boldsymbol{h}\in\mathcal{H}} \big(R_s(\boldsymbol{h}) + R_t^*(\boldsymbol{h})\big)$.

Proof. Here we provide a proof sketch. Detailed proof is given in Appendix B.1. According to Eq. (5.2.5), we have

$$\begin{aligned}&\frac{R_t(\boldsymbol{h})}{1-\pi_t^{K+1}} - R_s(\boldsymbol{h}) \\ =&R_t^*(\boldsymbol{h}) - R_s(\boldsymbol{h}) + \frac{\pi_t^{K+1}}{1-\pi_t^{K+1}} R_t^{K+1}(\boldsymbol{h}).\end{aligned} \tag{5.2.7}$$

Then, we can check that

$$R_t^*(\boldsymbol{h}) - R_s(\boldsymbol{h}) \leq \Lambda + d_{\mathcal{H}}^{\ell}(P_{X_t|\mathcal{Y}_s}, P_{X_s}), \tag{5.2.8}$$

$$\frac{\pi_t^{K+1}}{1-\pi_t^{K+1}} R_t^{K+1}(\boldsymbol{h}) \leq d_{\mathcal{H}}^{\ell}(P_{X_t|\mathcal{Y}_s}, P_{X_s}) + \Delta_o. \tag{5.2.9}$$

Combining Eq. (5.2.8), inequality (5.2.9) with inequality (5.2.7), we have

$$\frac{R_t(\boldsymbol{h})}{1-\pi_t^{K+1}} \leq R_s(\boldsymbol{h}) + 2d_{\mathcal{H}}^{\ell}(P_{X_t|\mathcal{Y}_s}, P_{X_s}) + \Lambda + \Delta_o.$$

□

Remark 4. *The condition $\mathbf{y}_{K+1} \in \mathcal{H}$ can be replaced by a weaker condition that there exists a sequence $\{\boldsymbol{h}_i\}_{i=1}^{+\infty}$ such that $\boldsymbol{h}_i$ converges uniformly to $\mathbf{y}_{K+1}$. Note that the hypothesis space $\mathcal{H}$ used in our algorithm satisfies*

the weaker condition automatically, thus, the condition $\mathbf{y}_{K+1} \in \mathcal{H}$ *can be removed when we use the hypothesis space* $\mathcal{H}$ *applied in our algorithm.*

The open-set difference Δ_o is the crucial term to bound the risk of $\boldsymbol{h}$ on unknown target classes, since

$$R_t^{K+1}(\boldsymbol{h}) \leq \frac{1-\pi_t^{K+1}}{\pi_t^{K+1}} \left(\Delta_o + d_{\mathcal{H}}^{\ell}(P_{X_t|\mathcal{Y}_s}, P_{X_s})\right). \tag{5.2.10}$$

The risk of $\boldsymbol{h}$ on unknown target classes is intimately linked to the open-set difference Δ_o:

$$\left|\pi_t^{K+1} R_t^{K+1}(\boldsymbol{h}) - (1-\pi_t^{K+1})\Delta_o\right| \leq d_{\mathcal{H}}^{\ell}(P_{X_t|\mathcal{Y}_s}, P_{X_s}).$$

When $\pi_t^{K+1} = 0$, Theorem 5 degenerates into the closed-set scenario with this theoretical bound [17]

$$R_t(\boldsymbol{h}) \leq R_s(\boldsymbol{h}) + 3d_{\mathcal{H}}^{\ell}(P_{X_t}, P_{X_s}) + \Lambda.$$

This is because when $\pi_t^{K+1} = 0$, the open-set difference is

$$\Delta_o \leq d_{\mathcal{H}}^{\ell}(P_{X_t|\mathcal{Y}_s}, P_{X_s}) = d_{\mathcal{H}}^{\ell}(P_{X_t}, P_{X_s}).$$

The significance of Theorem 5 is twofold. First, it highlights that the open-set difference Δ_o is the main term for controlling performance in open-set domain adaptation. Second, the bound shows a direct connection with the closed-set domain adaptation theory.

In addition, the open-set difference Δ_o consists of two parts: a positive term $R_t^{u,K+1}(\boldsymbol{h})$ and a negative term $R_s^{u,K+1}(\boldsymbol{h})$. A larger positive term implies more target samples are classified as unknown samples. The negative term is used to prevent the source samples from being classified as unknown. According to inequality (5.2.10), the negative term and the distance discrepancy jointly prevent all target samples from being recognized as unknown classes. In addition, Corollary 5.1 also tells us that the positive term and the negative term can be estimated from unlabeled samples. Using Natarajan Dimension Theory [161] to bound the source

risk $R_s(\boldsymbol{h})$, risks $R_t^{u,K+1}(\boldsymbol{h})$ and $R_s^{u,K+1}(\boldsymbol{h})$ by empirical estimates $\widehat{R}_s(\boldsymbol{h})$, $\widehat{R}_t^{u,K+1}(\boldsymbol{h})$ and $\widehat{R}_s^{u,K+1}(\boldsymbol{h})$ respectively, we have the following result.

Corollary 5.1. *Given a symmetric loss function ℓ satisfying the triangle inequality and bounded by B, and a hypothesis $\mathcal{H} \subset \{\boldsymbol{h} : \mathcal{X} \to \mathcal{Y}_t\}$ with conditions: 1) the constant vector value function $\boldsymbol{g} = \mathbf{y}_{K+1} \in \mathcal{H}$, 2) the Natarajan dimension of $\mathcal{H}$ is d, if a random labeled samples of size n_s is generated by source joint distribution $P_{X_sY_s}$-i.i.d. and a random unlabeled samples of size n_u is generated by target marginal distribution P_{X_t}-i.i.d., then for any $\boldsymbol{h} \in \mathcal{H}$ and $\delta \in (0,1)$ with probability at least $1 - 4\delta > 0$, we have*

$$
\begin{aligned}
|\frac{R_t(\boldsymbol{h})}{1-\pi_t^{K+1}} - \widehat{R}_s(\boldsymbol{h})| \leq\ & 2d_{\mathcal{H}}^{\ell}(P_{X_t|\mathcal{Y}_s}, P_{X_s}) + \widehat{\Delta}_o + \Lambda \\
& + 4B\sqrt{\frac{8d\log n_s + 16d\log(K+1) + 2\log 2/\delta}{n_s}} \\
& + 2B\sqrt{\frac{8d\log n_u + 16d\log(K+1) + 2\log 2/\delta}{\left(1-\pi_t^{K+1}\right)^2 n_u}},
\end{aligned}
$$

where $\mathcal{X}$ is the feature space, $\mathcal{Y}_t$ is the target label space, $\widehat{R}_s(\boldsymbol{h})$ is the empirical source risk, $R_t(\boldsymbol{h})$ is the target risk, $\Lambda = \min_{\boldsymbol{h}\in\mathcal{H}} R_s(\boldsymbol{h}) + R_*^t(\boldsymbol{h})$ and empirical open-set difference $\widehat{\Delta}_o = \frac{\widehat{R}_t^{u,K+1}(\boldsymbol{h})}{1-\pi_t^{K+1}} - \widehat{R}_s^{u,K+1}(\boldsymbol{h})$, here $R_s(\boldsymbol{h})$ are the risks defined in (5.2.2), $\widehat{R}_s^{u,K+1}(\boldsymbol{h})$, $\widehat{R}_t^{u,K+1}(\boldsymbol{h})$ are the empirical risks corresponding to $R_s^{u,K+1}(\boldsymbol{h})$, $R_t^{u,K+1}(\boldsymbol{h})$ defined in Eq. (5.2.6).

Proof. The proof is given in Appendix B.2. □

Next, we employ the open-set difference Δ_o to construct our algorithm — Distribution Alignment with Open Difference.

5.3 Proposed Algorithm

The importance of Theorem 5 is that it tells us the relationships between the three terms (i.e., the source risk, the distribution discrepancy, and the open-set difference) and the bound of the open-set domain adaptation. Inspired by these relationships, our initial focus is the following

optimization problem for unsupervised open-set domain adaptation:

$$\begin{aligned}\boldsymbol{h}^* = \underset{\boldsymbol{h}\in\mathcal{H}}{\arg\min}\ & \widehat{R}_s(\boldsymbol{h}) + \lambda d^{\ell}_{\mathcal{H}}(\widehat{P}_{X_s}, \widehat{P}_{X_t|\mathcal{Y}_s}) \\ & + \gamma\left(\frac{1}{1-\pi_t^{K+1}}\widehat{R}_t^{u,K+1}(\boldsymbol{h}) - \widehat{R}_s^{u,K+1}(\boldsymbol{h})\right),\end{aligned} \tag{5.3.1}$$

where λ and γ are two free hyper-parameters and the hypothesis space $\mathcal{H}$ is defined as a subset of the functional space $\{\boldsymbol{h} = [h_1, ..., h_{K+1}\ ]^T : h_c \in \mathcal{H}_k\}$, here $\mathcal{H}_k$ is the reproducing kernel Hilbert space (RKHS) with kernel k.

As proven by JDA [65], ARTL [6] and MEDA [96], incorporating conditional distributions into the original marginal distribution discrepancy can lead to superior domain adaptation performance. Hence, we have also added an additional conditional distribution discrepancy to the optimization problem in (5.3.1). Hence, the new problem becomes:

$$\begin{aligned}\boldsymbol{h}^* = \underset{\boldsymbol{h}\in\mathcal{H}}{\arg\min}\ & \widehat{R}_s(\boldsymbol{h}) + \lambda\mu D^2_{\boldsymbol{h},k}(\widehat{P}_{X_s}, \widehat{P}_{X_t|\mathcal{Y}_s}) \\ & + \lambda(1-\mu)\sum_{c=1}^{K} D^2_{\boldsymbol{h},k}(\widehat{P}_{X_s|\mathbf{y}_c}, \widehat{P}_{X_t|\mathbf{y}_c}) \\ & + \gamma\left(\frac{1}{1-\pi_t^{K+1}}\widehat{R}_t^{u,K+1}(\boldsymbol{h}) - \widehat{R}_s^{u,K+1}(\boldsymbol{h})\right),\end{aligned}$$

where $\mu \in [0, 1]$ is the adaptive factor [96] to convexly combine the contributions from both the empirical marginal distribution alignment and the empirical conditional distribution alignment. Note that the original $d^{\ell}_{\mathcal{H}}(\cdot,\cdot)$ is replaced with the projected MMD $D_{\boldsymbol{h},k}(\cdot,\cdot)$ in the new problem, because $d^{\ell}_{\mathcal{H}}(\cdot,\cdot)$ is difficult to estimate. In addition, we use the *squared loss* $\ell(\mathbf{y}, \mathbf{y}') = \|\mathbf{y} - \mathbf{y}'\|_2^2$ to design our algorithm.

Further, we have added the manifold regularization [121] to learn the geometric structure of the source and target domains. With this regularization, our algorithm can consistently achieve good performance when the setting degrades into a closed-set domain adaptation (i.e., where there are no unknown classes). This is because the state of the art closed-set algorithm ARTL [6] is a special case of DAOD, when there is no

open-set difference. Thus, the optimization problem can be rewritten as follows:

$$\begin{aligned}\boldsymbol{h}^* = \arg\min_{\boldsymbol{h}\in\mathcal{H}} \ & \widehat{R}_s(\boldsymbol{h}) + \lambda\mu D^2_{\boldsymbol{h},k}(\widehat{P}_{X_s}, \widehat{P}_{X_t|\mathcal{Y}_s}) \\ & + \lambda(1-\mu)\sum_{c=1}^{K} D^2_{\boldsymbol{h},k}(\widehat{P}_{X_s|\mathbf{y}_c}, \widehat{P}_{X_t|\mathbf{y}_c}) \\ & + \alpha\widehat{R}_t^{u,K+1}(\boldsymbol{h}) - \gamma\widehat{R}_s^{u,K+1}(\boldsymbol{h}) \\ & + \rho M_{\boldsymbol{h}}(\mathcal{S}_X, \mathcal{T}_u) + \sigma\|\boldsymbol{h}\|_k^2,\end{aligned} \tag{5.3.2}$$

where $\alpha = \gamma/(1-\pi^t_{K+1})$, ρ and σ are three free hyper-parameters, $\mathcal{T}_u$ denotes unlabeled target samples, $\mathcal{S}_X$ denotes source samples without labels, $M_{\boldsymbol{h}}(\mathcal{S}_X, \mathcal{T}_u)$ is the manifold regularization, and $\|\boldsymbol{h}\|_k^2$ is the squared norm of $\boldsymbol{h}$ in $\mathcal{H}_k$ to avoid over-fitting.

Next, we show how to formulate equation (5.3.2) using given samples. First, following the representer theorem, if the optimization problem (5.3.2) has a minimizer $\boldsymbol{h}^*$, then $\boldsymbol{h}^*$ can be written as

$$\boldsymbol{h}^*(\mathbf{x}) = \sum_{i=1}^{n_s+n_u} \boldsymbol{\beta}_i k(\mathbf{x}^i, \mathbf{x}), \quad \forall \mathbf{x} \in \mathcal{X},$$

where $\mathbf{x}^i \in \mathcal{S}_X \cup \mathcal{T}_u$ and $\boldsymbol{\beta}_i \in \mathbb{R}^{(K+1)\times 1}$ is the parameter. With this form of $\boldsymbol{h}^*$, we explain the computation of terms in Eq. (5.3.2).

5.3.1 Distribution Alignment

Since there are no labels for the target samples, we cannot directly compute

$$\mu D^2_{\boldsymbol{h},k}(\widehat{P}_{X_s}, \widehat{P}_{X_t|\mathcal{Y}_s}) + (1-\mu)\sum_{c=1}^{K} D^2_{\boldsymbol{h},k}(\widehat{P}_{X_s|\mathbf{y}_c}, \widehat{P}_{X_t|\mathbf{y}_c}) \tag{5.3.3}$$

Therefore, the pseudo target labels are used to help compute $\widehat{P}_{X_t|\mathcal{Y}_s}$ and $\widehat{P}_{X_t|\mathbf{y}_c}$ instead. Given the pseudo target samples for known classes $\mathcal{T}_{X,K}$, the pseudo target samples for c-th class $\mathcal{T}_{X,c}$ and the source samples for c-th class $\mathcal{S}_{X,c}$ we can compute (5.3.3) by the representer theorem and

kernel trick as follows:

$$\mathrm{tr}(\boldsymbol{\beta}^T \mathbf{K}\mathbf{M}\mathbf{K}\boldsymbol{\beta}), \tag{5.3.4}$$

where $\boldsymbol{\beta} = [\boldsymbol{\beta}_1, ..., \boldsymbol{\beta}_{n_s+n_u}]^T \in \mathbb{R}^{(K+1)\times(n_s+n_u)}$, $\mathbf{K}$ is the $(n_s+n_u) \times (n_s+n_u)$ kernel matrix $[k(\mathbf{x}^i, \mathbf{x}_j)]$, here $\mathbf{x}^i, \mathbf{x}^j \in \mathcal{S}_X \cup \mathcal{T}_u$, and $\mathbf{M} = \mu\mathbf{M}_0 + (1-\mu)\sum_{c=1}^{K} \mathbf{M}_c$ is the MMD matrix:

$$(\mathbf{M}_0)_{ij} = \begin{cases} \dfrac{1}{(n_s)^2}, & \mathbf{x}^i, \mathbf{x}^j \in \mathcal{S}_X, \\ \dfrac{1}{(n_u^K)^2}, & \mathbf{x}^i, \mathbf{x}_j \in \mathcal{T}_{X,K}, \\ 0, & \mathbf{x}^i \text{ or } \mathbf{x}_j \in \mathcal{T}_u \setminus \mathcal{T}_{X,K}, \\ -\dfrac{1}{n_s n_t^K}, & \text{otherwise;} \end{cases}$$

$$(\mathbf{M}_c)_{ij} = \begin{cases} \dfrac{1}{(n_s^c)^2}, & \mathbf{x}^i, \mathbf{x}^j \in \mathcal{S}_{X,c}, \\ \dfrac{1}{(n_u^c)^2}, & \mathbf{x}^i, \mathbf{x}_j \in \mathcal{T}_{X,c}, \\ -\dfrac{1}{n_s^c n_u^c}, & \mathbf{x}^i \in \mathcal{S}_{X,c}, \mathbf{x}_j \in \mathcal{T}_{X,c}, \\ -\dfrac{1}{n_s^c n_u^c}, & \mathbf{x}_j \in \mathcal{S}_{X,c}, \mathbf{x}^i \in \mathcal{T}_{X,c}, \\ 0, & \text{otherwise,} \end{cases}$$

where $n_s = |\mathcal{S}_X|$, $n_u^K = |\mathcal{T}_{X,K}|$, $n_s^c = |\mathcal{S}_{X,c}|$ and $n_u^c = |\mathcal{T}_{X,c}|$.

5.3.2 Manifold Regularization

The pair-wise affinity matrix is denoted as

$$\mathbf{W}_{ij} = \begin{cases} \mathrm{sim}(\mathbf{x}^i, \mathbf{x}^j), & \mathbf{x}^i \in \mathcal{N}_p(\mathbf{x}^j) \text{ or } \mathbf{x}_j \in \mathcal{N}_p(\mathbf{x}^i) \\ 0, & \text{otherwise;} \end{cases}$$

where $\mathrm{sim}(\mathbf{x}^i, \mathbf{x}^j)$ is the similarity function such as cosine similarity, $\mathcal{N}_p(\mathbf{x}^i)$ denotes the set of p-nearest neighbors to point $\mathbf{x}^i$ and p is a free parameter. The manifold regularization can then be evaluated as

follows:

$$M_{\boldsymbol{h}}(\mathcal{S}_X, \mathcal{T}_u) = \sum_{i,j=1}^{n_s+n_u} \|\boldsymbol{h}(\mathbf{x}^i) - \boldsymbol{h}(\mathbf{x}^j)\|_2^2 \mathbf{W}_{ij}$$
$$= \sum_{c=1}^{K+1} \sum_{i,j=1}^{n_s+n_u} h_c(\mathbf{x}^i) \mathbf{L}_{ij} h_c(\mathbf{x}^j),$$

where $\mathbf{x}_i, \mathbf{x}^j \in \mathcal{S}_X \cup \mathcal{T}_u$, $\mathbf{L}$ is the Laplacian matrix, which can be written as $\mathbf{D} - \mathbf{W}$, here $\mathbf{D}$ is a diagonal matrix and $\mathbf{D}_{ii} = \sum_{j=1}^{n_s+n_u} \mathbf{W}_{ij}$. Using the representer theorem and kernel trick, the manifold regularization $M_{\boldsymbol{h}}(\mathcal{S}_X, \mathcal{T}_u)$ can be written as

$$\mathrm{tr}(\boldsymbol{\beta}^T \mathbf{K} \mathbf{L} \mathbf{K} \boldsymbol{\beta}). \tag{5.3.5}$$

5.3.3 Open-Set Loss Function

We use a matrix to rewrite the loss function and open-set difference. Let the label matrix be $\mathbf{Y} \in \mathbb{R}^{(K+1)\times(n_s+n_u)}$:

$$\mathbf{Y}_{ij} = \begin{cases} 1, & \mathbf{x}^j \in \mathcal{S}_{X,i} \\ 0, & \text{otherwise} \end{cases} \quad \text{when} \quad i \leq K, \tag{5.3.6}$$

$$\mathbf{Y}_{ij} = \begin{cases} 1, & \mathbf{x}^j \in \mathcal{T}_{X,K+1} \\ 0, & \text{otherwise} \end{cases} \text{ when } \quad i = K+1. \tag{5.3.7}$$

The label matrix $\widetilde{\mathbf{Y}} \in \mathbb{R}^{(K+1)\times(n_s+n_u)}$ is

$$\widetilde{\mathbf{Y}}_{ij} = 1 \text{ iff } i = K+1 \text{ and } \mathbf{x}^j \in \mathcal{S}_X, \text{ otherwise } \widetilde{\mathbf{Y}}_{ij} = 0. \tag{5.3.8}$$

Then

$$\begin{aligned} &\widehat{R}_s(\boldsymbol{h}) + \alpha \widehat{R}_t^{u,K+1}(\boldsymbol{h}) - \gamma \widehat{R}_s^{u,K+1}(\boldsymbol{h}) + \sigma \|\boldsymbol{h}\|_k^2 \\ =&\|(\mathbf{Y} - \boldsymbol{\beta}^T \mathbf{K}) \mathbf{A}\|_F^2 - \gamma \|(\widetilde{\mathbf{Y}} - \boldsymbol{\beta}^T \mathbf{K}) \widetilde{\mathbf{A}}\|_F^2 + \sigma \mathrm{tr}(\boldsymbol{\beta}^T \mathbf{K} \boldsymbol{\beta}) \end{aligned} \tag{5.3.9}$$

where $\mathbf{A}$ is a $(n_s + n_u) \times (n_s + n_u)$ diagonal matrix with $\mathbf{A}_{ii} = \sqrt{\frac{1}{n_s}}$ if $\mathbf{x}^i \in \mathcal{S}_X$, $\mathbf{A}_{ii} = \sqrt{\frac{\alpha}{n_u}}$ if $\mathbf{x}^i \in \mathcal{T}_u$; $\widetilde{\mathbf{A}}$ is a $(n_s + n_u) \times (n_s + n_u)$ diagonal

matrix with $\widetilde{\mathbf{A}}_{ii} = \sqrt{\frac{1}{n_s}}$ if $\mathbf{x}^i \in \mathcal{S}_X$, $\widetilde{\mathbf{A}}_{ii} = 0$ if $\mathbf{x}^i \in \mathcal{T}_u$, and $\|\cdot\|_F$ is the Frobenius norm.

5.3.4 Overall Reformulation

Finally, based on Eqs. (5.3.4), (5.3.5) and (5.3.9), the optimization problem in (5.3.2) is reformulated as:

$$\boldsymbol{\beta}^* = \underset{\boldsymbol{\beta}\in\mathbb{R}^{(n_s+n_u)\times(K+1)}}{\arg\min} \mathcal{L}(\boldsymbol{\beta})$$

where

$$\begin{aligned}\mathcal{L}(\boldsymbol{\beta}) =& \|(\mathbf{Y} - \boldsymbol{\beta}^T\mathbf{K})\mathbf{A}\|_F^2 - \gamma\|(\widetilde{\mathbf{Y}} - \boldsymbol{\beta}^T\mathbf{K})\widetilde{\mathbf{A}}\|_F^2 \\ &+ \mathrm{tr}(\boldsymbol{\beta}^T\mathbf{K}(\lambda\mathbf{M} + \rho\mathbf{L})\mathbf{K}\boldsymbol{\beta}) + \sigma\mathrm{tr}(\boldsymbol{\beta}^T\mathbf{K}\boldsymbol{\beta}).\end{aligned} \tag{5.3.10}$$

5.3.5 Training

There is a negative term in $\mathcal{L}(\boldsymbol{\beta})$, hence it may be not correct to compute the optimizer by solving the equation $\frac{\partial\mathcal{L}(\boldsymbol{\beta})}{\partial\boldsymbol{\beta}} = \mathbf{0}$ directly. Maybe the "minimizer" solved by $\frac{\partial\mathcal{L}(\boldsymbol{\beta})}{\partial\boldsymbol{\beta}} = \mathbf{0}$ is a maximum point or a saddle point. Fortunately, the following theorem shows that there exists a unique optimizer that can be solved by $\frac{\partial\mathcal{L}(\boldsymbol{\beta})}{\partial\boldsymbol{\beta}} = \mathbf{0}$.

Theorem 6. *If the coefficient γ of $\widehat{R}_s^{u,K+1}(\boldsymbol{h})$ is smaller than 1 and the kernel k is universal, then the $\mathcal{L}(\boldsymbol{\beta})$ defined in Eq. (5.3.10) has a unique minimizer, which can be written as:*

$$\left((\mathbf{A}^2 - \gamma\widetilde{\mathbf{A}}^2 + \lambda\mathbf{M} + \rho\mathbf{L})\mathbf{K} + \sigma\mathbf{I}\right)^{-1}(\mathbf{A}^2\mathbf{Y}^T - \gamma\widetilde{\mathbf{A}}^2\widetilde{\mathbf{Y}}^T). \tag{5.3.11}$$

Proof. The proof of Theorem 6 is divided into three parts.

Claim 1:

$$\lim_{\|\boldsymbol{\beta}\|_{\ell^2}\to+\infty} \mathcal{L}(\boldsymbol{\beta}) = +\infty.$$

It is easy to check that

$$tr(\boldsymbol{\beta}^T\mathbf{K}(\lambda\mathbf{M} + \rho\mathbf{L})\mathbf{K}\boldsymbol{\beta}) + \sigma tr(\boldsymbol{\beta}^T\mathbf{K}\boldsymbol{\beta}) \geq 0,$$

since the projected MMD distance and manifold regularization are not negative. Consider $\|(\mathbf{Y}-\boldsymbol{\beta}^T\mathbf{K})\mathbf{A}\|_F^2-\gamma\|(\widetilde{\mathbf{Y}}-\boldsymbol{\beta}^T\mathbf{K})\widetilde{\mathbf{A}}\|_F^2$. Since the kernel k is universal and $\gamma<1$, the matrix $\mathbf{K}(\mathbf{A^2}-\gamma\widetilde{\mathbf{A}}^{\mathbf{2}})\mathbf{K}$ is symmetric and positive definite, which can be written as

$$\mathbf{K}(\mathbf{A^2}-\gamma\widetilde{\mathbf{A}}^{\mathbf{2}})\mathbf{K}=\mathbf{O}\boldsymbol{\Lambda}\mathbf{O}^{\mathbf{T}},$$

where $\mathbf{O}$ is the orthogonal matrix, and $\boldsymbol{\Lambda}$ is the diagonal matrix, whose diagonal elements $\{\lambda_i\}_{i=1}^{n^s+n^t}$ are positive.

Then,

$$\begin{aligned}
&\|(\mathbf{Y}-\boldsymbol{\beta}^T\mathbf{K})\mathbf{A}\|_F^2-\gamma\|(\widetilde{\mathbf{Y}}-\boldsymbol{\beta}^T\mathbf{K})\widetilde{\mathbf{A}}\|_F^2\\
&=tr[(\mathbf{Y}-\boldsymbol{\beta}^{\mathbf{T}}\mathbf{K})\mathbf{A^2}(\mathbf{Y}-\boldsymbol{\beta}^{\mathbf{T}}\mathbf{K})^{\mathbf{T}}]-\gamma\mathbf{tr}[(\widetilde{\mathbf{Y}}-\boldsymbol{\beta}^{\mathbf{T}}\mathbf{K})\widetilde{\mathbf{A}}^{\mathbf{2}}(\widetilde{\mathbf{Y}}-\boldsymbol{\beta}^{\mathbf{T}}\mathbf{K})^{\mathbf{T}}]\\
&=tr[\boldsymbol{\beta}^{\mathbf{T}}\mathbf{K}(\mathbf{A^2}-\gamma\widetilde{\mathbf{A}}^{\mathbf{2}})\mathbf{K}\boldsymbol{\beta}]-2tr[(\mathbf{Y}\mathbf{A^2}\mathbf{K}-\gamma\widetilde{\mathbf{Y}}\widetilde{\mathbf{A}}^{\mathbf{2}}\mathbf{K})\boldsymbol{\beta}]+constant\\
&=tr[\boldsymbol{\beta}^{\mathbf{T}}\mathbf{O}\boldsymbol{\Lambda}\mathbf{O}^{\mathbf{T}}\boldsymbol{\beta}]-2tr[(\mathbf{Y}\mathbf{A^2}\mathbf{K}-\gamma\widetilde{\mathbf{Y}}\widetilde{\mathbf{A}}^{\mathbf{2}}\mathbf{K})\boldsymbol{\beta}]+constant\\
&=tr[(\boldsymbol{\beta}^{\mathbf{T}}\mathbf{O})\boldsymbol{\Lambda}(\boldsymbol{\beta}^{\mathbf{T}}\mathbf{O})^{\mathbf{T}}]-2tr[(\mathbf{Y}\mathbf{A^2}\mathbf{K}-\gamma\widetilde{\mathbf{Y}}\widetilde{\mathbf{A}}^{\mathbf{2}}\mathbf{K})\boldsymbol{\beta}]+constant\\
&\geq ctr[(\boldsymbol{\beta}^{\mathbf{T}}\mathbf{O})\mathbf{I}(\boldsymbol{\beta}^{\mathbf{T}}\mathbf{O})^{\mathbf{T}}]-O(\|\boldsymbol{\beta}\|_{\ell^2})+constant\\
&=ctr[\boldsymbol{\beta}^{\mathbf{T}}\boldsymbol{\beta}]-O(\|\boldsymbol{\beta}\|_{\ell^2})+constant,
\end{aligned}$$

where c is the smallest diagonal element of the diagonal matrix $\boldsymbol{\Lambda}$.

Therefore,

$$\begin{aligned}
\lim_{\|\boldsymbol{\beta}\|_{\ell^2}\to+\infty}\mathcal{L}(\boldsymbol{\beta})&\geq\lim_{\|\boldsymbol{\beta}\|_{\ell^2}\to+\infty}ctr\boldsymbol{\beta}^{\mathbf{T}}\boldsymbol{\beta}-O(\|\boldsymbol{\beta}\|_{\ell^2})+constant\\
&=\lim_{\|\boldsymbol{\beta}\|_{\ell^2}\to+\infty}c\|\boldsymbol{\beta}\|_{\ell^2}^2-O(\|\boldsymbol{\beta}\|_{\ell^2})+constant=+\infty
\end{aligned}$$

Claim 2: There exist optimizers.

In Claim 1, we have proven that

$$\lim_{\|\boldsymbol{\beta}\|_{\ell^2}\to+\infty}\mathcal{L}(\boldsymbol{\beta})=+\infty,$$

which implies that there exists a large constant $r>0$, such that $\mathcal{L}(\boldsymbol{\beta})>\mathcal{L}(\mathbf{0})$, for any $\boldsymbol{\beta}\in\mathbb{R}^{(K+1)\times(n^s+n^t)}\setminus B_r(\mathbf{0})$, where $B_r(\mathbf{0})$ is an open ball with a radius pf r and a center of $\mathbf{0}$. Since $\mathcal{L}$ is a continuous function and the

Algorithm 3 DAOD

Input: Data $\mathcal{S}, \mathcal{T}_u$; #iterations T; #neighbor p and parameters $\lambda, \sigma, \rho, \alpha, \gamma, \mu$; threshold t; universal kernel function $k(\cdot, \cdot)$.

1. $\widetilde{Y}_t \leftarrow \text{OSNN}^{\text{cv}}(\mathcal{S}, \mathcal{T}_u, t)$;% Predict pseudo labels;
2. Compute $\mathbf{L}, \mathbf{K}$ using $\mathcal{S}$, $\mathcal{T}_u$, and $\widetilde{Y}_t$;
3. $i \leftarrow 1$

While $i < T + 1$

4. Compute $\mathbf{M}$ using $\mathcal{S}$, $\mathcal{T}_u$, and $\widetilde{Y}_t$;
5. Compute $\boldsymbol{\beta}$ by formula (5.3.11)
6. $\widetilde{Y}_t \leftarrow \boldsymbol{\beta}^T \mathbf{K}$;%Predict pseudo labels
7. $i \leftarrow i + 1$

Output: Predicted target labels $\widetilde{Y}_t$, classifier $\boldsymbol{\beta}^T \mathbf{K}$.

closed ball $\overline{B}_r(\mathbf{0})$ is a compact set, we know that there exist optimizers for $\mathcal{L}$ and these points must be contained in the open ball $B_r(\mathbf{0})$.

Claim 3: The solution is unique.

If a point $\boldsymbol{\beta}_0$ is a minimizer, then $\boldsymbol{\beta}_0$ satisfies the following equation:

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\beta}}(\boldsymbol{\beta}_0) = \mathbf{0}.$$

If the solution of $\frac{\partial \mathcal{L}}{\partial \boldsymbol{\beta}}(\boldsymbol{\beta}_0) = \mathbf{0}$ is unique, then the solution must be the unique minimizer.

We compute $\frac{\partial \mathcal{L}}{\partial \boldsymbol{\beta}}(\boldsymbol{\beta}_0) = \mathbf{0}$,

$$\begin{aligned} 0 = &-2(\mathbf{K}\mathbf{A}^2\mathbf{Y}^T - \gamma \mathbf{K}\widetilde{\mathbf{A}}^2\widetilde{\mathbf{Y}}^T) + 2\sigma \mathbf{K}\boldsymbol{\beta} \\ &+ 2(\mathbf{K}\mathbf{A}^2\mathbf{K} - \gamma \mathbf{K}\widetilde{\mathbf{A}}^2\mathbf{K})\boldsymbol{\beta} + \mathbf{2K}(\lambda \mathbf{M} + \rho \mathbf{L})\mathbf{K}\boldsymbol{\beta}. \end{aligned} \tag{5.3.12}$$

Note that the solution of Eq. (5.3.12) is unique and can be written as:

$$\boldsymbol{\beta} = \left((\mathbf{A}^2 - \gamma \widetilde{\mathbf{A}}^2 + \lambda \mathbf{M} + \rho \mathbf{L})\mathbf{K} + \sigma \mathbf{I}\right)^{-1} (\mathbf{A}^2 \mathbf{Y}^T - \gamma \widetilde{\mathbf{A}}^2 \widetilde{\mathbf{Y}}^T).$$

With Claim 1, Claim 2 and Claim 3, Theorem 6 is proven. □

To compute the true value of Eq. (5.3.11), it is best to use the groundtruth labels of the target domain. However, our focus is on unsupervised task, which means that it is impossible to obtain any true target labels and, as mentioned, pseudo labels can be used instead. These pseudo labels are generated by applying an open-set classifier that has

been trained on the source samples to the target samples.

In this chapter, we used *open-set Nearest Neighbor* (OSNN^{cv}-t) [162] to help us learn the pseudo labels. We select the two nearest neighbors $\mathbf{v}, \mathbf{u}$ from the test sample $\mathbf{s}$. If both nearest neighbors have the same label $\mathbf{y}_c$, $\mathbf{s}$ is classified with the label $\mathbf{y}_c$. Otherwise, the following ratio is calculated $\|\mathbf{v}-\mathbf{s}\|_2/\|\mathbf{u}-\mathbf{s}\|_2$, on the assumption that $\|\mathbf{v}-\mathbf{s}\|_2 \leq \|\mathbf{u}-\mathbf{s}\|_2$. If the ratio is smaller than or equal to a pre-defined threshold t, $0 < t < 1$, $\mathbf{s}$ is classified with the same label as $\mathbf{v}$. Otherwise, $\mathbf{s}$ is recognized as the unknown sample.

To make the pseudo labels more accurate, we use the iterative pseudo label refinement strategy, proposed by JDA [65]. The implementation details are demonstrated in algorithm 3.

5.4 Experiments and Evaluations

In this section, we first utilize real world datasets to verify the performance of DAOD. We then conduct experiments to examine the behavior of the parameters.

5.4.1 Real World Datasets

We evaluate our algorithm on three cross-domain recognition tasks: object recognition (**Office-31**, **Office-Home**), and face recognition (**PIE**). Table 5.1 lists the statistics of these datasets.

Table 5.1: Introduction of datasets.

Dataset	Type	#Sample	#Feature	#Class	Domain
Office-31	Object	4,110	4,096	31	A,W,D
Office-Home	Object	15,500	2,048	65	Ar,Cl,Pr,Rw
PIE	Face	1,1554	1,024	68	P1,...,P5

Office-31 [128] consists of three real-world object domains: AMAZON (**A**), DSLR (**D**) and WEBCAM (**W**). It has 4,652 images with 31 common categories. This means that there are 6 domain adaptation

tasks: **A** $\rightarrow$ **D**, **A** $\rightarrow$ **W**, **D** $\rightarrow$ **A**, **W** $\rightarrow$ **A**, **D** $\rightarrow$ **W**, **W** $\rightarrow$ **D**. Following the standard protocol and for a fair comparison with the other algorithms, we extracted feature vectors from the fully connected layer-7 (fc7) of the AlexNet [163]. We introduced an open set protocol for this dataset by taking classes 1-10 as the shared classes in alphabetical order. The classes 21-31 are used as the unknown classes in the target domain.

Office-Home [129] consists of 4 different domains: Artistic (**Ar**), Clipart (**Cl**), Product (**Pr**) and Real-World (**Rw**). Each domain contains images from 65 object classes. We constructed 12 OSDA tasks: **Ar** $\rightarrow$ **Cl**, **Ar** $\rightarrow$ **Pr**,..., **Rw** $\rightarrow$ **Ar**. In alphabetical order, we used the first 25 classes as the known classes and classes 26-65 as the unknown classes. Following the standard protocol and for a fair comparison with the other algorithms, we extracted feature vectors from ResNet-50.

PIE [164] contains 41, 368 facial images of 68 people in various poses, illuminations, and expression changes. The face images are captured by 13 synchronized cameras (different poses) and 21 flashes (different illuminations and/or expressions). We focused on 5 of 13 poses, i.e., **PIE1** (C05, left pose), **PIE2** (C07, upward pose), **PIE3** (C09, downward pose), **PIE4** (C27, frontal pose) and **PIE5** (C29, right pose). These facial images were cropped to a size of 32×32. We took classes 1-20 as the known classes and classes 21-68 as the unknown classes in the target domain. 20 tasks were tested: **PIE1**$\rightarrow$**PIE2**, **PIE1**$\rightarrow$**PIE3**,..., **PIE5**$\rightarrow$**PIE4**.

5.4.2 Baseline Algorithms

The baseline algorithms selected for comparison with DAOD are:

No Transfer:

• OSNN [162]. OSNN recognizes a sample as unknown by computing the ratio of similarity scores to the two most similar classes of the sample and then comparing the ratio with a pre-defined threshold.

Closed-Set:

• TCA [5] + OSNN. The aim in implementing TCA is to show that if

the UCSDA algorithm is used to solve the UOSDA problem, negative transfer will occur, leading to poor performance.

Open-Set:

• JDA [65] + OSNN. We extended JDA into the open-set setting. Joint distribution matching is the main step in JDA. Thus, we simply matched the known samples predicted by OSNN when the JDA algorithm was implemented.

• JGSA [35] + OSNN. We extended JGSA into the open-set setting. First, for learning new features, we implemented JGSA using the source samples and the known target samples predicted by OSNN. Then, we used OSNN to predict the pseudo labels. We repeated the process until convergence.

• ATI [27] + OSNN. ATI was the first UOSDA algorithm, but it requires the unknown source samples to implement. Therefore, to implement ATI under our setting, we used ATI to select the outliers, and then learned the new features for matching the source domain and target domain excluding selected outliers. Lastly, OSNN was used to predict the labels.

• OSBP [28]. OSBP utilizes adversarial neural networks and a binary cross entropy loss to learn the probability for the target samples, then uses the estimated probability to recognize the unknown samples.

5.4.3 Experimental Setup

Before reporting the detailed evaluation results, it is important to explain how DAOD's hyper-parameters are tuned. DAOD has several hyper-parameters: 1) the choice of the kernel function k; 2) the adaptation parameters $\lambda, \sigma, \rho, p, \mu$; 3) the open-set parameters α, γ; and 4) #iterations T and threshold $t \in (0, 1)$. Each parameter is discussed one by one next.

5.4.3.1 The kernel function k

As suggested in [62, 96], we use the Gaussian kernel

$$k(\mathbf{a}, \mathbf{b}) = \exp(-\frac{\|\mathbf{a} - \mathbf{b}\|_2^2}{2r^2}), \tag{5.4.1}$$

where the kernel bandwidth r is median($\|\mathbf{a}-\mathbf{b}\|_2$), $\forall \mathbf{a}, \mathbf{b} \in \mathcal{S}_X \cup \mathcal{T}_u$.

5.4.3.2 The adaptive factor μ

The adaptive factor μ expresses the relative importance of the marginal distributions and conditional distributions. Wang et al. [96] made the first attempt to compute μ by employing $\mathcal{A}$-distance [17], which is the special case $d_{\mathcal{H}}^{0-1}$ of the discrepancy distance $d_{\mathcal{H}}^{\ell}$. According to [17], the $\mathcal{A}$-distance can also be defined as the error of building a binary classifier from hypothesis set $\mathcal{H}$ to discriminate between the two domains. Wang et al. [96] used the linear hypothesis set to estimate $\mathcal{A}$-distance. Let $\epsilon(\boldsymbol{h})$ be the error of the linear classifier $\boldsymbol{h}$ discriminating source samples $\mathcal{S}_X$ and target samples $\mathcal{T}_u$. Then, the $\mathcal{A}$-distance

$$d_{\mathcal{A}}(\mathcal{S}_X, \mathcal{T}_u) = 2(1 - \epsilon(\boldsymbol{h})).$$

We use the same algorithm as [96] to estimate μ by

$$\mu = 1 - \frac{d_0}{d_0 + \sum_{c=1}^{K} d_c},$$

where $d_0 = d_{\mathcal{A}}(\mathcal{S}_X, \mathcal{T}_{X,K})$, $d_c = d_{\mathcal{A}}(\mathcal{S}_{X,c}, \mathcal{T}_{X,c})$ $(c = 1, ..., C)$. Here $\mathcal{T}_{X,K}$ is the set of the target samples predicted as known samples. This estimation has to be computed at every iteration of DAOD, since the predicted conditional distributions for the target may vary each time.

5.4.3.3 The open-set parameters α and γ

As shown in Fig. 5.3, DAOD is able to achieve consistently good performance within a same range $\alpha \in [0.2, 0.4]$ and $\gamma \in [0.15, 0.5]$, which shows the relative stability of DAOD given the correct tuning of these two parameters. Tuning should be done according to the following rules. First, the positive term $R_t^{u,K+1}$ and the negative term $R_s^{u,K+1}$ in the open-set difference are inferred from each other. A larger positive term means that more samples are recognized as the unknown classes. A larger negative term implies that more samples are classified as known classes. To

ensure that the positive and negative terms balance, the difference $|\alpha-\gamma|$ should not be too large. Further, the parameter α should be larger than γ, since the positive term's coefficient $1/(1-\pi_t^{K+1})$ is larger than 1. In this chapter, we set $\alpha = 0.4$ for all tasks and 1) $\gamma = 0.2$ for **Office-31**, and 2) $\gamma = 0.25$ for **Office-Home** and **PIE** datasets.

5.4.3.4 Other hyper-parameters

We run DAOD with a wide range of parameter values for λ, ρ, p, σ, t and T in subsection 5.4.6. The results are shown in Fig. 5.3. These results indicate that DAOD can provide a robust performance with a wide range of hyper-parameter values.

From our tests, the best choices of parameters are: $\lambda \in [50, 1000]$, $\rho \in [0, 1]$, $p \in [2, 32]$, $\sigma \in [0.2, 1.6]$, $t \in [0, 0.9]$ and DAOD can converge within 10 iterations. To sum up, the performance of DAOD stays robust with a large range of parameter choice. Therefore, the parameters do not need to be significantly fine-tuned in practical applications. In this chapter, we fixed $p = 10$, $\rho = 1$ and $\sigma = 1$, $T = 10$, $t = 0.5$ and set 1) $\lambda = 50$ for **Office-31**, and 2) $\lambda = 500$ for **Office-Home** and **PIE** datasets.

Although DAOD is easy to use, and its parameters do not have to be fine-tuned, we did explore how to further tune these parameters for research purposes. We chose the parameters according to following the rules: 1) The regularization term $\|\boldsymbol{h}\|_k^2$ is very important, so we tended to choose a slightly larger σ ($\sigma = 1$) to prevent DAOD from degenerating. 2) We chose ρ by following [121]. 3) p is set following [121, 165]. 4) distribution alignment is inevitable for DAOD, so we chose a larger λ ($\lambda \geq 50$) to make it count.

We use two types of accuracy [27, 28] to evaluate DAOD:

$$\text{Acc(OS)} = \frac{1}{K+1}\sum_{c=1}^{K+1}\frac{|x : x \text{ from class } c \bigwedge \widetilde{f}(\mathbf{x}) = c|}{|\mathbf{x} : \mathbf{x} \text{ from class } c|}, \tag{5.4.2}$$

and

$$\text{Acc(OS}^*) = \frac{1}{K}\sum_{c=1}^{K}\frac{|\mathbf{x} : \mathbf{x} \text{ from class } c \bigwedge \widetilde{f}(\mathbf{x}) = c|}{|\mathbf{x} : \mathbf{x} \text{ from class } c|}, \tag{5.4.3}$$

where $\widetilde{f}$ is the predicted classifier. Note that Acc(OS) is the main index for evaluating the performance of the UOSDA algorithms [27].

5.4.4 Experimental Results

The classification accuracy of the UOSDA tasks is shown in Table 5.2. The following facts can be observed from this table. 1) The closed-set algorithm TCA performs poorly on most tasks, even worse than the standard OSNN algorithm, indicating that negative transfer occurred. 2) All open-set algorithms achieve better classification accuracy than OSNN on most tasks. This is because the source samples and the known target samples have different distributions. 3) DAOD achieves much better performance Acc(OS) than the six baseline algorithms on most tasks (24 out of 38). The average classification accuracy $(\text{Acc(OS)}, \text{Acc(OS}^*))$ of DAOD on the 38 tasks is 69.3%, 70.4% respectively, gaining a performance improvement of 3.4%, 3.6% compared to the best baseline OSBP. 4) Generally, JDA+OSNN, JGSA+OSNN and ATI+OSNN algorithms do not perform as well as DAOD. A major limitation of these algorithms may be that they omit the selected unknown target samples when they construct a latent space to match the distributions for the known classes. This may result in the unknown samples being mixed with the known samples in the latent space. In DAOD, the negative term $R_s^{u,K+1}$ helps DAOD to avoid the problem suffered by JDA, JGSA and ATI. 5) The performance of the OSPB algorithm was generally worse than that of DAOD. The main reasons may be that: 1) OSBP only matches the marginal distributions, not the joint distributions; 2) OSBP does not keep the unknown target samples away from the known source samples, with the result that many unknown target samples are recognized as known samples. DAOD, however, uses the negative term $R_s^{u,K+1}$ to sep-

Table 5.2: Acc(OS*) and Acc(OS) (%) on Office-31, Office-Home and PIE Datasets.

Dataset	OSNN		TCA		JDA		JGSA		ATI		OSBP		DAOD	
	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS
A→W	56.0	54.0	44.8	54.8	63.0	64.8	75.7	75.2	70.6	69.7	69.1	70.1	**84.2**	**84.2**
A→D	75.4	71.9	68.0	67.1	70.1	70.6	74.8	73.3	85.9	84.0	76.4	76.6	**89.8**	**88.5**
D→A	62.6	60.3	53.4	52.7	60.4	60.7	62.4	61.5	68.3	67.6	62.3	62.5	**71.8**	**72.6**
D→W	93.0	88.1	84.6	80.9	98.4	94.7	98.0	93.2	95.8	94.1	94.6	**98.9**	**98.0**	96.0
W→A	58.6	56.8	56.1	55.6	62.5	62.6	64.0	62.9	64.0	62.8	**82.2**	**82.3**	72.9	74.2
W→D	99.3	93.3	97.8	94.8	99.3	96.1	**100.0**	94.4	97.8	94.5	96.8	**96.9**	97.5	96.3
Avg	74.2	70.7	69.2	68.5	75.6	74.9	79.2	76.7	80.4	78.8	80.2	80.4	**85.7**	**85.3**
Ar→Pr	39.4	40.6	37.7	37.9	59.7	59.0	64.1	63.3	70.4	68.6	69.2	68.4	**72.6**	**71.8**
Ar→Cl	32.1	33.7	24.4	24.1	39.1	39.6	45.9	46.0	54.2	53.1	53.3	53.1	**55.3**	**55.4**
Ar→Rw	56.6	57.0	55.7	55.3	67.5	66.4	74.1	72.8	78.1	77.3	**79.1**	**78.0**	78.2	77.6
Cl→Ar	32.3	34.0	31.3	32.1	41.9	42.1	43.8	44.5	59.1	57.8	58.2	57.9	**59.1**	**59.2**
Cl→Pr	39.1	40.3	34.8	34.8	49.1	48.9	55.8	55.8	68.3	66.7	**72.4**	**71.6**	70.8	70.1
Cl→Rw	46.9	47.7	41.4	41.2	59.7	59.1	62.8	62.5	75.3	74.3	72.3	71.4	**77.8**	**77.0**
Rw→Ar	51.4	52.1	49.4	49.2	55.8	55.1	56.9	56.4	70.8	70.0	68.2	66.5	**71.3**	**70.5**
Rw→Cl	38.0	39.2	34.9	34.1	44.1	43.9	48.7	48.6	55.4	55.2	**59.2**	**57.8**	58.4	**57.8**
Rw→Pr	59.2	59.2	57.3	56.5	68.0	68.2	66.5	65.3	79.4	78.3	80.8	78.6	**81.8**	**80.6**
Pr→Ar	38.5	39.7	33.2	33.4	48.4	48.0	55.8	55.5	62.6	61.2	61.0	59.6	**66.7**	**65.8**
Pr→Cl	35.0	36.3	35.8	36.1	41.2	41.1	44.1	44.4	54.1	53.9	56.9	55.7	**60.0**	**59.1**
Pr→Rw	59.6	59.7	58.3	57.5	70.4	68.9	73.5	72.3	81.1	79.9	83.9	82.1	**84.1**	**82.2**
Avg	44.0	45.0	41.2	41.0	53.8	53.4	57.7	57.3	67.4	66.4	67.9	66.7	**69.6**	**68.9**
P1→P2	32.1	34.3	20.6	21.4	42.1	41.3	55.4	54.4	44.0	41.9	**66.6**	**64.2**	57.3	56.5
P1→P3	46.5	48.3	20.2	20.3	50.0	49.1	54.4	53.5	56.3	53.6	**69.1**	**66.4**	53.1	52.2
P1→P4	60.1	61.2	30.7	30.5	62.3	61.2	63.2	61.8	67.9	64.6	80.0	76.2	**85.2**	**82.4**
P1→P5	22.9	26.1	10.6	11.5	28.3	28.2	35.8	35.7	45.4	43.3	**50.2**	**49.1**	47.3	46.1
P2→P1	35.6	37.9	25.4	25.5	47.9	47.3	68.5	67.2	59.5	56.7	54.2	52.9	**69.7**	**68.1**
P2→P3	61.5	62.5	38.8	38.3	62.9	61.4	62.5	61.3	56.3	53.6	63.5	61.5	**71.7**	**69.9**
P2→P4	71.0	71.4	49.3	48.5	71.6	69.6	78.6	76.9	77.1	73.5	81.3	87.6	**91.2**	**88.2**
P2→P5	28.5	31.2	20.4	20.7	37.3	37.1	49.0	48.0	36.7	34.9	44.2	41.2	**49.8**	**49.4**
P3→P1	43.3	45.2	20.1	20.4	51.1	50.6	66.9	65.5	**68.4**	**66.9**	61.0	61.3	**68.3**	66.6
P3→P2	53.5	54.8	37.3	36.5	64.2	62.5	66.9	65.2	55.0	52.4	64.6	64.1	**70.4**	**68.5**
P3→P4	64.9	65.4	34.6	34.2	68.5	66.6	75.6	73.8	74.0	70.5	76.9	74.7	**87.1**	**83.9**
P3→P5	34.6	37.0	12.7	13.0	39.2	39.0	42.5	41.8	47.1	44.8	46.7	46.3	**53.3**	**52.3**
P4→P1	56.5	57.7	24.8	24.6	64.2	62.4	75.8	73.9	66.8	63.7	68.7	67.2	**87.1**	**84.4**
P4→P2	78.1	78.0	64.0	62.1	75.2	72.4	78.3	76.1	78.1	74.4	**85.0**	82.2	84.8	**82.4**
P4→P3	78.3	78.3	33.8	33.3	81.5	78.9	**81.3**	**79.1**	61.7	58.7	67.6	66.9	80.0	77.6
P4→P5	43.1	44.8	17.1	17.7	52.1	50.9	**65.8**	**64.4**	48.5	46.2	63.8	59.9	61.3	59.9
P5→P1	23.2	25.7	11.6	12.8	29.6	30.2	46.4	45.9	23.5	30.2	**66.6**	**64.2**	60.6	59.2
P5→P2	26.5	28.4	18.3	18.3	31.0	31.1	**44.0**	**43.6**	36.7	34.9	35.8	35.4	34.8	35.0
P5→P3	31.0	32.7	12.3	13.3	33.1	32.9	**55.4**	**54.6**	41.9	39.9	46.3	45.1	44.4	44.6
P5→P4	37.2	38.9	19.4	20.0	49.7	49.1	63.8	62.7	58.6	55.8	53.5	52.2	**70.3**	**68.6**
Avg	46.4	48.0	26.2	26.1	52.1	51.1	61.5	60.3	55.2	53.0	62.2	61.0	**66.4**	**64.8**
All Avg	50.0	50.6	37.7	37.5	56.3	55.6	63.1	61.9	63.0	61.3	66.8	65.9	**70.4**	**69.3**

arate the source samples and unknown target samples.

5.4.5 Open-Set Parameters Analysis

From our analysis of the open-set parameters α and γ, we find that the relationship between α and γ is closely related to another parameter, the difference $\delta = \alpha - \gamma$. We conducte experiments on the **Office-31** dataset with α ranging from 0.2 to 1.2 and δ ranging from -0.2 to α. Due to space limitations, the average results on **Office-31** are reported in Fig. 5.2. According to Fig. 5.2, we made the following observations:

1) As δ increased, the accuracy of the unknown classes also increased,

since a larger positive term $R_t^{u,K+1}(\boldsymbol{h})$ means that more samples are recognized as unknown.

2) When $\delta < 0$ ($\alpha < \gamma$), for almost all $\alpha \in [0.2, 1.2]$, the performance Acc(OS) was poorer than the best baseline algorithm (dashed line). This is because when $\delta < 0$, more samples are recognized as known classes. Our theoretical results support this observation (Theorem 5) since the positive term's coefficient $1/(1-\pi_t^{K+1})$ is larger than the negative term's coefficient 1. Thus, δ should be larger than 0 ($\alpha > \gamma$).

3) All figures in Fig. 5.2 are similar for almost all α from 0.4 to 1.2, which implies that α may be not the most important factor influencing the performance of DAOD. Rather, the difference δ is likely to be more important.

4) Performance Acc(OS) begins to decrease when δ is larger than 0.25 because more known samples are classified as unknown with a larger δ.

5) When α ranged between 0.2 to 1.2 and δ was chosen from $[0.05, 0.2]$, the performance Acc(OS) of DAOD δ was superior to the best baseline.

6) Although α is not the main factor influencing the performance of DAOD, we compare figures ($\alpha < 1.0$) with figures ($\alpha \geq 1.0$) and find that a smaller α achieves slightly better performance than a larger α. In general, we select α from $[0.2, 0.4]$ and δ from $[0.05, 0.25]$.

5.4.6 Parameter Sensitivity, Ablation Study, Convergence Analysis.

These studies are conducted on different types of datasets to demonstrate that: 1) a wide range of parameter values can be chosen to obtain satisfactory performance, and 2) the open-set difference and distribution alignment term are important and necessary. We evaluated important parameters $\lambda, \sigma, \rho,\ p,\ t,\ \alpha,\ \gamma$ and T, reporting the average results for datasets **Office-31**, **Office-Home** and **PIE** respectively. The dashed line denotes the results of the best baseline algorithm with each dataset.

Distribution Alignment λ. We run DAOD with varying values of λ. Fig. 5.3(a) plots the classification accuracy w.r.t. to different values of λ.

From this figure, we can see that: 1) When $\lambda = 0$, the performance was the worst than the baseline. 2) After the increasing of the λ from 0 to 50, the performance dramatically increased to equal that of the baseline. 3) From 50 to 1000, DAOD was stable with values of around $0.85, 0.7$, and 0.65 on the three datasets. Overall, the performance of DAOD with most values of λ was better than the baselines. We also found that larger values of λ resulted in a better distribution alignment, and, if we chose λ from $[50, 1000]$, we obtained better results than the best baseline algorithm.

Manifold Regularization ρ. In these experiments, we run DAOD with varying values of ρ. Larger values of ρ increase the importance of manifold consistency in DAOD. From Fig. 5.3(b), we can see that: 1) DAOD's performance was steady and consistently good when $\rho \in [0, 1]$. 2) But, after the increasing of ρ from 1 to 5, its performance dramatically dropped below the baseline. 3) Further, DAOD's continued to fall below the baseline from 5 to 50. The reason for this poor performance at $\rho \in [5, 50]$ is that when ρ is large, DAOD mainly focuses on the geometric information of the samples and ignores other information. Choosing λ from $[0, 1]$, however, provides the best results.

#Nearest Neighbors p. We run DAOD with varying values of p. If $p \to +\infty$, two samples which are not at all similar are connected. If $p \to 0$, limited similarity information between samples is captured, thus p should not be too large or too small. Fig. 5.3(c) shows that if p is selected from $[2, 32]$, the performance of our algorithm is better than the baseline. When $p > 32$, the performance of **PIE** was worse than the baseline. One reason may be that when p is large, the samples from different classes are connected, resulting in that samples from different classes share similar scores. From Fig. 5.3(c), p can be selected from $[2, 32]$.

Regularization σ. We run DAOD with varying values of σ and plotted the classification accuracy as shown in Fig. 5.3(d). Theoretically, when $\sigma \to 0$, the classifier degenerates and overfitting occurs. When $\sigma \to +\infty$,

the classifier obtains a trivial result. From Fig. 5.3(d), we can see that: 1) When $\sigma = 0$, the performance was the worst and also much worse than the baseline. 2) However, after increasing of σ from 0 to 0.2, performance dramatically increased commensurate with the baseline. 3) From 0.2 to 1.6, DAOD was stable with values at around $0.85, 0.7$ and 0.65 on three datasets. 4) When $\sigma > 1.6$, the performance dramatically dropped again to below the baseline. According to Fig. 5.3(d), we can choose $\sigma \in [0.2, 1.6]$.

Threshold t. Fig. 5.3(b) shows the classification accuracy with varying values of t. Theoretically, the threshold t is determined by the openness $\mathbb{O}$. When openness $\mathbb{O} \to 1$, $t \to 0$. When openness $\mathbb{O} \to 0$, $t \to 1$. However, according to Fig. 5.3(e), DAOD performed steadily when the threshold t varies from $[0, 0.9]$. This is because: 1) As the number of iterations T increases, the effect of t tapers off. 2) OSNN$^{\text{cv}}$-t is not sensitive to t.

Open-Set Parameter α. Fig. 5.3(f) plots the classification accuracy w.r.t. different values. Theoretically, when $\alpha \to 0$, the classifier can not recognize unknown samples, whereas, when $\alpha \to +\infty$, the classifier classifies all samples as unknown. These conjectures are verified by the results in Fig. 5.3(f), where the performance reaches its maximal point at around $\alpha = 0.3$ and then gradually drops as α increases. Performance was worst and lower than the baselines when $\alpha > 0.4$, because at this parameter setting many samples from known classes are classified as unknown. In general, we can choose α from $[0.2, 0.4]$.

Open-Set Parameter γ. The classification accuracy w.r.t. different values of γ is shown in Fig. 5.3(g). Theoretically, when $\gamma \to +\infty$, DAOD keeps the unknown target samples away from known source samples. As a result, few samples are classified as unknown classes. When $\gamma \to 0$, more samples are classified as unknown, and when $\gamma < 0.15$, its performance was worse than the baselines. Conversely, as γ increased, DAOD's performance dramatically increased, reaching its maximal value at around $\gamma = 0.3$ before gradually dropping again as γ continues to

increase. In general, we can choose $\gamma \in [0.15, 0.5]$.

Ablation Study. 1) α and γ are the two parameters that control the contribution of the open-set difference. As shown in Fig. 5.3(f), setting α closer to 0 reduces the contribution of the open-set difference and performance degrades compared to the optimal value of about $\alpha = 0.3$. Further, as shown in Fig. 5.3(g), setting γ closer to 0 also reduces the contribution of the open-set difference and again performance degrades compared to the optimal value of about $\gamma = 0.3$. Therefore, we can safely draw the conclusion that our proposed open-set difference is a necessary term for open-set domain adaption. 2) λ is the parameter that controls the contribution of the distribution discrepancy. As shown in Fig. 5.3(a), when λ is 0, performance is much worse than at other values, which shows that this term also makes a significant contribution to the final domain adaptation performance. 3) ρ controls the contribution of the manifold regularization. Fig. 5.3(b) shows there is no significant change in performance when ρ is set in the range 0 to 1. These results indicate that ρ makes no significant contributions to DAOD. 4) σ is used to avoid overfitting. As shown in Fig. 5.3(d), performance drops significantly when σ is set to 0. Thus, the term $\|\boldsymbol{h}\|_k^2$ is important to our algorithm.

Convergence Analysis. The results of the convergence analysis on the number of iterations T are provided in Fig. 5.3(h). As shown, DAOD reached a steady performance in only a few iterations ($T < 10$). This is a clear indication of the advantages of DAOD's ability to be trained in unsupervised open-set domain adaptation tasks.

5.5 Summary

To the best of our knowledge, this is the first work to present a theoretical analysis for open-set domain adaptation. In deriving a theoretical bound, we discovered a special term, open-set difference, which is crucial for recognizing unknown target samples. Using this open-set difference, we then constructed an unsupervised open-set domain adaptation al-

gorithm, called Distribution Alignment with Open Difference (DAOD). The algorithm enables the unknown target samples to be separated from samples by using open-set difference. We conduct 38 real-world UOSDA tasks (including 20 face recognition tasks and 18 object recognition tasks) for evaluating DAOD and existing UOSDA algorithms. Extensive experiments demonstrate that DAOD outperforms the pioneer UOSDA algorithms ATI and OSBP.

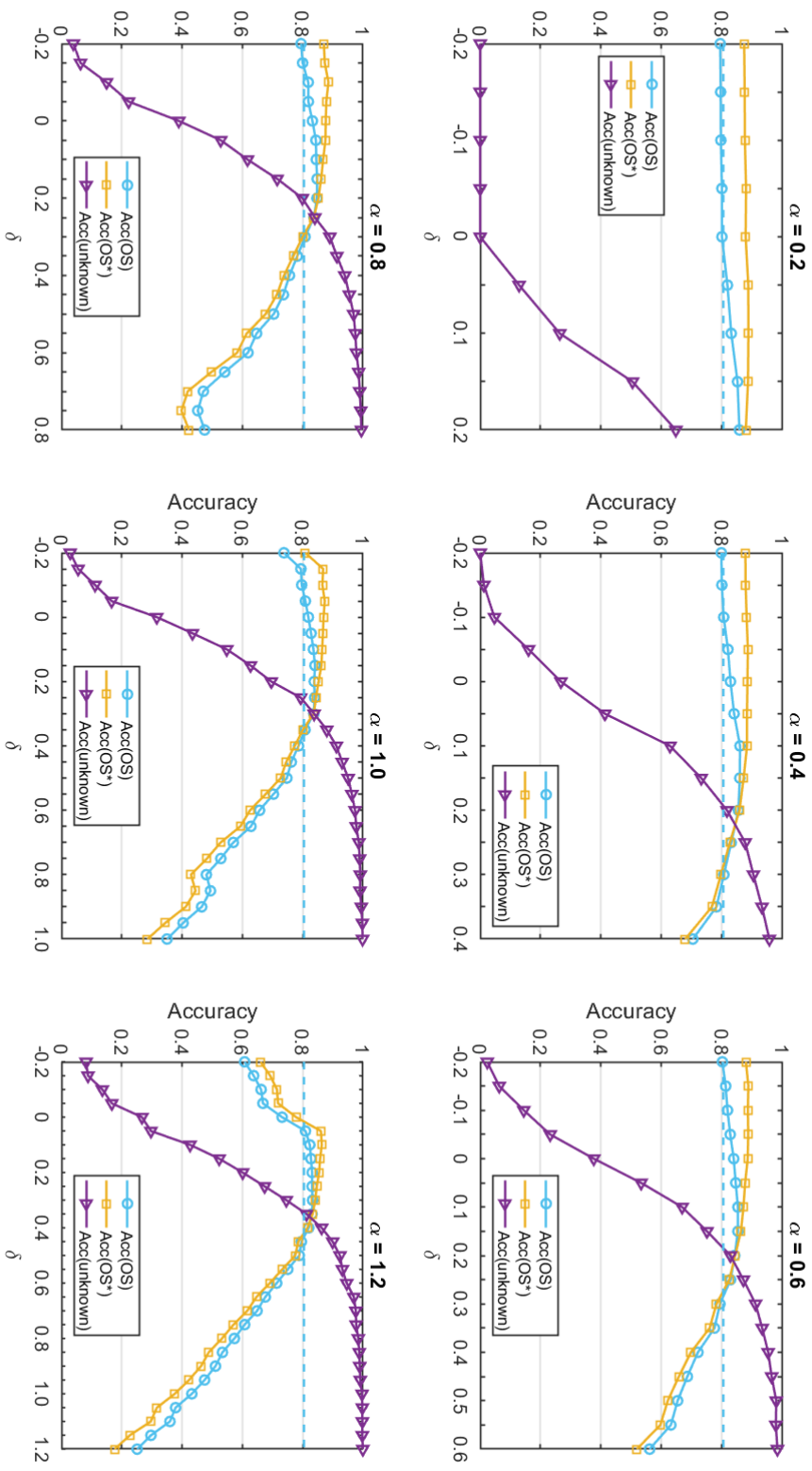


Figure 5.2: The horizontal axis is the difference in the open-set parameters $\delta = \alpha - \gamma$. In the figures, the difference δ is not larger than α, since the parameter γ is required to be larger than or equal to 0. If $\delta > 0$, α is larger than γ. If $\delta < 0$, γ is larger.

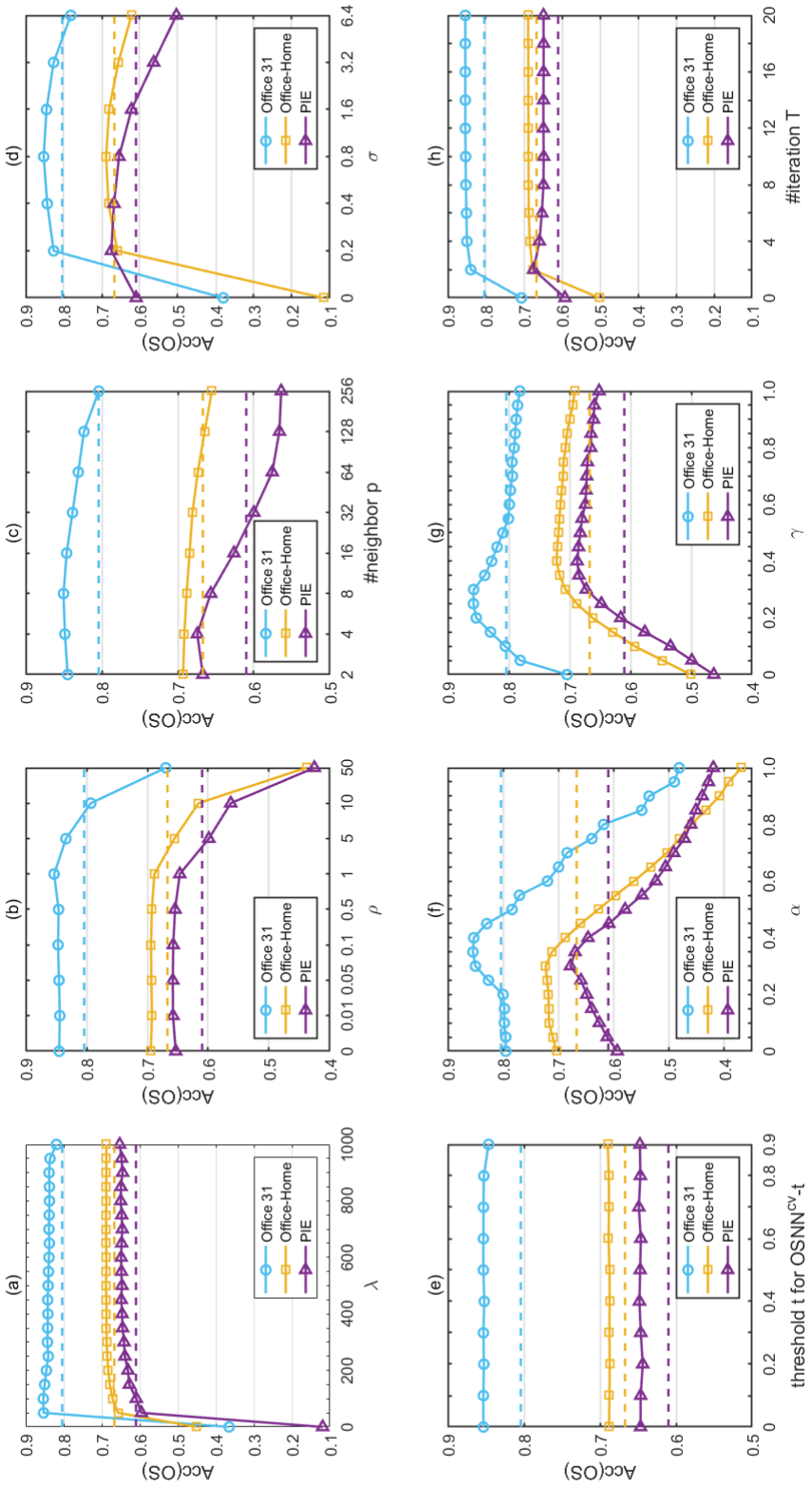


Figure 5.3: Parameter sensitivity study, ablation study and convergence analysis of the proposed DAOD algorithm.

Chapter 6

Learning in Open-World

Motivated by open-set domain adaption learning theory, open-set learning theory is introduced in this chapter.

The material of this chapter is based on the publication:
Zhen Fang and Jie Lu, Anjin Liu, Feng Liu, and Guangquan Zhang, Learning Bounds for Open-Set Learning,International Conference on Machine Learning (ICML), 2021.

6.1 Introduction

Supervised learning has achieved dramatic successes in many applications such as object detection [105], speech recognition [106] and natural language processing [107]. These successes are partly rooted in the closed-set assumption that training and test samples share a same label set. Under this assumption, the standard supervised learning is also regarded as closed-set learning (CSL) [108–110].

However, the closed-set assumption is not realistic during the testing phase (i.e., there are no labels in the samples) since it is not known whether the classes of test samples are from the label set of training samples. Test samples may come from some classes (unknown classes) that are not necessarily seen during training. These unknown classes can emerge unexpectedly and drastically weaken the performance of existing closed-set algorithms [111–113].

To solve supervised learning without closed-set assumption, Scheirer et al. [32] proposed a new problem setting, *open-set learning* (OSL), in which the test samples can come from any classes, even unknown classes.

An open-set classifier should classify samples from known classes into correct known classes while recognizing samples from unknown classes into unknown classes.

Remarkable advances have been achieved in open-set learning. The key challenge of OSL algorithms is to recognize the unknown classes accurately. To address this challenge, different strategies have been proposed such as open-space risk [32] and extreme value theory [116, 117]. Further, to adapt deep networks support to OSL, Bendale et al. [114], Ge et al. [115] proposed OpenMax and G-OpenMax, respectively.

While many OSL algorithms can be roughly interpreted as minimizing the open-space risk or using the extreme value theory, several disconnections still form non-negligible gaps between the theories and algorithms [108,166]. Very little theoretical groundwork has been undertaken to reveal the generalization ability of OSL from the perspective of statistical learning theory.

This work aims to bridge the gap between the theory and algorithm for OSL from the perspective of statistical learning theory. In particular, our theory answers an important question: under some assumptions, given training samples with size n, then *there exists an OSL algorithm such that the estimation error is close to* $O_p(1/\sqrt{n})$. This result reveals OSL problem can achieve an order of estimation error that is the same as CSL [57].

Since the test samples contain unknown classes, the distribution of test samples are intrinsically different from the distribution of training samples. Based on this fact, we aim to establish the OSL theory from transfer learning [167, 168], which learns knowledge for a given domain from a different, but relative domain. Using the transfer learning theory, we focus on constructing a suitable auxiliary domain, which contains the information of unknown classes. The construction of auxiliary domain depends on *covariate shift* [55, 169]. Transferring information from the auxiliary domain, we construct the generalization bound for OSL by using the transfer learning bound developed by [17], [52], [170].

Guided by our theory, we then devise an algorithm for OSL to bring the proposed OSL theory into reality. The novel algorithm auxiliary open-set risk (AOSR) is a neural network-based algorithm. AOSR mainly utilizes the instance-weighting strategy to align training samples and auxiliary samples generated by an auxiliary domain. Then, minimizing the auxiliary risk developed by our theory, AOSR can learn how to recognize unknown classes.

The contributions of this chapter are summarized as follows.

• We provide the theoretical analysis for open-set learning based on transfer learning and PAC theory. This is the *first* work to investigate the generalization error bound for open-set learning.

• Our theory answers an important question: under some assumptions, there exists an OSL algorithm such that the order of the estimation error is close to $O_p(1/\sqrt{n})$, if given training samples with size n.

• We conduct experiments on toy and benchmark datasets. Experiments support our theoretical results and show that our theoretical guided algorithm AOSR can achieve competitive performance compared with several popular baselines.

6.2 Open-Set Learning Theory

In this section, we introduce the basic notations used in this chapter and then provide theoretical analysis for open-set learning. All proofs can be found in Appendix C.

6.2.1 Problem Setting and Concepts

Here we recall the definition of open-set learning (OSL).

Assume that feature spaces $\mathcal{X}_s$, $\mathcal{X}_t$ and label sets $\mathcal{Y}_s$, $\mathcal{Y}_t$ satisfy $\mathcal{X}_s = \mathcal{X}_t$, $\mathcal{Y}_s \subset \mathcal{Y}_t$. Given sets of samples called the labeled source samples (training samples):

$$\mathcal{S} = \{(\mathbf{x}_s^i, \mathbf{y}_s^i)\}_{i=1}^{n_s} \sim P_{X_s Y_s} \ i.i.d.$$

Suppose that the joint distributions $P_{X_sY_s}$ and $P_{X_tY_t}$ satisfy that

$$P_{X_sY_s} = P_{X_tY_t|Y_t\in\mathcal{Y}_s}$$

The aim of ***open-set learning*** *(OSL) is to train a classifier g over $\mathcal{X}_t$ such that g classifies 1) target samples (test samples), whose labels are from $\mathcal{Y}_s$, into the correct classes; 2) target samples (test samples), whose labels are from $\mathcal{Y}_t - \mathcal{Y}_s$, as unknown.*

Note that the assumption $\mathcal{X}_s = \mathcal{X}_t$ and $P_{X_sY_s} = P_{X_tY_t|Y_t\in\mathcal{Y}_s}$. We introduce an equivalence, but simpler definition. Let $\mathcal{X}$ be the feature space ($\mathcal{X} = \mathcal{X}_s = \mathcal{X}_t$) and $\mathcal{Y}$ be the label set ($\mathcal{Y} = \mathcal{Y}_t$). *Known classes* are a subset of $\mathcal{Y}$. We define the label set of known classes as $\mathcal{Y}_k$ ($\mathcal{Y}_k = \mathcal{Y}_s$). Then, the *unknown classes* are from the space $\mathcal{Y}/\ \mathcal{Y}_k$. To be simple, we use one-hot vector $\mathbf{y}$ to present label y. The open-set learning problem can be redefined as follows:

Definition 18 (Open-Set Learning)**.** *Given independent and identically distributed (i.i.d.) samples $S = \{(\mathbf{x}^i, \mathbf{y}^i)\}_{i=1}^n$ drawn from $P_{XY|Y\in\mathcal{Y}_k}$. The aim of open-set learning is to train a classifier g using S such that g can classify 1) the samples from known classes into correct known classes; 2) the samples from unknown classes into unknown classes.*

Note that it is not necessary to classify unknown samples into correct unknown classes. For the sake of simplicity, we set all unknown samples are allocated to one big unknown class. Hence, without loss of generality, we assume that $\mathcal{Y}_k = \{\mathbf{y}_c\}_{c=1}^K$, $\mathcal{Y} = \{\mathbf{y}_c\}_{c=1}^{K+1}$, where the label $\mathbf{y}_c \in \mathbb{R}^{K+1}$ is a one-hot vector, whose c-th coordinate is 1 and other coordinates are 0. Label $\mathbf{y}_{K+1}$ represents unknown classes.

Given a loss function $\ell : \mathbb{R}^{K+1} \times \mathbb{R}^{K+1} \to \mathbb{R}_{\geq 0}$ and any scoring (hypothesis) function $\boldsymbol{h}$ from $\{\boldsymbol{h} : \mathcal{X} \to \mathbb{R}^{K+1}\}$, the *partial risks for known classes and unknown classes* are

$$\begin{aligned} R_P^k(\boldsymbol{h}) &= \int_{\mathcal{X}\times\mathcal{Y}_k} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}) \mathrm{d}P_{XY|Y\in\mathcal{Y}_k}(\mathbf{x}, \mathbf{y}), \\ R_P^u(\boldsymbol{h}) &= \int_{\mathcal{X}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}) \mathrm{d}P_{X|Y=\mathbf{y}_{K+1}}(\mathbf{x}). \end{aligned} \tag{6.2.1}$$

Then, the α-*risk* for P_{XY} is

$$R_P^{\alpha}(\boldsymbol{h}) = (1-\alpha)R_P^{k}(\boldsymbol{h}) + \alpha R_P^{u}(\boldsymbol{h}), \tag{6.2.2}$$

where α is the weight estimating the importance of unknown classes. When $\alpha = P(Y = \mathbf{y}_{K+1})$, it is easy to check that

$$R_P^{\alpha}(\boldsymbol{h}) = \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim P_{XY}}\ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}).$$

Similarly, given a different joint distribution Q_{XY}, we define $R_Q^{k}(\boldsymbol{h})$, $R_Q^{u}(\boldsymbol{h})$ and $R_Q^{\alpha}(\boldsymbol{h})$. Based on α-risk, we define almost agnostic probably approximate correct (PAC) for OSL.

Definition 19 (Almost Agnostic PAC Learnability)**.** *A hypothesis class $\mathcal{H} \subset \{\boldsymbol{h} : \mathcal{X} \to \mathbb{R}^{K+1}\}$ is almost agnostic PAC learnable for open-set learning, if given any $\epsilon_0 > 0$, there exists an OSL algorithm A_{ϵ_0} and $m_{\mathcal{H}} : (0,1)^2 \to \mathbb{N}$ with the following property: for any $0 < \epsilon, \delta < 1$, when running the algorithm A_{ϵ_0} on $n > m_{\mathcal{H}}(\epsilon, \delta)$ i.i.d. samples drawn from $P_{XY|Y\in\mathcal{Y}_k}$, the algorithm A_{ϵ_0} returns a hypothesis $\widehat{\boldsymbol{h}}$ such that, with probability of at least $1-\delta > 0$,*

$$R_P^{\alpha}(\widehat{\boldsymbol{h}}) \leq \min_{\boldsymbol{h}\in\mathcal{H}} R_P^{\alpha}(\boldsymbol{h}) + \epsilon + \epsilon_0.$$

Theorems 11 and 12 imply there exists almost agnostic PAC learnable $\mathcal{H}$ for open-set learning under mild assumptions.

6.2.2 Transfer Between Domains

Since there are no samples regarding the unknown classes, we cannot directly analyze the partial risk for unknown classes only using samples S from known classes. To analyze the partial risk for unknown classes, we introduce an auxiliary domain Q_{XY}, which is used to transfer the information from unknown classes.

Definition 20 (Auxiliary Domain)**.** *A domain Q_{XY} defined over $\mathcal{X}\times\mathcal{Y}$ is called the auxiliary domain for P_{XY}, if $Q_{X|Y\in\mathcal{Y}_k} = P_{X|Y\in\mathcal{Y}_k}, Q_{Y|X} = P_{Y|X}$*

and $P_X \ll Q_X$.

It is clear that P_{XY} and Q_{XY} are same if we restrict both of them in the support set of known classes.

Remark 5. *Since we do not have any information about samples from unknown classes in the training set, it is unknown whether* $Q_{X|Y=\mathbf{y}_{K+1}} = P_{X|Y=\mathbf{y}_{K+1}}$. *In Section 6.2.3, we will introduce how to construct* Q_{XY} *such that* $Q_{X|Y=\mathbf{y}_{K+1}}$ *is a uniform distribution. Namely, any sample drawn from* $Q_{X|Y=\mathbf{y}_{K+1}}$ *has the same probability.*

Then, it is interesting to know the discrepancy between $R_P^\alpha(\boldsymbol{h})$ and $R_Q^\alpha(\boldsymbol{h})$ given the same hypothesis $\boldsymbol{h}$. Before doing this, the disparity discrepancy between distributions need to be introduced. Using $\mathcal{H}\Delta\mathcal{H}$-divergence defined in Definition 8, we show that

Theorem 7. *Given a loss* ℓ *satisfying the triangle inequality, and a hypothesis space* $\mathcal{H} \subset \{\boldsymbol{h} : \mathcal{X} \to \mathbb{R}^{K+1}\}$, *if* Q_{XY} *is the auxiliary domain for* P_{XY}, *then for any* $\boldsymbol{h} \in \mathcal{H}$, *the difference* $|R_P^\alpha(\boldsymbol{h}) - R_Q^\alpha(\boldsymbol{h})|$ *is bounded by*

$$\alpha d_{\mathcal{H}}^{\ell}(P_{X|Y=\mathbf{y}_{K+1}}, Q_{X|Y=\mathbf{y}_{K+1}}) + \alpha\Lambda,$$

where $\alpha = Q(Y = \mathbf{y}_{K+1})$, $d_{\mathcal{H}}^{\ell}$ *is the* $\mathcal{H}\Delta\mathcal{H}$*-divergence defined in definition 8,*

$$\Lambda = \min_{\boldsymbol{h}' \in \mathcal{H}} \left(R_P^u(\boldsymbol{h}') + R_{Q,u}(\boldsymbol{h}') \right) \tag{6.2.3}$$

is the combined risk for the unknown classes, $R_P^\alpha(\boldsymbol{h})$ *is the* α*-risk for* P_{XY} *and* $R_Q^\alpha(\boldsymbol{h})$ *is the* α*-risk for* Q_{XY}.

Proof of Theorem 7.

$$\begin{aligned}
&|R_P^\alpha(\boldsymbol{h}) - R_Q^\alpha(\boldsymbol{h})| \\
=&|(1-\alpha)R_{P,k}(\boldsymbol{h}) + \alpha R_{P,u}(\boldsymbol{h}) - (1-\alpha)R_{Q,k}(\boldsymbol{h}) - \alpha R_{Q,u}(\boldsymbol{h})| \\
=&\Big|(1-\alpha)\int_{\mathcal{X}\times\mathcal{Y}_k} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}) \mathrm{d}P_{XY|Y\in\mathcal{Y}_k}(\mathbf{x}, \mathbf{y}) + \alpha R_{P,u}(\boldsymbol{h}) \\
&-(1-\alpha)\int_{\mathcal{X}\times\mathcal{Y}_k} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}) \mathrm{d}Q_{XY|Y\in\mathcal{Y}_k}(\mathbf{x}, \mathbf{y}) - \alpha R_{Q,u}(\boldsymbol{h})\Big| \\
=&\alpha\Big|R_{P,u}(\boldsymbol{h}) - R_{Q,u}(\boldsymbol{h})\Big| \quad \text{we have used } Q_{XY|Y\in\mathcal{Y}_k} = P_{XY|Y\in\mathcal{Y}_k}
\end{aligned}$$

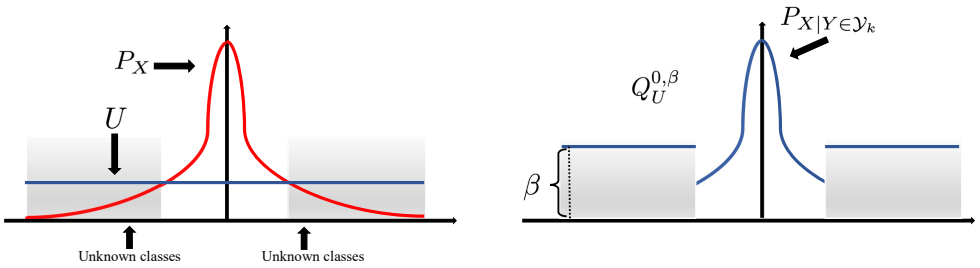


Figure 6.1: The left figure shows the auxiliary distribution U and marginal distribution P_X (red curve: distribution P_X; blue lines: distribution U introduced in Definition 21; grey regions: the regions for unknown classes). The right figure shows the marginal distribution $Q_U^{0,\beta}$ of ideal auxiliary domain generated by U, $P_{X|Y\in\mathcal{Y}_k}$ and $L_{0,\beta}$ (blue lines and curve: $Q_U^{0,\beta}$ defined in Definition 21).

$$\leq\alpha\Big|\int_{\mathcal{X}}\ell(\boldsymbol{h}(\mathbf{x}),\boldsymbol{h}'(\mathbf{x}))\mathrm{d}P_{X|Y=\mathbf{y}_{K+1}}(\mathbf{x})-\int_{\mathcal{X}}\ell(\boldsymbol{h}(\mathbf{x}),\boldsymbol{h}'(\mathbf{x}))\mathrm{d}Q_{X|Y=\mathbf{y}_{K+1}}(\mathbf{x})\Big|$$
$$+\alpha\int_{\mathcal{X}}\ell(\boldsymbol{h}'(\mathbf{x}),\mathbf{y}_{K+1})\mathrm{d}P_{X|Y=\mathbf{y}_{K+1}}(\mathbf{x})+\alpha\int_{\mathcal{X}}\ell(\boldsymbol{h}'(\mathbf{x}),\mathbf{y}_{K+1})\mathrm{d}Q_{X|Y=\mathbf{y}_{K+1}}(\mathbf{x})$$

the triangle inequality is used

$$\leq\alpha d_{\mathcal{H}}^{\ell}(P_{X|Y=\mathbf{y}_{K+1}},Q_{X|Y=\mathbf{y}_{K+1}})+\alpha\int_{\mathcal{X}}\ell(\boldsymbol{h}'(\mathbf{x}),\mathbf{y}_{K+1})\mathrm{d}P_{X|Y=\mathbf{y}_{K+1}}(\mathbf{x})$$
$$+\alpha\int_{\mathcal{X}}\ell(\boldsymbol{h}'(\mathbf{x}),\mathbf{y}_{K+1})\mathrm{d}Q_{X|Y=\mathbf{y}_{K+1}}(\mathbf{x}).$$

Hence,

$$|R_P^{\alpha}(\boldsymbol{h})-R_Q^{\alpha}(\boldsymbol{h})|=\min_{\boldsymbol{h}'\in\mathcal{H}}|R_P^{\alpha}(\boldsymbol{h})-R_Q^{\alpha}(\boldsymbol{h})|$$
$$\leq\alpha d_{\mathcal{H}}^{\ell}(P_{X|Y=\mathbf{y}_{K+1}},Q_{X|Y=\mathbf{y}_{K+1}})+\alpha\Lambda.$$

□

Theorem 7 implies that there exists a gap between $R_P^{\alpha}(\boldsymbol{h})$ and $R_Q^{\alpha}(\boldsymbol{h})$. The gap is related to domain discrepancy for unknown classes between P_{XY} and Q_{XY}. To further eliminate the gap between $R_P^{\alpha}(\boldsymbol{h})$ and $R_Q^{\alpha}(\boldsymbol{h})$, additional conditions about the hypothesis space $\mathcal{H}$ are indispensable.

Assumption 3 (Realization for Unknown Classes)**.** *A hypothesis $\mathcal{H}\subset\{\boldsymbol{h}:\mathcal{X}\to\mathcal{Y}\}$ is realization for unknown classes, if there exist a hypothesis function $\tilde{\boldsymbol{h}}\in\mathcal{H}$ and a distribution $\tilde{P}$ defined over $\mathcal{X}$ with* supp $\tilde{P}=\mathcal{X}$ *satisfying for any $\boldsymbol{h}\in\mathcal{H}$, there exists $\boldsymbol{h}'\in\mathcal{H}$ such that $\boldsymbol{h}'(\mathbf{x})=\mathbf{y}_{K+1}$, if*

$\tilde{\boldsymbol{h}}(\mathbf{x}) = \mathbf{y}_{K+1}$, *otherwise,* $\boldsymbol{h}'(\mathbf{x}) = \boldsymbol{h}(\mathbf{x})$*; and*

$$\int_{\mathcal{X}\times\mathcal{Y}} \ell(\phi \circ \tilde{\boldsymbol{h}}(\mathbf{x}), \phi(\mathbf{y}))\mathrm{d}P_{Y|X}(\mathbf{y}|\mathbf{x})\mathrm{d}\tilde{P}(\mathbf{x}) = 0,$$

where ϕ *is a function defined over* $\mathcal{Y}$ *and defined as follows* $\phi(\mathbf{y}) = \mathbf{y}_{K+1}$*, if* $\mathbf{y} = \mathbf{y}_{K+1}$*; otherwise,* $\phi(\mathbf{y}) = \mathbf{y}_1$*.*

Remark 6. *Assumption 3 implies that the hypothesis space* $\mathcal{H}$ *is complexity enough so that the unknown classes can be classified perfectly by many hypothesis functions. The assumption can be regarded as the open-set version of realization assumption [57, 171], which is a basic concept in statistical learning theory.*

Theorem 8. *Given a loss* ℓ *satisfying* $\ell(\mathbf{y}, \mathbf{y}') = 0$ *iff* $\mathbf{y} = \mathbf{y}'$*, and a hypothesis space* $\mathcal{H} \subset \{\boldsymbol{h} : \mathcal{X} \to \mathbb{R}^{K+1}\}$ *satisfying Assumption 3, if* Q_{XY} *is the auxiliary domain for* P_{XY} *and assume* $P_X \ll Q_X \ll \tilde{P}$*, where* $\tilde{P}$ *is the distribution introduced in Assumption 3, then for any* $0 < \alpha < 1$*,*

$$\min_{\boldsymbol{h}\in\mathcal{H}} R_Q^\alpha(\boldsymbol{h}) = \min_{\boldsymbol{h}\in\mathcal{H}} R_P^\alpha(\boldsymbol{h}),$$
$$\arg\min_{\boldsymbol{h}\in\mathcal{H}} R_Q^\alpha(\boldsymbol{h}) \subset \arg\min_{\boldsymbol{h}\in\mathcal{H}} R_P^\alpha(\boldsymbol{h}).$$

Proof of Theorem 8. **Step 1.** Note that

$$\int_{\mathcal{X}\times\mathcal{Y}} \ell(\phi \circ \tilde{\boldsymbol{h}}(\mathbf{x}), \phi(\mathbf{y}))\mathrm{d}P_{Y|X}(\mathbf{x})\mathrm{d}\tilde{P}(\mathbf{x}) = 0,$$

hence, if we set $\tilde{P}_{X,Y} = \tilde{P}P_{Y|X}$, then

$$\int_{\mathcal{X}\times\mathcal{Y}} \ell(\phi \circ \tilde{\boldsymbol{h}}(\mathbf{x}), \phi(\mathbf{y}))\mathrm{d}\tilde{P}_{X,Y|Y\in\mathcal{Y}_k}(\mathbf{x}, \mathbf{y}) = 0,$$
$$\int_{\mathcal{X}} \ell(\phi \circ \tilde{\boldsymbol{h}}(\mathbf{x}), \phi(\mathbf{y}_{K+1}))\mathrm{d}\tilde{P}_{X|Y=\mathbf{y}_{K+1}}(\mathbf{x}) = 0.$$

Note that $\ell(\mathbf{y}, \mathbf{y}') = 0$ iff $\mathbf{y} = \mathbf{y}'$, hence, $\tilde{\boldsymbol{h}}(\mathbf{x}) = \mathbf{y}_{K+1}$, for any $\mathbf{x}$ from supp $\tilde{P}_{X|Y=\mathbf{y}_{K+1}}$ a.e. $\tilde{P}$ and $\tilde{\boldsymbol{h}}(\mathbf{x}) \neq \mathbf{y}_{K+1}$, for any $\mathbf{x}$ from supp $\tilde{P}_{X|Y\in\mathcal{Y}_k}$ a.e. $\tilde{P}$.

Step 2. Because $P_X \ll Q_X \ll \tilde{P}$, then supp $\tilde{P}_{X|Y\in\mathcal{Y}_k} \supset$ supp $Q_{X|Y\in\mathcal{Y}_k} \supset$ supp $P_{X|Y\in\mathcal{Y}_k}$ and supp $\tilde{P}_{X|Y=\mathbf{y}_{K+1}} \supset$ supp $Q_{X|Y=\mathbf{y}_{K+1}} \supset$ supp $P_{X|Y=\mathbf{y}_{K+1}}$.

Step 3. We need to check that $\min_{\boldsymbol{h}\in\mathcal{H}} R_P^{\alpha}(\boldsymbol{h}) = (1-\alpha)\min_{\boldsymbol{h}\in\mathcal{H}} R_{P,k}(\boldsymbol{h})$. First, it is clear that $\min_{\boldsymbol{h}\in\mathcal{H}} R_P^{\alpha}(\boldsymbol{h}) \geq (1-\alpha)\min_{\boldsymbol{h}\in\mathcal{H}} R_{P,k}(\boldsymbol{h})$. If there exists $\boldsymbol{h}_P \in \mathcal{H}$ such that $\min_{\boldsymbol{h}\in\mathcal{H}} R_P^{\alpha}(\boldsymbol{h}) > (1-\alpha)\min_{\boldsymbol{h}\in\mathcal{H}} R_{P,k}(\boldsymbol{h}_P)$.

Set $\tilde{\boldsymbol{h}}_P(\mathbf{x}) = \mathbf{y}_{K+1}$, if $\tilde{\boldsymbol{h}}(\mathbf{x}) = \mathbf{y}_{K+1}$; otherwise, $\tilde{\boldsymbol{h}}_P(\mathbf{x}) = \boldsymbol{h}_P(\mathbf{x})$, hence, using the results of Step 1 and Step 2, we know $\{\mathbf{x} : \tilde{\boldsymbol{h}}(\mathbf{x}) = \mathbf{y}_{K+1}\} \supset \text{supp } P_{X|Y=\mathbf{y}_{K+1}}$. Then,

$$
\begin{aligned}
&(1-\alpha)\int_{\mathcal{X}\times\mathcal{Y}_k} \ell(\boldsymbol{h}_P(\mathbf{x}),\mathbf{y})\mathrm{d}P_{X,Y|Y\in\mathcal{Y}_k}(\mathbf{x},\mathbf{y}) \\
=&(1-\alpha)\int_{\{\text{supp } P_{X|Y\in\mathcal{Y}_k}\}\times\mathcal{Y}_k} \ell(\boldsymbol{h}_P(\mathbf{x}),\mathbf{y})\mathrm{d}P_{X,Y|Y\in\mathcal{Y}_k}(\mathbf{x},\mathbf{y}) \\
=&(1-\alpha)\int_{\{\text{supp } P_{X|Y\in\mathcal{Y}_k}\}\times\mathcal{Y}_k} \ell(\tilde{\boldsymbol{h}}_P(\mathbf{x}),\mathbf{y})\mathrm{d}P_{X,Y|Y\in\mathcal{Y}_k}(\mathbf{x},\mathbf{y}) \\
&\text{have used } \tilde{\boldsymbol{h}}(\mathbf{x}) \neq \mathbf{y}_{K+1}, \text{ for } \mathbf{x}\in\text{supp } \tilde{P}_{X|Y\in\mathcal{Y}_k} \text{ a.e. } \tilde{P} \\
=&(1-\alpha)\int_{\{\text{supp } P_{X|Y\in\mathcal{Y}_k}\}\times\mathcal{Y}_k} \ell(\tilde{\boldsymbol{h}}_P(\mathbf{x}),\mathbf{y})\mathrm{d}P_{X,Y|Y\in\mathcal{Y}_k}(\mathbf{x},\mathbf{y}) + 0 \\
=&(1-\alpha)\int_{\{\text{supp } P_{X|Y\in\mathcal{Y}_k}\}\times\mathcal{Y}_k} \ell(\tilde{\boldsymbol{h}}_P(\mathbf{x}),\mathbf{y})\mathrm{d}P_{X,Y|Y\in\mathcal{Y}_k}(\mathbf{x},\mathbf{y}) \\
&+\alpha\int_{\text{supp } P_{X|Y=\mathbf{y}_{K+1}}} \ell(\tilde{\boldsymbol{h}}(\mathbf{x}),\mathbf{y}_{K+1})\mathrm{d}P_{X|Y=\mathbf{y}_{K+1}}(\mathbf{x}) \\
=&(1-\alpha)\int_{\{\text{supp } P_{X|Y\in\mathcal{Y}_k}\}\times\mathcal{Y}_k} \ell(\tilde{\boldsymbol{h}}_P(\mathbf{x}),\mathbf{y})\mathrm{d}P_{X,Y|Y\in\mathcal{Y}_k}(\mathbf{x},\mathbf{y}) \\
&+\alpha\int_{\text{supp } P_{X|Y=\mathbf{y}_{K+1}}} \ell(\tilde{\boldsymbol{h}}_P(\mathbf{x}),\mathbf{y}_{K+1})\mathrm{d}P_{X|Y=\mathbf{y}_{K+1}}(\mathbf{x}) \\
=&R_P^{\alpha}(\tilde{\boldsymbol{h}}_P) \geq \min_{\boldsymbol{h}\in\mathcal{H}} R_P^{\alpha}(\boldsymbol{h}),
\end{aligned}
$$

hence, $\min_{\boldsymbol{h}\in\mathcal{H}} R_P^{\alpha}(\boldsymbol{h}) = (1-\alpha)\min_{\boldsymbol{h}\in\mathcal{H}} R_{P,k}(\boldsymbol{h})$. Similarly, we can prove that $\min_{\boldsymbol{h}\in\mathcal{H}} R_Q^{\alpha}(\boldsymbol{h}) = (1-\alpha)\min_{\boldsymbol{h}\in\mathcal{H}} R_{Q,k}(\boldsymbol{h})$. Because $Q_{X|Y\in\mathcal{Y}_k} = P_{X|Y\in\mathcal{Y}_k}$, hence, $\min_{\boldsymbol{h}\in\mathcal{H}} R_{Q,k}(\boldsymbol{h}) = \min_{\boldsymbol{h}\in\mathcal{H}} R_{P,k}(\boldsymbol{h})$. Using the results of Step 3, we obtain that

$$
\min_{\boldsymbol{h}\in\mathcal{H}} R_Q(\boldsymbol{h}) = \min_{\boldsymbol{h}\in\mathcal{H}} R_P(\boldsymbol{h}). \tag{6.2.4}
$$

Step 4. Given any $\boldsymbol{h}^* \in \arg\min_{\boldsymbol{h}\in\mathcal{H}} R_P^{\alpha}(\boldsymbol{h})$, then we construct $\tilde{\boldsymbol{h}}^*$ such

that

$$\tilde{\boldsymbol{h}}^*(\mathbf{x}) = \mathbf{y}_{K+1}, \quad \text{if} \quad \tilde{\boldsymbol{h}}(\mathbf{x}) = \mathbf{y}_{K+1}; \quad \text{otherwise}, \quad \tilde{\boldsymbol{h}}^*(\mathbf{x}) = \boldsymbol{h}^*(\mathbf{x}).$$

It is clear that $\tilde{\boldsymbol{h}}^* \in \mathcal{H}$ according to assumption 3.

Then,

$$\begin{aligned}
&R_P^\alpha(\boldsymbol{h}^*) \\
\geq&(1-\alpha)\int_{\mathcal{X}\times\mathcal{Y}_k} \ell(\boldsymbol{h}^*(\mathbf{x}),\mathbf{y})\mathrm{d}P_{X,Y|Y\in\mathcal{Y}_k}(\mathbf{x},\mathbf{y}) \\
=&(1-\alpha)\int_{\{\text{supp } P_{X|Y\in\mathcal{Y}_k}\}\times\mathcal{Y}_k} \ell(\boldsymbol{h}^*(\mathbf{x}),\mathbf{y})\mathrm{d}P_{X,Y|Y\in\mathcal{Y}_k}(\mathbf{x},\mathbf{y}) \\
=&(1-\alpha)\int_{\{\text{supp } P_{X|Y\in\mathcal{Y}_k}\}\times\mathcal{Y}_k} \ell(\tilde{\boldsymbol{h}}^*(\mathbf{x}),\mathbf{y})\mathrm{d}P_{X,Y|Y\in\mathcal{Y}_k}(\mathbf{x},\mathbf{y}) \\
&\text{have used } \tilde{\boldsymbol{h}}(\mathbf{x}) \neq \mathbf{y}_{K+1}, \text{ for } \mathbf{x} \in \text{supp } \tilde{P}_{X|Y\in\mathcal{Y}_k} \text{ a.e. } \tilde{P} \\
=&(1-\alpha)\int_{\{\text{supp } P_{X|Y\in\mathcal{Y}_k}\}\times\mathcal{Y}_k} \ell(\tilde{\boldsymbol{h}}^*(\mathbf{x}),\mathbf{y})\mathrm{d}P_{X,Y|Y\in\mathcal{Y}_k}(\mathbf{x},\mathbf{y}) + 0 \\
=&(1-\alpha)\int_{\{\text{supp } P_{X|Y\in\mathcal{Y}_k}\}\times\mathcal{Y}_k} \ell(\tilde{\boldsymbol{h}}^*(\mathbf{x}),\mathbf{y})\mathrm{d}P_{X,Y|Y\in\mathcal{Y}_k}(\mathbf{x},\mathbf{y}) \\
&+\alpha\int_{\text{supp } P_{X|Y=\mathbf{y}_{K+1}}} \ell(\tilde{\boldsymbol{h}}(\mathbf{x}),\mathbf{y}_{K+1})\mathrm{d}P_{X|Y=\mathbf{y}_{K+1}}(\mathbf{x}) \\
=&(1-\alpha)\int_{\{\text{supp } P_{X|Y\in\mathcal{Y}_k}\}\times\mathcal{Y}_k} \ell(\tilde{\boldsymbol{h}}^*(\mathbf{x}),\mathbf{y})\mathrm{d}P_{X,Y|Y\in\mathcal{Y}_k}(\mathbf{x},\mathbf{y}) \\
&+\alpha\int_{\text{supp } P_{X|Y=\mathbf{y}_{K+1}}} \ell(\tilde{\boldsymbol{h}}^*(\mathbf{x}),\mathbf{y}_{K+1})\mathrm{d}P_{X|Y=\mathbf{y}_{K+1}}(\mathbf{x}) \\
=&R_P^\alpha(\tilde{\boldsymbol{h}}^*).
\end{aligned}$$

Hence, for any $\boldsymbol{h}^* \in \arg\min_{\boldsymbol{h}\in\mathcal{H}} R_P^\alpha(\boldsymbol{h})$,

$$\begin{aligned}
&\int_{\mathcal{X}} \ell(\boldsymbol{h}^*(\mathbf{x}),\mathbf{y}_{K+1})\mathrm{d}P_{X|Y=\mathbf{y}_{K+1}}(\mathbf{x}) \\
=&\int_{\text{supp}P_{X|Y=\mathbf{y}_{K+1}}} \ell(\boldsymbol{h}^*(\mathbf{x}),\mathbf{y}_{K+1})\mathrm{d}P_{X|Y=\mathbf{y}_{K+1}}(\mathbf{x}) = 0.
\end{aligned}$$

Similarly, we can prove that for any $\boldsymbol{h}^* \in \arg\min_{\boldsymbol{h}\in\mathcal{H}} R_Q^\alpha(\boldsymbol{h})$,

$$\int_{\mathcal{X}} \ell(\boldsymbol{h}^*(\mathbf{x}),\mathbf{y}_{K+1})\mathrm{d}Q_{X|Y=\mathbf{y}_{K+1}}(\mathbf{x}) = 0.$$

Step 5. Given any $h_Q \in \arg\min_{\boldsymbol{h}\in\mathcal{H}} R_Q^\alpha(\boldsymbol{h})$, we can find that (using result of Step 3)

$$R_Q^\alpha(h_Q) = (1-\alpha)R_{Q,k}(h_Q) = (1-\alpha)R_{P,k}(h_Q),$$

and

$$\int_{\mathcal{X}} \ell(\boldsymbol{h}_Q(\mathbf{x}), \mathbf{y}_{K+1}) \mathrm{d}Q_{X|Y=\mathbf{y}_{K+1}}(\mathbf{x}) = 0.$$

Because $P_X \ll Q_X$, we know

$$P_{X|Y=\mathbf{y}_{K+1}} \ll Q_{X|Y=\mathbf{y}_{K+1}},$$

which implies that

$$\int_{\mathcal{X}} \ell(\boldsymbol{h}_Q(\mathbf{x}), \mathbf{y}_{K+1}) \mathrm{d}P_{X|Y=\mathbf{y}_{K+1}}(\mathbf{x}) = 0.$$

Hence,

$$R_Q^\alpha(\boldsymbol{h}_Q) = (1-\alpha)R_{Q,k}(\boldsymbol{h}_Q) = (1-\alpha)R_{P,k}(\boldsymbol{h}_Q) + \alpha * 0 = R_P^\alpha(\boldsymbol{h}_Q).$$

Using the result (see Eq. (6.2.4)) of Step 3,

$$\min_{\boldsymbol{h}\in\mathcal{H}} R_Q^\alpha(\boldsymbol{h}) = \min_{\boldsymbol{h}\in\mathcal{H}} R_P^\alpha(\boldsymbol{h}).$$

We obtain that

$$\boldsymbol{h}_Q \in \arg\min_{\boldsymbol{h}\in\mathcal{H}} R_P^\alpha(\boldsymbol{h}),$$

this implies

$$\arg\min_{\boldsymbol{h}\in\mathcal{H}} R_Q^\alpha(\boldsymbol{h}) \subset \arg\min_{\boldsymbol{h}\in\mathcal{H}} R_P^\alpha(\boldsymbol{h}).$$

□

6.2.3 Construction of Ideal Auxiliary Domain

As mentioned above, the auxiliary domain plays an important role to address the open-set learning problem from a transfer learning perspective.

Thus, in this subsection, we first show how to construct an ideal auxiliary domain and then demonstrate how to estimate the ideal auxiliary domain via finite samples.

Definition 21 (Ideal Auxiliary Domain (IAD))**.** *Given the distribution P_{XY} defined in Problem 18 and an auxiliary distribution U defined over $\mathcal{X}$ such that $P_{X|Y\in\mathcal{Y}_k} \ll U$, then the ideal auxiliary domain regarding to P_{XY} is*

$$Q_U^{0,\beta} P_{Y|X},$$

where $Q_U^{0,\beta}$ is the marginal distribution defined over $\mathcal{X}$:

$$Q_U^{0,\beta}(A) = \gamma \int_A L_{0,\beta}(r(\mathbf{x}))\mathrm{d}U(\mathbf{x}), \quad here \tag{6.2.5}$$

$$\gamma = \frac{1}{1+\beta U(r=0)}, \tag{6.2.6}$$

$$L_{0,\beta}(x) = \begin{cases} x+\beta, & x \le 0, \\ x, & x > 0, \end{cases}$$

A is any U-measurable set, $r(\mathbf{x})$ is the density ratio between $P_{X|Y\in\mathcal{Y}_k}$ and Usuch that

$$P_{X|Y\in\mathcal{Y}_k}(A) = \int_A r(\mathbf{x})\mathrm{d}U(\mathbf{x}),$$

and $\beta > 0$ is a parameter to tune the density of $Q_U^{0,\beta}$ for unknown classes.

In Definition 21, the probability value of distribution $Q_U^{0,\beta}$ in space $\mathcal{X}/$ supp r is a constant β. In detail, if P_{XY} has no overlap between known and unknown classes, any sample from $Q_{X|Y=\mathbf{y}_{K+1}}$ shares same probability (see Fig. 6.1). In addition, an auxiliary distribution U satisfying $P_{X|Y\in\mathcal{Y}_k} \ll U$ is needed. The samples drawn from U can be generated by a gaussian distribution or uniform distribution with suitable support set. Given finite samples $T = \{\tilde{\mathbf{x}}^j\}_{j=1}^m$ drawn (i.i.d.) from a given distribution U as introduced in Definition 21. We introduce how to use T and S to construct an approximate form of $Q_U^{0,\beta}$ introduced in Definition 21. To simple, we provide a mild assumption as follows:

Definition 22. *Distributions $P_{X|Y\in\mathcal{Y}_k}$ and U introduced in Definition 21 are continuous distributions with density functions $p(\mathbf{x})$ and $q(\mathbf{x})$, respectively.*

Remark 7. *The assumption that $P_{X|Y\in\mathcal{Y}_k}$ and U are continuous can be replaced by a weaker assumption: $P_{X|Y\in\mathcal{Y}_k}, U \ll \mu$, where μ is a measure defined over $\mathcal{X}$. With the weaker assumption, all theorems still hold.*

Note that the density ratio $r = p/q$ required in $Q_U^{0,\beta}$ is unknown. To compute the density ratio r using S and T, the density ratio estimation algorithms are indispensable. Considering the property of statistical convergence, we use *kernelized variant of unconstrained least-squares importance fitting* (KuLSIF) [172] to estimate the density ratio in the theoretical part: given RKHS space $\mathcal{H}_k$,

$$\min_{w\in\mathcal{H}_k} \frac{\sum_{\mathbf{x}\in T} w^2(\mathbf{x})}{m} - 2\frac{\sum_{(\mathbf{x},\mathbf{y})\in S} w(\mathbf{x})}{n} + \lambda\|w\|_k^2, \tag{6.2.7}$$

where λ is the regularization parameter. Then, we assume $\widehat{w}$ is the solution of Eq. (6.2.7). After instance re-weighting, we regard the following measure

$$\widehat{Q}_T^{\tau,\beta} = \frac{\gamma}{m}\sum_{\mathbf{x}\in T} L_{\tau,\beta}(\widehat{w}(\mathbf{x}))\delta_{\mathbf{x}}, \tag{6.2.8}$$

as the approximation of $Q_U^{0,\beta}$, where γ is defined in Eq. (6.2.6),

$$L_{\tau,\beta}(x) = \begin{cases} x+\beta, & x \le \tau; \\ x, & 2\tau \le x; \\ \big(1-\frac{\beta}{\tau}\big)x + 2\beta, & \tau < x < 2\tau, \end{cases}$$

and $\tau > 0$ is the threshold to select whether a sample $\mathbf{x} \in T$ is from unknown classes or known classes.

6.2.4 Empirical Estimation for IAD Risk

In this subsection, we first set the ideal auxiliary domain $Q_U^{0,\beta}P_{Y|X}$ as Q_{XY}, then we analyze the IAD risk $R_Q^\alpha(\boldsymbol{h})$ from an approximate view,

where $\alpha = 1 - 1/\big(1 + \beta U(r = 0)\big)$. In detail, the IAD risk $R_Q^\alpha(\boldsymbol{h})$ can be written as follows

$$R_Q^\alpha(\boldsymbol{h}) = \int_{\mathcal{X}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}) \mathrm{d}P_{Y|X}(\mathbf{y}|\mathbf{x}) \mathrm{d}Q_U^{0,\beta}(\mathbf{x}). \tag{6.2.9}$$

Then, we use $\widehat{Q}_T^{\tau,\beta}$ (Eq. (6.2.8)) to construct auxiliary risk to approximate IAD risk.

Definition 23 (Auxiliary Risk)**.** *Given samples S with size n drawn from $P_{XY|Y\in\mathcal{Y}_k}$ and T with size m drawn from U, i.i.d., then the auxiliary risk for a hypothesis function $\boldsymbol{h}$ is*

$$\widehat{R}_{S,T}^{\tau,\beta}(\boldsymbol{h}) = \widehat{R}_S(\boldsymbol{h}) + \Delta_{S,T}^{\tau,\beta}(\boldsymbol{h}), \tag{6.2.10}$$

where

$$\begin{aligned}
&\widehat{R}_S(\boldsymbol{h}) = \frac{1}{n} \sum_{(\mathbf{x},\mathbf{y})\in S} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}),\\
&\Delta_{S,T}^{\tau,\beta}(\boldsymbol{h}) = \max\{\widehat{R}_T^{\tau,\beta}(\boldsymbol{h}, \mathbf{y}_{K+1}) - \widehat{R}_S(\boldsymbol{h}, \mathbf{y}_{K+1}), 0\},\\
&\widehat{R}_T^{\tau,\beta}(\boldsymbol{h}, \mathbf{y}_{K+1}) = \frac{1}{\gamma} \int_{\mathcal{X}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}) \mathrm{d}\widehat{Q}_T^{\tau,\beta}(\mathbf{x})\\
&\qquad\qquad = \frac{1}{m} \sum_{\mathbf{x}\in T} L_{\tau,\beta}(\widehat{w}(\mathbf{x})) \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}),\\
&\widehat{R}_S(\boldsymbol{h}, \mathbf{y}_{K+1}) = \frac{1}{n} \sum_{(\mathbf{x},\mathbf{y})\in S} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}),
\end{aligned}$$

here $\widehat{Q}_T^{\tau,\beta}$ is defined in Eq. (6.2.8) and γ is defined in Eq. (6.2.6).

Theorem 9 implies that $(1-\alpha)\widehat{R}_{S,T}^{\tau,\beta}(\boldsymbol{h})$ can approximate $R_Q^\alpha(\boldsymbol{h})$ uniformly.

Theorem 9. *Assume assumption 22 holds, the feature space $\mathcal{X}$ is compact and the hypothesis space $\mathcal{H} \subset \{\boldsymbol{h} : \mathcal{X} \to \mathbb{R}^{K+1}\}$ has finite Natarajan dimension [57]. Let the RKHS $\mathcal{H}_k$ be the Hilbert space with gaussian kernel. Suppose that loss function is bounded by c, the density $r \in \mathcal{H}_k$ and set the regularization parameter $\lambda = \lambda_{n,m}$ in* KuLSIF *(see Eq. (6.2.7)) such that*

$$\lim_{n,m\to 0} \lambda_{n,m} = 0, \quad \lambda_{n,m}^{-1} = O(\min\{n, m\}^{1-\delta}),$$

where $0 < \delta < 1$ *is any constant, then for any* $0 \leq \alpha < 1$,

$$\begin{aligned}&\sup_{\boldsymbol{h}\in\mathcal{H}} |(1-\alpha)\widehat{R}^{\tau,\beta}_{S,T}(\boldsymbol{h}) - R^{\alpha}_{Q}(\boldsymbol{h})| \\ \leq& c\big(\max\{1, \frac{\beta}{\tau}\} + \beta\big) O_p(\lambda^{\frac{1}{2}}_{n,m}) + \gamma c\beta U(0 < r \leq 2\tau),\end{aligned}$$

where O_p *denotes the probabilistic order,* $\gamma = 1-\alpha, \beta = \frac{\alpha}{\gamma U(r=0)}$, $\widehat{R}^{\tau,\beta}_{S,T}(\boldsymbol{h})$ *is defined in Eq. (6.2.10), and* $R^{\alpha}_{Q}(\boldsymbol{h})$ *is the IAD risk defined in Eq. (6.2.9).*

Note that $U(0 < r \leq 2\tau) \to 0$, if $\tau \to 0$, and Theorem 9 has indicated that if we omit the term $(1-\alpha)c\beta U(0 < p/q \leq 2\tau)$ and set $m \geq n$, the gap between $(1-\alpha)\widehat{R}^{\tau,\beta}_{S,T}(\boldsymbol{h})$ and $R^{\alpha}_{Q}(\boldsymbol{h})$ is close to $O_p(1/\sqrt{n})$ by choosing a small δ.

6.2.5 Main Theoretical Results

In this subsection, we analyze the relationship between $R^{\alpha}_{P}(\boldsymbol{h})$ and $\widehat{R}^{\tau,\beta}_{S,T}(\boldsymbol{h})$ based on Theorems 7, 8 and 9.

Theorem 10 (Uniform Bound Based on Transfer Learning)**.** *Given the same conditions and assumptions in Theorems* 7 *and* 9*, then for any* $0 \leq \alpha < 1$, $\boldsymbol{h} \in \mathcal{H}$,

$$\begin{aligned}&|(1-\alpha)\widehat{R}^{\tau,\beta}_{S,T}(\boldsymbol{h}) - R^{\alpha}_{P}(\boldsymbol{h})| \\ \leq& c\big(\max\{1, \frac{\beta}{\tau}\} + \beta\big) O_p(\lambda^{\frac{1}{2}}_{n,m}) + \gamma c\beta U(0 < r \leq 2\tau) \\ &+\alpha d^{\ell}_{\mathcal{H}}(P_{X|Y=\mathbf{y}_{K+1}}, Q_{X|Y=\mathbf{y}_{K+1}}) + \alpha\Lambda,\end{aligned}$$

where $\lambda_{n,m}$ *is defined in Theorem* 9, $\gamma = 1-\alpha, \beta = \frac{\alpha}{\gamma U(r=0)}$, $\widehat{R}^{\tau,\beta}_{S,T}(\boldsymbol{h})$ *is defined in Eq. (6.2.10),* $d^{\ell}_{\mathcal{H}}$ *is the* $\mathcal{H}\Delta\mathcal{H}$*-divergence defined in definition 8,* Λ *is the combined risk defined in Eq. (6.2.3) and* O_p *is the probabilistic order (independent of* c, β, τ *and* α*).*

Theorem 10 indicates that the gap between $(1-\alpha)\widehat{R}^{\tau,\beta}_{S,T}(\boldsymbol{h})$ and $R^{\alpha}_{P}(\boldsymbol{h})$ is controlled by four special terms. The combined risk Λ and domain discrepancy for unknown classes can be regarded as constants. The other two terms could be small enough, if $n, m \to +\infty$ and τ is a small value.

Theorem 11 (Estimation Error for OSL)**.** *Given the same conditions and assumptions in Theorems* 8 *and* 9*, for any* $0 \leq \alpha < 1$*, if we assume* $\widehat{\boldsymbol{h}} \in \arg\min_{\boldsymbol{h}\in\mathcal{H}} \widehat{R}^{\tau,\beta}_{S,T}(\boldsymbol{h})$*, then* $|R^{\alpha}_{P}(\widehat{\boldsymbol{h}}) - \min_{\boldsymbol{h}\in\mathcal{H}} R^{\alpha}_{P}(\boldsymbol{h})|$ *has an upper bound*

$$c\big(\max\{1, \frac{\beta}{\tau}\} + \beta\big)O_p(\lambda^{\frac{1}{2}}_{n,m}) + 4\gamma c\beta U(0 < r \leq 2\tau),$$

where $\lambda_{n,m}$ *is defined in Theorem* 9*,* $\gamma = 1 - \alpha, \beta = \frac{\alpha}{\gamma U(r=0)}$*,* $\widehat{R}^{\tau,\beta}_{S,T}(\boldsymbol{h})$ *is defined in Eq. (6.2.10) and* O_p *is the probabilistic order (independent of* c, β, τ *and* α*).*

If we select a small τ to make $U(0 < r \leq 2\tau)$ small enough and set $m \geq n$, then under some assumptions, the following optimization problem

$$\min_{\boldsymbol{h}\in\mathcal{H}} \widehat{R}^{\tau,\beta}_{S,T}(\boldsymbol{h}) \tag{6.2.11}$$

is almost *classifier-consistent* with estimation error close to $O_p(1/\sqrt{n})$. Additionally, the weight estimation in Theorem 11 is crucial. To weaken the effect of weight estimation in area supp $P_{X|Y\in\mathcal{Y}_k}$, we introduce a proxy for $\widehat{R}^{\tau,\beta}_{S,T}(\boldsymbol{h})$.

Definition 24 (Proxy of Auxiliary Risk)**.** *Given samples* S *with size* n *drawn from* $P_{XY|Y\in\mathcal{Y}_k}$ *and* T *with size* m *drawn from* U*, i.i.d., then the auxiliary risk for a hypothesis function* $\boldsymbol{h}$ *is*

$$\widetilde{R}^{\tau,\beta}_{S,T}(\boldsymbol{h}) = \widehat{R}_{S}(\boldsymbol{h}) + \frac{\alpha\gamma'}{1-\alpha}\widehat{R}^{\tau,\beta}_{S,T,u}(\boldsymbol{h}), \tag{6.2.12}$$

where $\gamma' = 1/U(r = 0)$*,* $\widehat{R}_{S}(\boldsymbol{h})$ *is defined in Definition 23,*

$$\widehat{R}^{\tau,\beta}_{S,T,u}(\boldsymbol{h}) = \frac{1}{m}\sum_{\mathbf{x}\in T} L^{-}_{\tau,\beta}(\widehat{w}(\mathbf{x}))\ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}),$$

here

$$L^{-}_{\tau,\beta}(x) = \begin{cases} x + \beta, & x \leq \tau; \\ 0, & 2\tau \leq x; \\ -\dfrac{\tau+\beta}{\tau}x + 2\tau + 2\beta, & \tau < x < 2\tau. \end{cases}$$

Then, a result similar to Theorem 11 for auxiliary risk $\widetilde{R}_{S,T}^{\tau,\beta}$ is given as follows.

Theorem 12 (Estimation Error for OSL)**.** *Given the same conditions and assumptions in Theorem* 11*, for any* $0 \leq \alpha < 1$*, if we assume* $\widetilde{\boldsymbol{h}} \in \arg\min_{\boldsymbol{h}\in\mathcal{H}} \widetilde{R}_{S,T}^{\tau,\beta}(\boldsymbol{h})$*, then* $|R_P^{\alpha}(\widetilde{\boldsymbol{h}}) - \min_{\boldsymbol{h}\in\mathcal{H}} R_P^{\alpha}(\boldsymbol{h})|$ *has an upper bound*

$$c\gamma'\big(1+\tau+\frac{\beta}{\tau}+\beta\big)O_p(\lambda_{n,m}^{\frac{1}{2}}) + 4c\gamma'\alpha\beta U(0 < r \leq 2\tau),$$

where $\lambda_{n,m}$*,* β *are introduced in Theorem 11,* γ' *and* $\widetilde{R}_{S,T}^{\tau,\beta}(\boldsymbol{h})$ *are defined in Definition 24, and* O_p *is the probabilistic order (independent of* c, β, τ, γ' *and* α*).*

6.3 A Principle Guided OSL Algorithm

Inspired by Theorem 12, we focus on the following problem

$$\min_{\boldsymbol{\Theta}} \big(\widehat{R}_S(\boldsymbol{h}_{\boldsymbol{\Theta}}) + \mu\widehat{R}_{S,T,u}^{\tau,\beta}(\boldsymbol{h}_{\boldsymbol{\Theta}})\big), \tag{6.3.1}$$

where μ is a positive parameter, $\widehat{R}_{S,T,u}^{\tau,\beta}(\boldsymbol{h})$ is defined in Eq. (6.2.12), $\boldsymbol{h}_{\boldsymbol{\Theta}}$ is a hypothesis function based on a neural network, and $\boldsymbol{\Theta}$ is parameters of the neural network. To optimize $\boldsymbol{h}_{\boldsymbol{\Theta}}$ to solve the minimization problem defined in Eq. (6.3.1), we have the following five steps.

Step 1 (Feature Encoding). Train the samples S to get a closed-set classifier $\boldsymbol{h}_{\boldsymbol{\Theta}_0}$, and designate the output of second to the last layer (without softmax) $\boldsymbol{l}$ of $\boldsymbol{h}_{\boldsymbol{\Theta}_0}$ as the encoded feature vector, i.e., $X_{\text{encoder}} = \boldsymbol{l}(X)$. The new encoded feature space is denoted as $\mathcal{X}_{\text{encoder}}$.

Step 2 (Initialize the Auxiliary Domain). Randomly generate samples T from space $\mathcal{X}_{\text{encoder}}$. By default, we generate T by uniform distribution and set the size m is $3n$. We update the samples $S = \{(\boldsymbol{l}(\mathbf{x}), \mathbf{y}) : (\mathbf{x}, \mathbf{y}) \in S\}$.

Step 3 (Construct the Auxiliary Domain). Estimate the weights $\widehat{w}$ with samples S and T as the input. The higher the weight is, the more

likely a generated sample belongs to the known classes. The parameters selection details are as follows.

Weight estimation algorithm: In the theoretical part, KuLSIF is selected to estimate weights. Kernel mean matching (KMM) [173] is also an alternative solution [174]. However, in practice, KuLSIF and KMM have time complexity $O((m+n)^2)$ [172], which may be not suitable for large datasets. The kernel bandwidth selection also impact the overall performance. Therefore, we recommend to use the outlier sample score (with range $[0,1]$) given by isolation forest (iForest) [175] as the sample weights, which has time complexity $O((n+m)\log(n+m))$. Close to 1 means known classes while close to 0 means unknown classes.

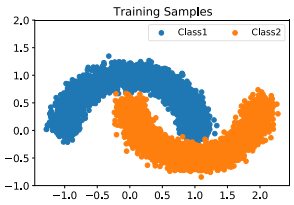
The τ is a threshold to split the generated samples T into known and unknown samples. Considering we are using iForest, based on the predicted sample score $[s_1, ..., s_m]$ (descending order), we set $\tau = s_{[t*m]}$, where $t \in (0,1)$ is the proportion that the generated samples selected as unknown samples. We set $t = 10\%$ as default.

The β and μ control jointly the importance of correctly classified unknown samples. We set μ as a dynamical parameter depending on β: $\mu = \frac{n\beta}{n'+0.0001}$, where n' is number of samples in training samples actually predicted as unknown. For example, if $\beta = 0.05$, n is 1000, there are 10 samples in training samples are predicted as unknown, then $\mu \approx 5$.

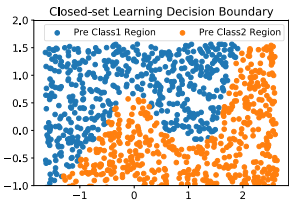
Step 4 (Softmax$_{K+1}$). Initialize an open-set learning neural network with samples S and T as the input and $K+1$ Softmax nodes as the output.

Step 5 (Open-Set Learning). Train the **Softmax**$_{K+1}$ neural network with the cost function defined in Eq. (6.3.1) with both S and T.

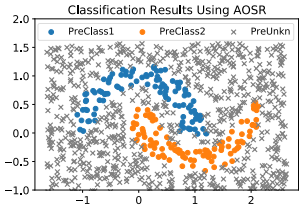
First, we implement AOSR on toy dataset with different sample size to reveal the relationship between sample size n and error $\left(O(1/\sqrt{n})\right)$. Then, we evaluate the efficacy of AOSR on benchmark datasets.



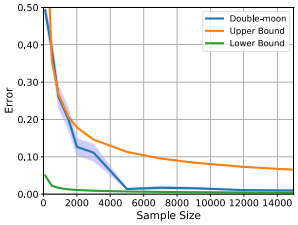
(a) Double-moon dataset



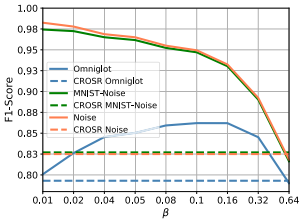
(b) Closed-set classification



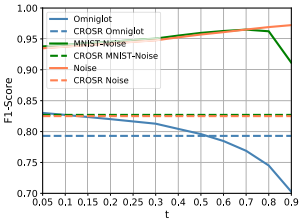
(c) Open-set classification



(d) Error and Sample Size



(e) Parameter Analysis for β



(f) Parameter Analysis for t

Figure 6.2: (a) is the training samples for double-moon dataset. (b) is the decision regions under closed-set learning setting for double-moon dataset. (c) is the decision regions under open-set learning setting for double-moon dataset. (d) is the relationship between error and sample size. (e) is parameter analysis for β. (f) is parameter analysis for t.

6.3.1 Datasets

In this chapter, we verify the efficacy of algorithm AOSR on double-moon dataset and two real world datasets:

Double-moon dataset (toy). The double-moon dataset consists of two different clusters. Samples from different clusters are regarded as known samples with different label. Samples from other region are regarded as unknown samples drawn from uniform distribution, i.i.d. The ratio between the sizes of known and unknown samples is 1.

MNIST dataset (real). Following the set up in [176], we use **MNIST** [177] as the training samples and use **Omniglot** [178], **MNIST-Noise**, and **Noise** datasets as unknown classes. **Omniglot** contains alphabet characters. **Noise** is synthesized by sampling each pixel value from a uniform distribution on $[0, 1]$. **MNIST-Noise** is synthesized by adding noise on **MNIST** as the test samples. Each dataset has $10,000$ test samples.

CIFAR-10 dataset (real). Following [176], we use **CIFAR-**10 [179] as training samples and collect unknown samples from ImageNet and

LSUN. We resized/cropped them so that they would be the same size as the known samples. Hence, we generate four datasets **ImageNet-crop**, **ImageNet-resize**, **LSUN-crop** and **LSUN-resize** as unknown classes. Each dataset has $10,000$ test samples.

Following [176], [180], we use **MNIST** [177], **SVHN** [181] and **CIFAR-10** [179] to construct different OSL tasks. For **MNIST**, **SVHN** and **CIFAR-**10, each dataset is randomly divided into 6 known classes and 4 unknown classes. In addition, we construct **CIFAR+**10 and **CIFAR+**50 by randomly selection 6 known classes and 10 or 50 unknown classes from **CIFAR-**100 [179].

6.3.2 Open-Set Learning Demonstration

Here we break down the entire learning process and demonstrate the inter-media process of each step on the toy dataset. This experiment is aiming to provide an visualization aid on understanding the open-set learning process.

To start with, we plot the double-moon dataset in Fig. 6.2(a). The objective of closed-set learning is to build a classifier that can split the samples with different labels. To achieve this goal, we build a simple neural network with sparse categorical cross-entropy as the loss function.

The closed-set learning result is shown in Fig. 6.2(b). In this case, the closed-set classifier splits the samples with different labels well. However, the closed-set classifier does not consider the boundary of support set for training domain, that is, any new samples that does not located in the support set, the closed-set classifier still gives a known label.

Fig. 6.2(c) is the open-set learning result. To recognize the unknown samples, the open-set classifier should delineate a boundary between the known and unknown classes. To achieve this goal, we use $\mathbf{Softmax}_{K+1}$ as the final output and Eq. (6.3.1) as the cost function. The AOSR will push the neural network to give label $\mathbf{y}_{K+1}$ on unknown samples.

6.3.3 Experimental Setup

AOSR has several hyper-parameters: β, t, μ and m. For all tasks, we set $m = 3n,\ t = 10\%$ as default. μ is a dynamic parameter depending on β. β is selected from 0.01 to 2.5. Details on the selection of parameters are available at `github.com/Anjin-Liu/Openset_Learning_AOSR`.

For datasets **MNIST**, Omnilot, **MNIST-Noise**, **Noise**, we use the same setting of [176] and [180] to extract the features. Same as [176], DHRNet-92 is used as the backbone for **CIFAR-**10, ImageNet and LSUN datasets. For different tasks **MNIST**, **SVHN**, **CIFAR-**10, **CIFAR+**10 and **CIFAR+**50, the backbone is the re-designed VGGNet used by [176] and [180].

We select baseline algorithms as follows: SoftMax, OpenMax [114], Counterfactual [118], CROSR [176], C2AE [119], and CGDL [180].

6.3.3.1 Evaluation

Following [176], the macro-average F1 scores are used to evaluate OSL. The area under the receiver operating characteristic (AUROC) [118] is also frequently used [118]. Note that AUROC used in [118] is suitable for global threshold-based OSL algorithms that recognize unknown samples by a fix threshold [118]. However, AOSR recognizes unknown samples based on the score of hypothesis function, thus, AOSR uses different thresholds for different samples. This implies that AUROC used in [118] may be not suitable for our algorithm. In this chapter, we use macro-average F1 scores to evaluate our algorithm.

6.3.4 Experimental Evaluation and Result Analysis

Experiment results on double-moon dataset are summarized in Fig. 6.2(d). We implement double-moon dataset with varying size n [1]. We also generate n test samples. For a different sample size, we run 100 times and report the mean accuracy and standard error in Fig. 6.2(d). Based on

[1] select n from $[1, 5, 9, 15, 20, 30, 50, 70, 90, 120, 150] * 100$

Fig. 6.2(d), the accuracy increases as the increase of training sample size n increases. When $n \to 15,000$, the accuracy approximates at 100%. In particular, the green curve $0.5/\sqrt{n}$ and the yellow curve $8/\sqrt{n}$ jointly control the curve of accuracy, implying the error of AOSR is controlled by $O(1/\sqrt{n})$.

Experiment results on real datasets are summarized in Tables 6.1, 6.2 and 6.3. For all tasks, we run AOSR 5 times and report the mean results by using F1 score [182]. In general, AOSR shows the promising performance when compared to baseline algorithms. The effectiveness of AOSR indicates that our theory is effective and practical.

Parameter analysis for β and t is given in Fig. 6.2(e), (f). We run AOSR with varying values of β, t on **MNIST** tasks. From Fig. 6.2(e), we observe that 1) when β increases from 0.01 to 0.64, the F1 scores for **Noise** and **MNIST-Noise** decrease; 2) as increasing β from 0.01 to 0.16, the F1 score for Omniglot increases. When $\beta > 1.6$, the performance for Omniglot dramatically dropped to baseline. Additionally, according to Fig. 6.2(f), we find that by changing t in the range of $[0.05, 0.30]$, AOSR achieve stable performance.

Ablation study on datasets **MNIST**, Omnilot, **MNIST-Noise** and **Noise** is shown in Table 6.4. By adjusting different components of AOSR, Table 6.4 indicates that each component of AOSR is important and necessary. Note that if we replace iForest by KMM in AOSR, the performance (0.907) is close to AOSR (0.910). This implies that KMM may be a good choice, if we omit KMM's time complexity.

6.4 Discussion

Relation with Generative Models. Algorithms based on generative models are the mainstream for OSL. CGDL [180], C2AE [119] and Counterfactual [118] are the representative works based on generative models. AOSR can be regarded as the weight-based generative model, but is very different from the mainstream generative model-based algorithms (fea-

ture map-based generative model [118, 119, 180]). Form the theoretical perspective, it is necessary to develop theory to guarantee the generalization ability of feature map-based generative models. Here we propose an interesting and important problem: *how to develop generalization theory for feature map-based generative models under open-set assumption?*

Relation with PU Learning. Positive-unlabeled learning (PU learning) [183] is a special binary classification task, which assumes only unlabeled samples and positive samples (i.e., samples with positive labels) are available. Our theory is deeply related to PU learning. If we regard the known samples S and the auxiliary samples as the positive samples and the unlabeled samples, respectively. Then, our theory degenerates into the PU learning theory.

Remaining Problems in OSL Theory. We list several interesting and important problems for OSL theory as follows.

1. How to construct weaker assumption to replace assumption 3 for achieving similar results?

2. Without assumption 3, what will happen?

3. Is it possible for OSL to achieve agnostic PAC learnability and achieve fast learning rate $O_p(1/n^a)$, for $a > 0.5$?

4. Is it possible to construct OSL theory by stability theory [184]?

6.5 Summary

This chapter mainly focuses on the learning theory for open-set learning. The generalization error bounds proved in our work provide the first almost-PAC-style guarantee on open-set learning. Based on our theory, a principle guided algorithm AOSR is proposed. Experiments on double-moon dataset indicate that the error of AOSR is controlled by order $O(1/\sqrt{n})$, which is consistent with theoretical observations. Experiments on real datasets indicate that AOSR achieves competitive performance when compared with baselines.

Table 6.1: The performance on dataset CIFAR-10 is evaluated by macro-averaged F1-scores in 11 classes (10 known classes and 1 unknown class). We report the experimental results reproduced by [176]. A larger score is better.

Algorithm	ImageNet-crop	ImageNet-resize	LSUN-crop	LSUN-resize
Softmax	0.639	0.653	0.642	0.647
Openmax [114]	0.660	0.684	0.657	0.668
LadderNet+Softmax [176]	0.640	0.646	0.644	0.647
LadderNet+Openmax [176]	0.653	0.670	0.652	0.659
DHRNet+Softmax [176]	0.645	0.649	0.650	0.649
DHRNet+Openmax [176]	0.655	0.675	0.656	0.664
Counterfactual [118]	0.636	0.635	0.650	0.648
CROSR [176]	0.721	0.735	0.720	0.749
C2AE [119]	0.837	0.826	0.783	0.801
CGDL [180]	**0.840**	**0.832**	0.806	0.812
Ours (AOSR)	0.798	0.795	**0.839**	**0.838**

Table 6.2: The performance on dataset MNIST is evaluated by macro-averaged F1-scores in 11 classes.

Algorithm	Omniglot	MNIST-Noise	Noise
Softmax	0.595	0.801	0.829
Openmax	0.780	0.816	0.826
CROSR	0.793	0.827	0.826
Ours (AOSR)	**0.825**	**0.953**	**0.953**

Table 6.3: The performance on MNIST, SVHN, CIFAR-10, CIFAR+10 and CIFAR+50 are evaluated by macro-averaged F1 scores. We report the experimental results reported by [180] .

Algorithm	MNIST	SVHN	CIFAR-10	CIFAR+10	CIFAR+50
Softmax	0.768	0.725	0.600	0.701	0.637
Openmax [114]	0.798	0.737	0.623	0.731	0.676
CROSR [176]	0.803	0.753	0.668	0.769	0.684
GDFR [113]	0.821	0.716	0.700	**0.776**	0.683
CGDL [180]	0.837	0.776	0.655	0.760	0.695
Ours (AOSR)	**0.850**	**0.842**	**0.705**	0.773	**0.706**

Table 6.4: Ablation study on dataset MNIST, Omnilot, MNIST-Noise and Noise.

Tasks	Only iForest	β=0	μ=0	w/KMM	w/KuLSIF	AOSR
Avg	0.680	0.677	0.677	0.907	0.855	**0.910**

Chapter 7

Bridging Theory and Algorithm for Semi-supervised Heterogeneous Domain Adaptation

A novel algorithm and theory for semi-supervised heterogeneous domain adaptation are introduced in this chapter.

7.1 Introduction

Homogeneous domain adaptation is based on two assumptions: 1) that the source and target domains share the same label sets; and 2) that the source and target domains share the same feature spaces. In reality, it is not easy to seek a source domain with the same feature space (i.e., homogeneous) as the target domain of interest [23–26]. Hence, a more general and challenging problem heterogeneous domain adaptation (HeDA) has also been considered, where the source and target domains are from different feature spaces [29, 30]. Depending on whether labeled samples are available in the target domain, HeDA is roughly divided into three categories: unsupervised HeDA [25, 185], supervised HeDA [24, 186], *semi-supervised HeDA* (SsHeDA) [120, 124, 187]. Of these three paradigms, SsHeDA is, arguably, the most attractive, since collecting a few labeled target samples is affordable.

The SsHeDA aims to train a target classifier with unlabeled and a few labeled target samples, and labeled source samples. The current SsHeDA algorithms bridge the heterogeneity between domains by either projecting the source and target samples onto a common feature space

using different feature transformations [120,188,189] or by projecting the source samples into the target feature space with a special source feature transformation [190–192]. Representative feature transformation techniques are spectral mapping [193], kernel matching [194] and manifold alignment [120].

Yet, despite the many SsHeDA models and algorithms that have been proposed, very little theoretical groundwork has been undertaken to reveal the nature of SsHeDA problem or why the current solutions work as they do. Zhou et al. [190] proposed the unique theoretical work providing generalization error for the algorithm *Sparse Heterogeneous Feature Representation* (SHFR) [190]. However, the generalization error for SHFR limited the loss function to hinge loss and the feature transformation to linear mapping. Besides, the proposed generalization error does not provide an explanation as to why the labeled source and unlabeled target samples can help reduce the need for labeled target samples. Providing such a theory is our main purpose.

To develop the SsHeDA theory, a straightforward idea is to simply extend the semi-supervised HoDA (SsHoDA) theory to the heterogeneous situation by introducing feature transformations. Existing SsHoDA theory is based on theoretical analysis of a *weighted sum of source and target risks* (weighted risk). Researchers provided a uniform bound on the target risk of a classifier trained to minimize the weighted risk. The bound has shown that the need of labeled target samples is reduced by reducing the weight of the target risk [195]. However, an obstacle appears in the heterogeneous situation. The combined risk [53, 195], as a constant term in the SsHoDA uniform bound, becomes a function related to feature transformations, which means an estimation is not possible without sufficient labeled target samples (see Section 7.3.3).

Motivated by the compatibility condition introduced by probably approximately correct (PAC) theory [196], we therefore develop a theory of SsHeDA from a thoroughly new perspective compared to previous domain adaptation theories. Our bold strategy is to explain why the

SsHeDA problem can be addressed by proving a novel generalization error of SsHeDA. By reducing the size of the target feature transformation space, the generalization error illustrates how the labeled source and unlabeled target samples, together with a suitable compatibility condition, can reduce the need for labeled target samples.

Guided by our theory, we then devise a SsHeDA algorithm to bring the proposed SsHeDA theory into the reality. *Kernel Heterogeneous Domain Alignment* (KHDA) is a kernel method designed for small datasets. KHDA maintains two main branches, where the first branch aims to transfer knowledge from the source to the target domain, and the second branch aims to transfer knowledge from the labeled target samples to the unlabeled target samples.

Our algorithm is proved to be highly accurate in a set of experiments comprising 9 baseline algorithms, 60 object recognition tasks and 26 text classification tasks. Extensive experiments demonstrate that KHDA achieves competitive performance compared with all the baselines.

Our contributions are summarized as follows:

• We introduce the concepts of compatibility, transfer error rates and uniform sample complexity as new tools for estimating the need for labeled target samples. Co-opting these concepts provides a thoroughly new perspective for theoretically analyzing domain adaptation problems.

• We propose a generalization error estimation for target risk in SsHeDA. To the best of our knowledge, this is the first work on SsHeDA to explain why combining labeled source samples with unlabeled target samples can reduce the need for labeled target samples.

• We develop a SsHeDA algorithm based on our theoretical work: KHDA. KHDA is a theoretical-guided algorithm that depends on the kernel method.

7.2 Problem Setting and Concepts

In this section, we recall the definition of SsHeDA problem and introduce some important notations used in this thesis.

7.2.1 Problem Setting

Let $\mathcal{X}_s \subset \mathbb{R}^{d_s}$, $\mathcal{X}_t \subset \mathbb{R}^{d_t}$ be the feature (input) spaces and $\mathcal{Y} = \{\mathbf{y}_1, ..., \mathbf{y}_K\}$ be a label (output) space. Then, we propose the SsHeDA problem as follows: Given sets of samples called the labeled source, labeled target and unlabeled target samples:

$$\begin{aligned} \mathcal{S} &= \{(\mathbf{x}_s^i, \mathbf{y}_s^i)\}_{i=1}^{n_s} \sim P_{X_s Y_s} \ i.i.d. \\ \mathcal{T}_l &= \{(\mathbf{x}_l^i, \mathbf{y}_l^i)\}_{i=1}^{n_l} \sim P_{X_t Y_t} \ i.i.d. \\ \mathcal{T}_u &= \{\mathbf{x}_u^i\}_{i=1}^{n_u} \sim P_{X_t} \quad i.i.d. \end{aligned}$$

where $n_l \ll n_u$, $n_l \ll n_s$. The aim of semi-supervised heterogeneous domain adaptation is to train a classifier $g : \mathcal{X}_t \to \mathcal{Y}$ such that g classifies the unlabeled target samples accurately by using labeled and unlabeled samples. For simplicity, let

$$\mathbf{X}_s = [\mathbf{x}_s^1, ..., \mathbf{x}_s^{n_s}], \quad \mathbf{X}_l = [\mathbf{x}_l^1, ..., \mathbf{x}_l^{n_l}], \quad \mathbf{X}_u = [\mathbf{x}_u^1, ..., \mathbf{x}_u^{n_u}],$$

and let $\mathbf{X}_s^c \in \mathbb{R}^{d_s \times n_s^c}$ and $\mathbf{X}_l^c \in \mathbb{R}^{d_t \times n_l^c}$ be the sub-matrices of $\mathbf{X}_s$ and $\mathbf{X}_l$, whose column vectors are samples with label c. Let $\mathbf{X}_t \in \mathbb{R}^{d_t \times n_t}$ be $[\mathbf{X}_l, \mathbf{X}_u]$. Here n_s^c, n_l^c are the number of labeled source and target samples with label c, and $n_t = n_l + n_u$. For convenience, given any data matrix $\mathbf{X}$, $\mathbf{x} \in \mathbf{X}$ means that $\mathbf{x}$ is a column vector of $\mathbf{X}$.

7.2.2 Concepts and Notations

Before introducing our main results, we need to introduce the following concepts and notations.

7.2.2.1 Empirical distributions and Transformations

We denote the notation $\widehat{P}_{\mathbf{X}}$ as the corresponding empirical distribution over any samples matrix $\mathbf{X} = [\mathbf{x}^1, ..., \mathbf{x}^n]$:

$$\widehat{P}_{\mathbf{X}} = \frac{1}{n} \sum_{i=1}^{n} \delta_{\mathbf{x}^i},$$

where $\delta_{\mathbf{x}^i}$ is the Dirac measure defined in $\mathbf{x}^i$. For example, $\widehat{P}_{\mathbf{X}_s}$ is the empirical distribution corresponding to $\mathbf{X}_s$.

Given a latent space $\mathcal{X} \subset \mathbb{R}^d$, we denote

$$\mathcal{F}_s \subset \{\boldsymbol{T} : \mathcal{X}_s \to \mathcal{X}\}, \quad \mathcal{F}_t \subset \{\boldsymbol{T} : \mathcal{X}_t \to \mathcal{X}\}$$

as source and target transformation spaces, respectively. Given a transformation $\boldsymbol{T}$, we define the transformed data matrix as

$$\boldsymbol{T}(\mathbf{X}) = [\boldsymbol{T}(\mathbf{x}^1), ..., \boldsymbol{T}(\mathbf{x}^n)].$$

7.2.2.2 Hypothesis Space and Risks

We consider a multi-class classification task with a hypothesis space $\mathcal{H}$ consisting of scoring functions

$$\begin{aligned} \boldsymbol{h} : \ \mathcal{X} &\to \mathbb{R}^{1\times|\mathcal{Y}|} = \mathbb{R}^{1\times K} \\ \mathbf{x} &\to [h_1(\mathbf{x}), ..., h_K(\mathbf{x})], \end{aligned}$$

where $h_c(\mathbf{x})$ $(c = 1, ..., K)$ indicates the confidence in the prediction of label c. Given $\ell : \mathbb{R}^K \times \mathbb{R}^K \to \mathbb{R}_{\geq 0}$ as the symmetric loss function, the risks of $\boldsymbol{h} \in \mathcal{H}$ w.r.t. ℓ under $P_{\boldsymbol{T}_s(X_s)Y_s}$ and $P_{\boldsymbol{T}_t(X_t)Y_t}$ are given by

$$\begin{aligned} R_s(\boldsymbol{h} \circ \boldsymbol{T}_s) &= \mathbb{E}\ \ell(\boldsymbol{h} \circ \boldsymbol{T}_s(X_s), Y_s), \\ R_t(\boldsymbol{h} \circ \boldsymbol{T}_t) &= \mathbb{E}\ \ell(\boldsymbol{h} \circ \boldsymbol{T}_t(X_t), Y_t), \end{aligned}$$

It is convenient to use notations $\widehat{R}_s(\boldsymbol{h} \circ \boldsymbol{T}_s)$ and $\widehat{R}_t(\boldsymbol{h} \circ \boldsymbol{T}_t)$ to represent the empirical risks corresponding to the risks $R_s(\boldsymbol{h} \circ \boldsymbol{T}_s)$ and $R_t(\boldsymbol{h} \circ \boldsymbol{T}_t)$, respectively.

7.3 Theoretical foundation for SsHeDA

This section presents two necessary concepts, then reviews the existing SsHoDA theory and discusses the main obstacle to extend the SsHoDA theory into the heterogeneous situation. Lastly, we introduce our main

theoretical results.

7.3.1 Uniform Sample Complexity

Let ψ be the function from $\mathcal{H} \times \mathcal{F}_s \times \mathcal{F}_t \times \mathcal{P}_s \times \mathcal{P}_t \times \mathcal{P}_t(X)$ to $\mathbb{R}$, where $\mathcal{P}_s, \mathcal{P}_t, \mathcal{P}_t(X)$ are probability spaces over spaces $\mathcal{X}_s \times \mathcal{Y}$, $\mathcal{X}_t \times \mathcal{Y}$ and $\mathcal{X}_t$, respectively.

Given any distributions $P_{X_sY_s} \in \mathcal{P}_s$, $P_{X_lY_l} \in \mathcal{P}_t$ and $P_{X_t} \in \mathcal{P}_t(X)$ (in SsHeDA, $P_{X_lY_l} = P_{X_tY_t}$), if random samples $\mathcal{S}$ with size n_s , $\mathcal{T}_l$ with size n_l and $\mathcal{T}_u$ with size n_u are drawn from $P_{X_sY_s}$, $P_{X_lY_l}$ and P_{X_t}, i.i.d., respectively, for any $0 < \delta < 1$, $\epsilon > 0$, we denote

$$m_s^{\psi}(\epsilon, \delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t),\ m_l^{\psi}(\epsilon, \delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t),\ m_u^{\psi}(\epsilon, \delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t),$$

as the smallest numbers of samples such that if

$$\begin{aligned} n_s &> m_s^{\psi}(\epsilon, \delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t), \\ n_l &> m_l^{\psi}(\epsilon, \delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t), \\ n_u &> m_u^{\psi}(\epsilon, \delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t), \end{aligned}$$

with a probability of at least $1 - \delta > 0$, then for any $\boldsymbol{h} \in \mathcal{H}$, $\boldsymbol{T}_s \in \mathcal{F}_s$ and $\boldsymbol{T}_t \in \mathcal{F}_t$, we have $|\psi - \widehat{\psi}| < \epsilon$, where

$$\begin{aligned} \psi &= \psi(\boldsymbol{h}, \boldsymbol{T}_s, \boldsymbol{T}_t, P_{X_sY_s}, P_{X_lY_l}, P_{X_t}), \\ \widehat{\psi} &= \psi(\boldsymbol{h}, \boldsymbol{T}_s, \boldsymbol{T}_t, \widehat{P}_{\mathcal{S}}, \widehat{P}_{\mathcal{T}_l}, \widehat{P}_{\mathcal{T}_u}), \end{aligned}$$

here $\widehat{P}_{\mathcal{S}}, \widehat{P}_{\mathcal{T}_l}, \widehat{P}_{\mathcal{T}_u}$ are empirical distributions corresponding to $\mathcal{S}$, $\mathcal{T}_l$ and $\mathcal{T}_u$.

If the smallest sample numbers are finite, then we say ψ has uniform estimation. It is clear that

1. $m_l^{\psi}(\epsilon, \delta, \mathcal{H}_1, \mathcal{F}_s, \mathcal{F}_t) \leq m_l^{\psi}(\epsilon, \delta, \mathcal{H}_2, \mathcal{F}_s, \mathcal{F}_t)$, if $\mathcal{H}_1 \subset \mathcal{H}_2$;
2. $m_l^{\psi}(\epsilon, \delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_1) \leq m_l^{\psi}(\epsilon, \delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_2)$, if $\mathcal{F}_1 \subset \mathcal{F}_2$.

7.3.2 Compatibility and Transfer Error Rate

Compatibility is proposed by [196] to develop the PAC-model style framework for semi-supervised learning. By using the notation of compatibility, the semi-supervised PAC theoretical model provides a unified framework for analyzing why unlabeled samples can help to reduce the need for labeled samples.

To investigate how the source samples and unlabeled target samples reduce the need for labeled target samples in the SsHeDA problem, we define the SsHeDA compatibility.

Definition 25. *Given hypothesis space $\mathcal{H}$, transformation spaces $\mathcal{F}_s, \mathcal{F}_t$ and probability spaces $\mathcal{P}_s$ and $\mathcal{P}_t(X)$ over spaces $\mathcal{X}_s \times \mathcal{Y}$, $\mathcal{X}_t$, respectively, the heterogeneous domain adaptation compatibility is a function $\chi : \mathcal{H} \times \mathcal{F}_s \times \mathcal{F}_t \times \mathcal{P}_s \times \mathcal{P}_t(X) \to [0, 1]$.*

Definition 26 (Transfer Error Rate)**.** *Given the HeDA compatibility χ, the incompatibility of $\boldsymbol{h}, \boldsymbol{T}_s, \boldsymbol{T}_t$ with distributions $P_{X_sY_s}$ and P_{X_t} is*

$$1 - \chi(\boldsymbol{h}, \boldsymbol{T}_s, \boldsymbol{T}_t, P_{X_sY_s}, P_{X_t}),$$

which is also called the transfer error rate, $\mathrm{err}(\boldsymbol{h}, \boldsymbol{T}_s, \boldsymbol{T}_t)$*, when χ, $P_{X_sY_s}$, P_{X_t} are clear from the context.*

For given samples $\mathcal{S} \sim P_{X_sY_s}$, $\mathcal{T}_u \sim P_{X_t}$, we use $\widehat{\mathrm{err}}(\boldsymbol{h}, \boldsymbol{T}_s, \boldsymbol{T}_t)$ to denote $1 - \chi(\boldsymbol{h}, \boldsymbol{T}_s, \boldsymbol{T}_t, \widehat{P}_{\mathcal{S}}, \widehat{P}_{\mathcal{T}_u})$ as the empirical form of $\mathrm{err}(\boldsymbol{h}, \boldsymbol{T}_s, \boldsymbol{T}_t)$.

Then, we define the hypothesis function and target transformation whose incompatibility is at most a given value α:

Definition 27. *Given the threshold $\alpha \geq 0$, $\mathcal{H}(\alpha), \mathcal{F}_t(\alpha)$ are*

$$\{\boldsymbol{h} \in \mathcal{H} \mid \exists\, \boldsymbol{T}_s \in \mathcal{F}_s,\ \boldsymbol{T}_t \in \mathcal{F}_t,\ \text{s.t. } \mathrm{err}(\boldsymbol{h}, \boldsymbol{T}_s, \boldsymbol{T}_t) \leq \alpha\},$$
$$\{\boldsymbol{T}_t \in \mathcal{F}_t \mid \exists\, \boldsymbol{T}_s \in \mathcal{F}_s,\ \boldsymbol{h} \in \mathcal{H},\ \text{s.t. } \mathrm{err}(\boldsymbol{h}, \boldsymbol{T}_s, \boldsymbol{T}_t) \leq \alpha\}.$$

Remark 8. *It is clear that $\mathcal{H}(\alpha) \subset \mathcal{H}, \mathcal{F}_t(\alpha) \subset \mathcal{F}$.*

Next, we need an assumption to estimate the difference between $\widehat{\mathrm{err}}(\boldsymbol{h}, \boldsymbol{T}_s, \boldsymbol{T}_t)$ and $\mathrm{err}(\boldsymbol{h}, \boldsymbol{T}_s, \boldsymbol{T}_t)$.

Assumption 4. *Given spaces $\mathcal{H}$, $\mathcal{F}_s$, $\mathcal{F}_t$ and transfer error rate* $\mathrm{err}(\boldsymbol{h}, \boldsymbol{T}_s, \boldsymbol{T}_t)$, *for any $\epsilon > 0$ and $0 < \delta < 1$,*

$$m_s^{\mathrm{err}}(\epsilon, \delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t) < +\infty,$$
$$m_u^{\mathrm{err}}(\epsilon, \delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t) < +\infty,$$
$$m_l^{\mathrm{err}}(\epsilon, \delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t) = 0.$$

Lastly, we introduce an example of transfer error rate.

Example 2. *One can set* $\mathrm{err}(\boldsymbol{h}, \boldsymbol{T}_s, \boldsymbol{T}_t)$ *as*

$$(1-\tau)R_s(\boldsymbol{h} \circ \boldsymbol{T}_s)/B + \tau d^{\ell}_{\boldsymbol{h},\mathcal{H}}(P_{\boldsymbol{T}_s(X_s)}, P_{\boldsymbol{T}_t(X_t)})/B,$$

where $d^{\ell}_{\boldsymbol{h},\mathcal{H}}$ is disparity discrepancy defined in Definition 8, τ is the weight and B is the supremum of ℓ. If $\mathcal{H} \circ \mathcal{F}_s, \mathcal{H} \circ \mathcal{F}_t$ have finite Natarajan dimension, then $\mathrm{err}(\boldsymbol{h}, \boldsymbol{T}_s, \boldsymbol{T}_t)$ *satisfies assumption* 4.

7.3.3 Obstacle to Extend SsHoDA Theory

Here we introduce the obstacle to extend SsHoDA theory in HeSsDA.

In SsHoDA theory [195], weighted risk is defined as

$$R_\beta(\boldsymbol{h}) = \beta R_t(\boldsymbol{h}) + (1-\beta)R_s(\boldsymbol{h}),$$

where β $(0 < \beta < 1)$ is the weight. Utilizing weighted risk, the following theorem shows that the number of labeled target samples can be reduced in homogeneous situation.

Theorem 13. *Let ℓ be the symmetric loss satisfying the triangle inequality, feature spaces $\mathcal{X}_s, \mathcal{X}_t$ be space $\mathcal{X}$, and $\mathcal{F}_s$, $\mathcal{F}_t$ be $\{\mathbf{I}\}$, where $\mathbf{I}$ is identical mapping from $\mathcal{X}$ to $\mathcal{X}$. If random labeled source samples with size n_s are generated by $P_{X_sY_s}$, i.i.d., random labeled target samples with size n_l are generated by $P_{X_tY_t}$, i.i.d., and unlabeled target samples with size n_u are generated by P_{X_t}, i.i.d., for any $0 < \delta < 1$, $0 < \gamma_1$,$\gamma_2 < 1$,*

and $\epsilon > 0$*, if*

$$\begin{aligned} n_s > \max\{&m_s^{d_{\mathcal{H}}^\ell}(\frac{\gamma_1\epsilon}{2(1-\beta)}, \frac{\gamma_2\delta}{2}, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t), \\ &m_s^{R_s}(\frac{\gamma_1\epsilon}{2(1-\beta)}, \frac{\gamma_2\delta}{2}, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t)\}, \\ n_u > &m_u^{d_{\mathcal{H}}^\ell}(\frac{\gamma_1\epsilon}{2(1-\beta)}, \frac{\gamma_2\delta}{2}, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t), \\ n_l > &m_l^{R_t}(\frac{(1-\gamma_1)\epsilon}{2\beta}, (1-\gamma_2)\delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t), \end{aligned}$$

where $d_{\mathcal{H}}^\ell$ *is* $\mathcal{H}\Delta\mathcal{H}$*-divergence defined in Definition 7,* R_s, R_t *are* $R_s(\boldsymbol{h}), R_t(\boldsymbol{h})$*, then with a probability at least* $1-\delta > 0$*, for any* $\boldsymbol{h} \in \mathcal{H}$*,*

$$R_t(\widehat{\boldsymbol{h}}) \leq R_t(\boldsymbol{h}_t) + 2(1-\beta)\big(d_{\mathcal{H}}^\ell(\widehat{P}_{\mathbf{X}_s}, \widehat{P}_{\mathbf{X}_u}) + \Lambda\big) + \epsilon,$$

where $\widehat{\boldsymbol{h}} \in \arg\min_{\boldsymbol{h}\in\mathcal{H}} \widehat{R}_\beta(\boldsymbol{h})$*,* $\boldsymbol{h}_t \in \arg\min_{\boldsymbol{h}\in\mathcal{H}} \widehat{R}_t(\boldsymbol{h})$ *and* Λ *is known as the combined risk.*

Before proving Theorem 13, we introduce two necessary lemmas.

Lemma 14.

$$|R_\beta(\boldsymbol{h}) - R_t(\boldsymbol{h})| \leq (1-\beta)\big(d_{\mathcal{H}}^\ell(P_{X_s}, P_{X_t}) + \Lambda\big),$$

where $\Lambda = \min_{\boldsymbol{h}\in\mathcal{H}} \big(R_s(\boldsymbol{h}) + R_t(\boldsymbol{h})\big)$.

Proof. $|R_\beta(\boldsymbol{h}) - R_t(\boldsymbol{h})| = (1-\beta)|R_s(\boldsymbol{h}) - R_t(\boldsymbol{h})| \leq (1-\beta)\big(d_{\mathcal{H}}^\ell(P_{X_s}, P_{X_t}) + \Lambda\big)$. □

Given finite samples, we estimate the difference between $|\widehat{R}_\beta(\boldsymbol{h}) - R_\beta(\boldsymbol{h})|$.

Lemma 15. *If*

$$\begin{aligned} n_s > &m_s^{R_s}(\gamma_1\frac{\epsilon}{1-\beta}, \gamma_2\delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t), \\ n_l > &m_l^{R_t}((1-\gamma_1)\frac{\epsilon}{\beta}, (1-\gamma_2)\delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t), \end{aligned}$$

then

$$|R_\beta(\boldsymbol{h}) - \widehat{R}_\beta(\boldsymbol{h})| < \epsilon,$$

for any $\boldsymbol{h} \in \mathcal{H}$*, with a probability of at least* $1 - \delta > 0$*, where* $\mathcal{F}_s = \mathcal{F}_t = \{\mathbf{I}\}$*, here* $\mathbf{I}$ *is the identical mapping.*

Proof. If

$$n_s > m_s^{R_s}(\gamma_1 \frac{\epsilon}{1-\beta}, \gamma_2\delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t),$$

with a probability of at least $1 - \gamma_2\delta$, we have

$$(1-\beta)|R_s(\boldsymbol{h}) - \widehat{R}_s(\boldsymbol{h})| < \gamma_1\epsilon.$$

If

$$n_l > m_l^{R_t}((1-\gamma_1)\frac{\epsilon}{\beta}, (1-\gamma_2)\delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t),$$

with a probability of at least $1 - (1-\gamma_2)\delta$, we have

$$\beta|R_t(\boldsymbol{h}) - \widehat{R}_t(\boldsymbol{h})| < \gamma_1\epsilon.$$

Hence, with a probability of at least $1 - \delta$, we have

$$|R_\beta(\boldsymbol{h}) - \widehat{R}_\beta(\boldsymbol{h})| \leq (1-\beta)|R_s(\boldsymbol{h}) - \widehat{R}_s(\boldsymbol{h})| + \beta|R_t(\boldsymbol{h}) - \widehat{R}_t(\boldsymbol{h})| < \epsilon.$$

□

Then, we prove Theorem 13.

Proof of Theorem 13. **Step 1.** If

$$\begin{aligned}
&n_s > \max\{m_s^{d_{\mathcal{H}}^\ell}(\frac{\gamma_1\epsilon}{2(1-\beta)}, \delta_1, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t), m_s^{R_s}(\frac{\gamma_1\epsilon}{2(1-\beta)}, \delta_2, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t)\},\\
&n_u > m_u^{d_{\mathcal{H}}^\ell}(\frac{\gamma_1\epsilon}{2(1-\beta)}, \delta_1, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t),\\
&n_l > m_l^{R_t}(\frac{(1-\gamma_1)\epsilon}{2\beta}, \delta_3, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t),
\end{aligned}$$

then with a probability of at least $1-\delta_1-\delta_2-\delta_3$

$$\begin{aligned} R_t(\widehat{\boldsymbol{h}}) &\le R_\beta(\widehat{\boldsymbol{h}}) + (1-\beta)\big(d_{\mathcal{H}}^{\ell}(P_{X_s}, P_{X_t}) + \Lambda\big) && \text{Lemma 14} \\ &< \widehat{R}_\beta(\widehat{\boldsymbol{h}}) + (1-\beta)\big(d_{\mathcal{H}}^{\ell}(\widehat{P}_{\mathbf{X}_s}, \widehat{P}_{\mathbf{X}_u}) + \Lambda\big) + \frac{\epsilon}{2} && \text{Lemma 15.} \end{aligned}$$

Step 2. If

$$\begin{aligned} &n_s > \max\{m_s^{d_{\mathcal{H}}^{\ell}}(\frac{\gamma_1\epsilon}{2(1-\beta)}, \delta_1, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t), m_s^{R_s}(\frac{\gamma_1\epsilon}{2(1-\beta)}, \delta_2, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t)\}, \\ &n_u > m_u^{d_{\mathcal{H}}^{\ell}}(\frac{\gamma_1\epsilon}{2(1-\beta)}, \delta_1, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t), \\ &n_l > m_l^{R_t}(\frac{(1-\gamma_1)\epsilon}{2\beta}, \delta_3, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t), \end{aligned}$$

then with a probability of at least $1-\delta_1-\delta_2-\delta_3$,

$$\begin{aligned} &\widehat{R}_\beta(\widehat{\boldsymbol{h}}) + (1-\beta)\big(d_{\mathcal{H}}^{\ell}(\widehat{P}_{\mathbf{X}_s}, \widehat{P}_{\mathbf{X}_u}) + \Lambda\big) \\ \le &\widehat{R}_\beta(\boldsymbol{h}_t) + (1-\beta)\big(d_{\mathcal{H}}^{\ell}(\widehat{P}_{\mathbf{X}_s}, \widehat{P}_{\mathbf{X}_u}) + \Lambda\big) \\ < &R_\beta(\boldsymbol{h}_t) + (1-\beta)\big(d_{\mathcal{H}}^{\ell}(\widehat{P}_{\mathbf{X}_s}, \widehat{P}_{\mathbf{X}_u}) + \Lambda\big) + \frac{\epsilon}{2} \quad \text{Lemma 15} \\ \le &R_t(\boldsymbol{h}_t) + 2(1-\beta)\big(d_{\mathcal{H}}^{\ell}(\widehat{P}_{\mathbf{X}_s}, \widehat{P}_{\mathbf{X}_u}) + \Lambda\big) + \frac{\epsilon}{2} \quad \text{Lemma 14} \end{aligned}$$

Step 3. Combining **Step 1** with **Step 2**, we have

$$\begin{aligned} &n_s > \max\{m_s^{d_{\mathcal{H}}^{\ell}}(\frac{\gamma_1\epsilon}{2(1-\beta)}, \delta_1, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t), m_s^{R_s}(\frac{\gamma_1\epsilon}{2(1-\beta)}, \delta_2, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t)\}, \\ &n_u > m_u^{d_{\mathcal{H}}^{\ell}}(\frac{\gamma_1\epsilon}{2(1-\beta)}, \delta_1, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t), \\ &n_l > m_l^{R_t}(\frac{(1-\gamma_1)\epsilon}{2\beta}, \delta_2, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t), \end{aligned}$$

then with a probability of at least $1-\delta_1-\delta_2-\delta_3$ (note that do not $1-2\delta_1-2\delta_2-2\delta_3$, because we have used the property of uniform estimation),

$$R_t(\widehat{\boldsymbol{h}}) < R_t(\boldsymbol{h}_t) + 2(1-\beta)\big(d_{\mathcal{H}}^{\ell}(\widehat{P}_{\mathbf{X}_s}, \widehat{P}_{\mathbf{X}_u}) + \Lambda\big) + \epsilon.$$

Let $\delta_1 = \delta_2 = \gamma_2\delta/2$ and $\delta_3 = (1-\gamma_2)\delta$, Theorem 13 is completed. □

Remark 9. *The number of labeled target samples*

$$m_l^{R_t}(\frac{(1-\gamma_1)\epsilon}{2\beta}, (1-\gamma_2)\delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t)$$

is reduced, with a decrease of β*. Especially, if we set* $\beta \to 0$*, then* $m_l^{R_t}(\frac{(1-\gamma_1)\epsilon}{2\beta}, (1-\gamma_2)\delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t) \to 0$*. Hence, we can reduce the number of labeled target samples by adjusting* β*.*

It is natural for us to extend the above theorem to the heterogeneous situation by using the weighted risk and transformations $\boldsymbol{T}_s$ and $\boldsymbol{T}_t$. However, the combined risk Λ results in the main obstacle. In heterogeneous situation, Λ is formulated as

$$\min_{\boldsymbol{h}\in\mathcal{H}} \big(R_s(\boldsymbol{h}\circ\boldsymbol{T}_s) + R_t(\boldsymbol{h}\circ\boldsymbol{T}_t)\big). \tag{7.3.1}$$

Eq. (7.3.1) shows that Λ is a function on variables $\boldsymbol{T}_s, \boldsymbol{T}_t$, hence, it is not a fixed value. To estimate Λ using finite samples, the labeled target samples are indispensable. The following theorem provides a reason why Λ is the main obstacle.

Theorem 16. *There exist hypothesis space* $\mathcal{H}$ *and transformation spaces* $\mathcal{F}_s$*,* $\mathcal{F}_t$ *such that for any* $\epsilon > 0$ *and* $0 < \delta < 1$*,*

$$m_l^{\Lambda}(\epsilon, \delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t) \geq m_l^{R_t}(\epsilon, \delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t),$$

where Λ *is defined in Eq. (7.3.1) and* R_t *is* $R_t(\boldsymbol{h}\circ\boldsymbol{T}_t)$*.*

Proof of Theorem 16. We set $\mathcal{H}$ and $\mathcal{F}_s$ satisfying the following conditions

$$\mathcal{H}\circ\mathcal{F}_s = \{\text{a constant value mapping}\}, \quad |\mathcal{H}| = 1.$$

Then,

$$\Lambda = \min_{\boldsymbol{h}\in\mathcal{H}} R_s(\boldsymbol{h}\circ\boldsymbol{T}_s) + R_t(\boldsymbol{h}\circ\boldsymbol{T}_t) = R_s(\boldsymbol{h}\circ\boldsymbol{T}_s) + R_t(\boldsymbol{h}\circ\boldsymbol{T}_t).$$

To estimate $|\Lambda - \widehat{\Lambda}|$,

$$|\Lambda - \widehat{\Lambda}| = |R_t(\boldsymbol{h} \circ \boldsymbol{T}_t) - \widehat{R}_t(\boldsymbol{h} \circ \boldsymbol{T}_t)|.$$

Hence, we have

$$m_l^{\Lambda}(\epsilon, \delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t) = m_l^{R_t}(\epsilon, \delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t).$$

We have completed the proof.

We also provide another proof as follows. Let the space $\mathcal{F}_s$ be large enough: for any $\boldsymbol{h} \in \mathcal{H}$ and $\boldsymbol{T}_t \in \mathcal{F}_t$, there exists $\boldsymbol{T}_s \in \mathcal{F}_s$ such that

$$\big(R_s(\boldsymbol{h} \circ \boldsymbol{T}_s) - \widehat{R}_s(\boldsymbol{h} \circ \boldsymbol{T}_s)\big)\big(R_t(\boldsymbol{h} \circ \boldsymbol{T}_t) - \widehat{R}_t(\boldsymbol{h} \circ \boldsymbol{T}_t)\big) \geq 0.$$

and

$$|\mathcal{H}| = 1,$$

then we can check that

$$|\Lambda - \widehat{\Lambda}| \geq |R_t(\boldsymbol{h} \circ \boldsymbol{T}_t) - \widehat{R}_t(\boldsymbol{h} \circ \boldsymbol{T}_t)|,$$

which implies

$$m_l^{\Lambda}(\epsilon, \delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t) \geq m_l^{R_t}(\epsilon, \delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t).$$

□

Note that there is a coefficient $2(1-\beta)$ of Λ in the bound of Theorem 13. Hence, to estimate $2(1-\beta)\Lambda$ given ϵ and δ, the number of labeled target samples needed is at least

$$m_l^{\Lambda}(\frac{\epsilon}{2(1-\beta)}, \delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t). \tag{7.3.2}$$

Combining Theorem 13 and Theorem 16 with Eq. (7.3.2), the number of labeled target samples to obtain a similar bound in Theorem 13 is at

least

$$\max\left\{m_l^{R_t}(\frac{\epsilon}{2\beta},\delta,\mathcal{H},\mathcal{F}_s,\mathcal{F}_t), m_l^{R_t}(\frac{\epsilon}{2(1-\beta)},\delta,\mathcal{H},\mathcal{F}_s,\mathcal{F}_t)\right\},$$

which is greater than or equal to $m_l^{R_t}(\epsilon,\delta,\mathcal{H},\mathcal{F}_s,\mathcal{F}_t)$.

Remark 10. *According to Theorem 13, in homogeneous situation, for any hypothesis $\mathcal{H}$, when $\beta \to 0$, the need for labeled target samples is close to 0. However, in heterogeneous situation, there exist hypothesis $\mathcal{H}$ and transformation spaces $\mathcal{F}_s, \mathcal{F}_t$ such that for any weight $\beta \in (0,1)$, the number of labeled target samples is at least $m_l^{R_t}(\epsilon,\delta,\mathcal{H},\mathcal{F}_s,\mathcal{F}_t)$.*

The above discussion indicates that there exist hypothesis and transformation spaces $\mathcal{H}, \mathcal{F}_s, \mathcal{F}_t$ such that the weight β does not play an important role in reducing the need for labeled target samples in SsHeDA.

7.3.4 Theoretical Analysis

We start this section from a basic theorem, which contains our main idea about SsHeDA theory.

Theorem 17. *Let $\mathcal{H}$ be the hypothesis space, $\mathcal{F}_s, \mathcal{F}_t$ be the source and target transformation spaces, and $\mathrm{err}(\cdot)$ be the transfer error rate satisfying Assumption 4. If random labeled source samples with size n_s are drawn from $P_{X_sY_s}$, i.i.d., random labeled target samples with size n_l are drawn from $P_{X_lY_l}$, i.i.d., and unlabeled target samples with size n_u are drawn from P_{X_t}, i.i.d., for $0 < \delta, \gamma_1 < 1$, $0 \le \alpha < 1$ and $\epsilon > 0$, if*

$$\begin{aligned}
&n_s > m_s^{\mathrm{err}}(\epsilon,\gamma_1\delta,\mathcal{H},\mathcal{F}_s,\mathcal{F}_t),\\
&n_u > m_u^{\mathrm{err}}(\epsilon,\gamma_1\delta,\mathcal{H},\mathcal{F}_s,\mathcal{F}_t),\\
&n_l > m_l^{R_t}(\epsilon,(1-\gamma_1)\delta,\mathcal{H},\mathcal{F}_s,\mathcal{F}_t(\alpha+\epsilon)).
\end{aligned}$$

Then, with a probability of at least $1-\delta$, for any $\boldsymbol{h} \in \mathcal{H}$, $\boldsymbol{T}_s \in \mathcal{F}_s$ and $\boldsymbol{T}_t \in \mathcal{F}_t$ with $\min_{\boldsymbol{h}^* \in \mathcal{H}, \boldsymbol{T}_s^* \in \mathcal{F}_s} \widehat{\mathrm{err}}(\boldsymbol{h}^*, \boldsymbol{T}_s^*, \boldsymbol{T}_t) < \alpha$,

$$|R_t(\boldsymbol{h}\circ\boldsymbol{T}_t) - \widehat{R}_t(\boldsymbol{h}\circ\boldsymbol{T}_t)| < \epsilon.$$

Proof of Theorem 17. **Step 1**. For any $\epsilon > 0$ and $0 < \delta_1 < 1$, if

$$n_s > m_s^{\mathrm{err}}(\epsilon, \delta_1, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t),$$
$$n_u > m_u^{\mathrm{err}}(\epsilon, \delta_1, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t),$$

then with the probability of at least $1 - \delta_1$, we have

$$\mathrm{err}(\boldsymbol{h}, \boldsymbol{T}_s, \boldsymbol{T}_t) - \widehat{\mathrm{err}}(\boldsymbol{h}, \boldsymbol{T}_s, \boldsymbol{T}_t) < \epsilon,$$

for any $\boldsymbol{h} \in \mathcal{H}$, $\boldsymbol{T}_s \in \mathcal{F}_s$ and $\boldsymbol{T}_t \in \mathcal{F}_t$.

Note that the constrict $\min_{\boldsymbol{h}\in\mathcal{H},\boldsymbol{T}_s\in\mathcal{F}_s} \widehat{\mathrm{err}}(\boldsymbol{h}, \boldsymbol{T}_s, \boldsymbol{T}_t) \leq \alpha$, which implies that there exist $\widetilde{\boldsymbol{h}}, \widetilde{\boldsymbol{T}}_s$ such that $\mathrm{err}(\widetilde{\boldsymbol{h}}, \widetilde{\boldsymbol{T}}_s, \boldsymbol{T}_t) \leq \alpha + \epsilon$, then under the constrict $\min_{\boldsymbol{h}\in\mathcal{H},\boldsymbol{T}_s\in\mathcal{F}_s} \widehat{\mathrm{err}}(\boldsymbol{h}, \boldsymbol{T}_s, \boldsymbol{T}_t) \leq \alpha$, we have

$$\boldsymbol{T}_t \in \mathcal{F}_t(\alpha + \epsilon).$$

Step 2. For any $\epsilon > 0$, $0 < \delta_2 < 1$, if

$$n_l > m_l^{R_t}(\epsilon, \delta_2, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t(\alpha + \epsilon)),$$

then with a probability of at least $1 - \delta_2 > 0$, for any $\boldsymbol{h} \in \mathcal{H}$ and $\boldsymbol{T}_t \in \mathcal{F}_t(\alpha + \epsilon)$, we have

$$|R_t(\boldsymbol{h} \circ \boldsymbol{T}_t) - \widehat{R}_t(\boldsymbol{h} \circ \boldsymbol{T}_t)| < \epsilon. \tag{7.3.3}$$

Step 3. Combining Step 1 and Step 2, we have: If

$$n_s > m_s^{\mathrm{err}}(\epsilon, \delta_1, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t),$$
$$n_u > m_u^{\mathrm{err}}(\epsilon, \delta_1, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t),$$
$$n_l > m_l^{R_t}(\epsilon, \delta_2, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t(\alpha + \epsilon)),$$

then with the probability of at least $1-\delta_1-\delta_2$,

$$|R_t(\boldsymbol{h}\circ\boldsymbol{T}_t)-\widehat{R}_t(\boldsymbol{h}\circ\boldsymbol{T}_t)|<\epsilon,$$

for any $\boldsymbol{h}\in\mathcal{H}$, $\boldsymbol{T}_s\in\mathcal{F}_s$ and $\boldsymbol{T}_t\in\mathcal{F}_t$ with $\min_{h\in\mathcal{H},\boldsymbol{T}_s\in\mathcal{F}_s}\widehat{\mathrm{err}}(\boldsymbol{h},\boldsymbol{T}_s,\boldsymbol{T}_t)\leq\alpha$. Let $\delta_1=\gamma_1\delta$ and $\delta_1=(1-\gamma_1)\delta$. We have finished the proof. □

Observing the above theorem, when γ_1 is close to 0, then the number of labeled target samples in Theorem 17 is less than that for estimating $R_t(\boldsymbol{h}\circ\boldsymbol{T}_t)$ directly. The crucial reason is because the space $\mathcal{F}_t$ is replaced by a smaller space $\mathcal{F}_t(\alpha+\epsilon)$.

To further reduce the number of labeled target samples, we can replace condition $\min_{\boldsymbol{h}^*\in\mathcal{H},\boldsymbol{T}_s^*\in\mathcal{F}_s}\widehat{\mathrm{err}}(\boldsymbol{h}^*,\boldsymbol{T}_s^*,\boldsymbol{T}_t)<\alpha$ by

$$\min_{\boldsymbol{T}_s^*\in\mathcal{F}_s}\widehat{\mathrm{err}}(\boldsymbol{h},\boldsymbol{T}_s^*,\boldsymbol{T}_t)<\alpha,$$

then the number of labeled target samples can be reduced to

$$m_l^{R_t}(\epsilon,(1-\gamma_1)\delta,\mathcal{H}(\alpha+\epsilon),\mathcal{F}_s,\mathcal{F}_t(\alpha+\epsilon)).$$

Though Theorem 17 provides an explanation of the uniform sample complexity for the labeled target samples, we still cannot explain the representative algorithms by constructing different transfer error rates. This is because the transfer error rate is not related to any information about labeled target samples, which can be used to help align the heterogeneous spaces. Hence, we add an additional constraint called the heterogeneous space alignment $d(\boldsymbol{h},\boldsymbol{T}_s,\boldsymbol{T}_t)$, which is able to be estimated by labeled source samples and labeled target samples. Motivated by previous work [23, 96, 197], we can set heterogeneous space alignment as

• class-conditional distribution alignment

$$\sum_{c=1}^{K}d^{\ell}_{\boldsymbol{h},\mathcal{H}}(P_{\boldsymbol{T}_t(X_t)|Y_t=\mathbf{y}_c},P_{\boldsymbol{T}_s(X_s)|Y_s=\mathbf{y}_c}). \tag{7.3.4}$$

• projected MMD alignment for class

$$\sum_{c=1}^{K} D_{\boldsymbol{h}}^2(P_{\boldsymbol{T}_t(X_t)|Y_t=\mathbf{y}_c}, P_{\boldsymbol{T}_s(X_s)|Y_s=\mathbf{y}_c}). \tag{7.3.5}$$

• MMD alignment for class

$$\sum_{c=1}^{K} D_{\mathcal{F}}^2(P_{\boldsymbol{T}_t(X_t)|Y_t=\mathbf{y}_c}, P_{\boldsymbol{T}_s(X_s)|Y_s=\mathbf{y}_c}). \tag{7.3.6}$$

Theorem 18. *Given the same conditions and assumption in Theorem 17, for any* $0 < \delta < 1$ *and* $\alpha, \epsilon > 0$,

$$\begin{aligned} n_s &> m_s^{\mathrm{err}}(\epsilon, \gamma_1\delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t), \\ n_u &> m_u^{\mathrm{err}}(\epsilon, \gamma_1\delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t), \\ n_l &> m_l^{R_t}(\epsilon, (1-\gamma_1)\delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t(\alpha+\epsilon)), \end{aligned}$$

where γ_1 *is from* $(0, 1)$ *and* R_t *is* $R_t(\boldsymbol{h} \circ \boldsymbol{T}_t)$. *Then with probability of at least* $1 - \delta > 0$, *for any* $\boldsymbol{h} \in \mathcal{H}$, $\boldsymbol{T}_s \in \mathcal{F}_s$ *and* $\boldsymbol{T}_t \in \mathcal{F}_t$ *with* $\min_{\boldsymbol{h}^* \in \mathcal{H}, \boldsymbol{T}_s^* \in \mathcal{F}_s} \left(\widehat{\mathrm{err}}(\boldsymbol{h}^*, \boldsymbol{T}_s^*, \boldsymbol{T}_t) + \widehat{d}(\boldsymbol{h}^*, \boldsymbol{T}_s^*, \boldsymbol{T}_t)\right) < \alpha$,

$$|R_t(\boldsymbol{h} \circ \boldsymbol{T}_t) - \widehat{R}_t(\boldsymbol{h} \circ \boldsymbol{T}_t)| < \epsilon,$$

where $\widehat{d}(\boldsymbol{h}, \boldsymbol{T}_s, \boldsymbol{T}_t)$ *is the empirical form of heterogeneous space alignment defined in Eq. (7.3.4), Eq. (7.3.5) or Eq. (7.3.6).*

Proof of Theorem 18. Let

$$\underline{\boldsymbol{h}}, \underline{\boldsymbol{T}_s} \in \operatorname*{arg\,min}_{\boldsymbol{h}^* \in \mathcal{H}, \boldsymbol{T}_s^* \in \mathcal{F}_s} \left(\widehat{\mathrm{err}}(\boldsymbol{h}^*, \boldsymbol{T}_s^*, \boldsymbol{T}_t) + \widehat{d}(\boldsymbol{h}^*, \boldsymbol{T}_s^*, \boldsymbol{T}_t)\right).$$

We have

$$\left(\widehat{\mathrm{err}}(\underline{\boldsymbol{h}}, \underline{\boldsymbol{T}_s}, \boldsymbol{T}_t) + \widehat{d}(\underline{\boldsymbol{h}}, \underline{\boldsymbol{T}_s}, \boldsymbol{T}_t)\right) \leq \alpha.$$

Note that

$$\widehat{d}(\underline{\boldsymbol{h}}, \underline{\boldsymbol{T}_s}, \boldsymbol{T}_t) \geq 0.$$

Hence,

$$\min_{\boldsymbol{h}^*\in\mathcal{H},\boldsymbol{T}_s^*\in\mathcal{F}_s}\widehat{\mathrm{err}}(\boldsymbol{h}^*,\boldsymbol{T}_s^*,\boldsymbol{T}_t)\leq\widehat{\mathrm{err}}(\underline{\boldsymbol{h}},\underline{\boldsymbol{T}_s},\boldsymbol{T}_t)\leq\alpha. \tag{7.3.7}$$

Using Theorem 17 and inequality (7.3.7), we prove the theorem. □

Reducing the space size implies that the estimation error decreases and approximate error may increase. To estimate the approximate error, Theorem 19 provides an answer.

Theorem 19. *Let ℓ be the loss satisfying the triangle inequality. If* $\mathrm{err}(\boldsymbol{h},\boldsymbol{T}_s,\boldsymbol{T}_t)=\frac{1}{B}\big(R_s(\boldsymbol{h}\circ\boldsymbol{T}_s)+d^{\ell}_{\boldsymbol{h},\mathcal{H}}(P_{\boldsymbol{T}_s(X_s)},P_{\boldsymbol{T}_t(X_t)})\big)$, $d(\boldsymbol{h},\boldsymbol{T}_s,\boldsymbol{T}_t)=\frac{1}{B}\Lambda$, *where $d^{\ell}_{\boldsymbol{h},\mathcal{H}}$ is disparity discrepancy defined in Definition 8 and* $\Lambda=\min_{\boldsymbol{h}\in\mathcal{H}}\big(R_s(\boldsymbol{h}\circ\boldsymbol{T}_s)+R_t(\boldsymbol{h}\circ\boldsymbol{T}_t)\big)$, $\frac{B}{2}$ *is the supremum of ℓ, then the approximate error*

$$\min_{\boldsymbol{h}\in\mathcal{H}(\alpha_1),\boldsymbol{T}_t\in\mathcal{F}_t(\alpha_2)}R_t(\boldsymbol{h}\circ\boldsymbol{T}_t)\leq B\alpha_{\min},$$

for $\alpha_1,\alpha_2>\alpha_{\min}$, where

$$\alpha_{\min}=\min_{\boldsymbol{h}\in\mathcal{H},\boldsymbol{T}_s\in\mathcal{F}_s,\mathbf{T}_t\in\mathcal{F}_t}\big(\mathrm{err}(\boldsymbol{h},\boldsymbol{T}_s,\boldsymbol{T}_t)+d(\boldsymbol{h},\boldsymbol{T}_s,\boldsymbol{T}_t)\big).$$

Proof of Theorem 19. **Step 1.** It is easy to check that

$$\begin{aligned}
&R_t(\boldsymbol{h}\circ\boldsymbol{T}_t)-R_s(\boldsymbol{h}\circ\boldsymbol{T}_s)\\
=&\int_{\mathcal{X}_t\times\mathcal{Y}}\ell(\boldsymbol{h}\circ\boldsymbol{T}_t(\mathbf{x}),\mathbf{y})\mathrm{d}P_{X_tY_t}(\mathbf{x},\mathbf{y})-\int_{\mathcal{X}_s\times\mathcal{Y}}\ell(\boldsymbol{h}\circ\boldsymbol{T}_s(\mathbf{x}),\mathbf{y})\mathrm{d}P_{X_sY_s}(\mathbf{x},\mathbf{y})\\
=&\int_{\mathcal{X}\times\mathcal{Y}}\ell(\boldsymbol{h}(\mathbf{x}),\mathbf{y})\mathrm{d}P_{\boldsymbol{T}_t(X_t)Y_t}(\mathbf{x},\mathbf{y})-\int_{\mathcal{X}\times\mathcal{Y}}\ell(\boldsymbol{h}(\mathbf{x}),\mathbf{y})\mathrm{d}P_{\boldsymbol{T}_s(X_s)Y_s}(\mathbf{x},\mathbf{y})\\
\leq&\int_{\mathcal{X}\times\mathcal{Y}}\ell(\boldsymbol{h}(\mathbf{x}),\boldsymbol{h}'(\mathbf{x}))\mathrm{d}P_{\boldsymbol{T}_t(X_t)Y_t}(\mathbf{x},\mathbf{y})-\int_{\mathcal{X}\times\mathcal{Y}}\ell(\boldsymbol{h}(\mathbf{x}),\boldsymbol{h}'(\mathbf{x}))\mathrm{d}P_{\boldsymbol{T}_s(X_s)Y_s}(\mathbf{x},\mathbf{y})\\
&+R_t(\boldsymbol{h}'\circ\boldsymbol{T}_t)+R_s(\boldsymbol{h}'\circ\boldsymbol{T}_s),
\end{aligned}\tag{7.3.8}$$

where $\boldsymbol{h}'$ is any scoring function in $\mathcal{H}$. In the above inequality, we have used following triangle inequality: $|\ell(\boldsymbol{h}(\mathbf{x}),\mathbf{y})-\ell(\boldsymbol{h}'(\mathbf{x}),\mathbf{y})|\leq\ell(\boldsymbol{h}(\mathbf{x}),\boldsymbol{h}'(\mathbf{x}))$.

Then, according to the definition of $P_{X_tY_t}$, we can check that

$$\int_{\mathcal{X}\times\mathcal{Y}} \ell(\boldsymbol{h}(\mathbf{x}), \boldsymbol{h}'(\mathbf{x}))\mathrm{d}P_{\boldsymbol{T}_t(X_t)Y_t}(\mathbf{x},\mathbf{y}) = \int_{\mathcal{X}} \ell(\boldsymbol{h}(\mathbf{x}), \boldsymbol{h}'(\mathbf{x}))\mathrm{d}P_{\boldsymbol{T}_t(X_t)}(\mathbf{x}). \tag{7.3.9}$$

Using Fubini's theorem, it is also easy to check that

$$\int_{\mathcal{X}\times\mathcal{Y}} \ell(\boldsymbol{h}(\mathbf{x}), \boldsymbol{h}'(\mathbf{x}))\mathrm{d}P_{\boldsymbol{T}_s(X_s)Y_s}(\mathbf{x},\mathbf{y}) = \int_{\mathcal{X}} \ell(\boldsymbol{h}(\mathbf{x}), \boldsymbol{h}'(\mathbf{x}))\mathrm{d}P_{\boldsymbol{T}_s(X_s)}(\mathbf{x}). \tag{7.3.10}$$

Based on (7.3.8), (7.3.9), (7.3.10) and the definition of the discrepancy distance, we have

$$\begin{aligned}
&R_t(\boldsymbol{h}\circ\boldsymbol{T}_t) - R_s(\boldsymbol{h}\circ\boldsymbol{T}_s)\\
\leq& R_t(\boldsymbol{h}'\circ\boldsymbol{T}_t) + R_s(\boldsymbol{h}'\circ\boldsymbol{T}_s)\\
+&\left|\int_{\mathcal{X}\times\mathcal{Y}} \ell(\boldsymbol{h}(\mathbf{x}), \boldsymbol{h}'(\mathbf{x}))\mathrm{d}P_{\boldsymbol{T}_t(X_t)}(\mathbf{x}) - \int_{\mathcal{X}\times\mathcal{Y}} \ell(\boldsymbol{h}(\mathbf{x}), \boldsymbol{h}'(\mathbf{x}))\mathrm{d}P_{\boldsymbol{T}_s(X_s)}(\mathbf{x})\right|\\
\leq& R_t(\boldsymbol{h}'\circ\boldsymbol{T}_t) + R_s(\boldsymbol{h}'\circ\boldsymbol{T}_s) + d^{\ell}_{\boldsymbol{h},\mathcal{H}}(P_{\boldsymbol{T}_t(X_t)}, P_{\boldsymbol{T}_s(X_s)}).
\end{aligned}$$

Hence,

$$\begin{aligned}
&R_t(\boldsymbol{h}\circ\boldsymbol{T}_t) - R_s(\boldsymbol{h}\circ\boldsymbol{T}_s)\\
\leq& \min_{\overline{\boldsymbol{h}}\in\mathcal{H}} \left(R_t(\overline{\boldsymbol{h}}\circ\boldsymbol{T}_t) + R_s(\overline{\boldsymbol{h}}\circ\boldsymbol{T}_s)\right) + d^{\ell}_{\boldsymbol{h},\mathcal{H}}(P_{\boldsymbol{T}_t(X_t)}, P_{\boldsymbol{T}_s(X_t)}).
\end{aligned} \tag{7.3.11}$$

Note that

$$\begin{aligned}
\mathrm{err}(\boldsymbol{h},\boldsymbol{T}_s,\boldsymbol{T}_t) &= \frac{1}{B} d^{\ell}_{\boldsymbol{h},\mathcal{H}}(P_{\boldsymbol{T}_t(X_t)}, P_{\boldsymbol{T}_s(X_t)}),\\
d(\boldsymbol{h},\boldsymbol{T}_s,\boldsymbol{T}_t) &= \frac{1}{B}\min_{\overline{\boldsymbol{h}}\in\mathcal{H}} \left(R_t(\overline{\boldsymbol{h}}\circ\boldsymbol{T}_t) + R_s(\overline{\boldsymbol{h}}\circ\boldsymbol{T}_s)\right),
\end{aligned}$$

hence,

$$R_t(\boldsymbol{h}\circ\boldsymbol{T}_t) - R_s(\boldsymbol{h}\circ\boldsymbol{T}_s) \leq B\mathrm{err}(\boldsymbol{h},\boldsymbol{T}_s,\boldsymbol{T}_t) + Bd(\boldsymbol{h},\boldsymbol{T}_s,\boldsymbol{T}_t).$$

Step 2. Let

$$\underline{\boldsymbol{h}},\underline{\boldsymbol{T}_s},\underline{\boldsymbol{T}_t} \in \mathop{\arg\min}_{\boldsymbol{h}\in\mathcal{H},\boldsymbol{T}_s\in\mathcal{F}_s,\boldsymbol{T}_t\in\mathcal{F}_t} \mathrm{err}(\boldsymbol{h},\boldsymbol{T}_s,\boldsymbol{T}_t) + d(\boldsymbol{h},\boldsymbol{T}_s,\boldsymbol{T}_t).$$

Then

$$\min_{\boldsymbol{h}\in\mathcal{H}(\alpha_1),T_t\in\mathcal{F}_t(\alpha_2)} R_t(\boldsymbol{h}\circ\boldsymbol{T}_t) \leq \min_{\mathbf{h}\in\mathcal{H}(\alpha_{\min}),\mathbf{T}_t\in\mathcal{F}_t(\alpha_{\min})} R_t(\boldsymbol{h}\circ\boldsymbol{T}_t) \leq R_t(\underline{\boldsymbol{h}}\circ\underline{\boldsymbol{T}_t})$$
$$\leq B\big(\mathrm{err}(\underline{\boldsymbol{h}},\underline{\boldsymbol{T}_s},\underline{\boldsymbol{T}_t}) + d(\underline{\boldsymbol{h}},\underline{\boldsymbol{T}_s},\underline{\boldsymbol{T}_t})\big) = B\alpha_{\min}.$$

We have finished the proof. □

We use the combined error $\Lambda(\boldsymbol{h},\boldsymbol{T}_s,\boldsymbol{T}_t)/B$ as the heterogeneous space alignment term in above theorem. The combined error is deeply related to the conditional distribution discrepancy [13], hence, it can be used to align heterogeneous spaces. In addition, if we require

$$R_t(\boldsymbol{h}\circ\boldsymbol{T}_t) \leq C\big(\mathrm{err}(\boldsymbol{h},\boldsymbol{T}_s,\boldsymbol{T}_t) + d(\boldsymbol{h},\boldsymbol{T}_s,\boldsymbol{T}_t)\big),$$

we can also obtain a result similar to the above theorem.

7.4 Bring SsHeDA Theory into the Reality

Although we have shown that, under mild assumptions, we can use labeled source and unlabeled target samples to reduce the need for the labeled target samples, it is still unknown how to apply the proposed theory to design SsHeDA algorithms. This section shows how to design a loss function according to Theorem 18, i.e., bringing Theorem 18 into the reality. Besides, we revise slightly our theory-based optimization problem shown in Eq. (7.4.1) for achieving better performance.

7.4.1 Loss Function Motivated by Theorem 18

As discussed in Theorem 18, we should consider the optimization problem as follows:

$$\begin{aligned}&\min_{\boldsymbol{h}\in\mathcal{H},\boldsymbol{T}_t\in\mathcal{F}_t} \widehat{R}_t(\boldsymbol{h}\circ\boldsymbol{T}_t),\\ &\text{with}\ \min_{\boldsymbol{h}^*\in\mathcal{H},\boldsymbol{T}_s^*\in\mathcal{F}_s} \big(\widehat{\mathrm{err}}(\boldsymbol{h}^*,\boldsymbol{T}_s^*,\boldsymbol{T}_t) + \widehat{d}(\boldsymbol{h}^*,\boldsymbol{T}_s^*,\boldsymbol{T}_t)\big) \leq \alpha.\end{aligned} \tag{7.4.1}$$

However, such constraint optimization problem cannot be easily solved. Following [198], we replace the constraint in problem (7.4.1) as a penalty, thus, we revise the loss function of problem (7.4.1) as

$$\widehat{R}_t(\boldsymbol{h} \circ \boldsymbol{T}_t) + \lambda \min_{\boldsymbol{h}^* \in \mathcal{H}, \boldsymbol{T}_s \in \mathcal{F}_s} \big(\widehat{\mathrm{err}}(\boldsymbol{h}^*, \boldsymbol{T}_s, \boldsymbol{T}_t) + \widehat{d}(\boldsymbol{h}^*, \boldsymbol{T}_s, \boldsymbol{T}_t)\big),$$

where λ ($\lambda > 0$) is a free parameter. Hence, the revised problem (7.4.1) is

$$\min_{\boldsymbol{h}, \boldsymbol{h}^* \in \mathcal{H}, \boldsymbol{T}_s \in \mathcal{F}_s, \boldsymbol{T}_t \in \mathcal{F}_t} \widehat{R}_t(\boldsymbol{h} \circ \boldsymbol{T}_t) + \lambda \widehat{\mathrm{err}}(\boldsymbol{h}^*, \boldsymbol{T}_s, \boldsymbol{T}_t) + \lambda \widehat{d}(\boldsymbol{h}^*, \boldsymbol{T}_s, \boldsymbol{T}_t). \quad (7.4.2)$$

It is important to construct transfer error rate $\mathrm{err}(\boldsymbol{h}, \boldsymbol{T}_s, \boldsymbol{T}_t)$ and heterogeneous space alignment $d(\boldsymbol{h}, \boldsymbol{T}_s, \boldsymbol{T}_t)$. Motivated by [96, 199], in our kernel-based algorithm

$$\begin{aligned} \mathrm{err}(\boldsymbol{h}, \boldsymbol{T}_s, \boldsymbol{T}_t) &= R_s(\boldsymbol{h} \circ \boldsymbol{T}_s) + \rho D^2_{\boldsymbol{h}}(P_{\boldsymbol{T}_t(X_t)}, P_{\boldsymbol{T}_s(X_s)}), \\ d(\boldsymbol{h}, \boldsymbol{T}_s, \boldsymbol{T}_t) &= \rho \sum_{c=1}^{K} D^2_{\boldsymbol{h}}(P_{\boldsymbol{T}_t(X_t)|Y_t=\mathbf{y}_c}, P_{\boldsymbol{T}_s(X_s)|Y_s=\mathbf{y}_c}), \end{aligned} \quad (7.4.3)$$

where ρ ($\rho > 0$) is free parameter.

7.4.2 Target Distribution Alignment

Note that the number of labeled target samples is far less than the number of unlabeled target samples. Hence, the empirical distribution discrepancy between labeled target and unlabeled target samples may be large. To achieve better performance, the target distribution alignment $\widehat{d}_t(\boldsymbol{h}, \boldsymbol{T}_t, \boldsymbol{T}_t)$ is considered:

In our kernel-based algorithm

$$D^2_{\boldsymbol{h}}(\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_l)}, \widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_u)}) + \sum_{c=1}^{K} D^2_{\boldsymbol{h}}(\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_l^c)}, \widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_u^{\bar{c}})}). \quad (7.4.4)$$

7.4.3 Overall Loss Function

Inspired by the above discussions, to solve the SsHeDA problem well, we need to take care of the following optimization problem

$$\min_{\boldsymbol{h},\boldsymbol{h}^*\in\mathcal{H},\boldsymbol{T}_s\in\mathcal{F}_s,\boldsymbol{T}_t\in\mathcal{F}_t} \mathcal{L}(\boldsymbol{h},\boldsymbol{h}^*,\boldsymbol{T}_s,\boldsymbol{T}_t), \tag{7.4.5}$$

where

$$\begin{aligned}\mathcal{L}(\boldsymbol{h},\boldsymbol{h}^*,\boldsymbol{T}_s,\boldsymbol{T}_t) &= \widehat{R}_t(\boldsymbol{h}\circ\boldsymbol{T}_t)+\lambda\widehat{R}_t(\boldsymbol{h}^*\circ\boldsymbol{T}_t)\\ +\lambda\big(\widehat{d}(\boldsymbol{h}^*,\boldsymbol{T}_s,\boldsymbol{T}_t)&+\widehat{\mathrm{err}}(\boldsymbol{h},\boldsymbol{T}_s,\boldsymbol{T}_t)\big)+\rho\widehat{d}_t(\boldsymbol{h},\boldsymbol{T}_t,\boldsymbol{T}_t),\end{aligned} \tag{7.4.6}$$

here $\widehat{\mathrm{err}}(\boldsymbol{h},\boldsymbol{T}_s,\boldsymbol{T}_t)$, $\widehat{d}(\boldsymbol{h}^*,\boldsymbol{T}_s,\boldsymbol{T}_t)$ are the empirical forms of Eq. (7.4.3), and $\widehat{d}_t(\boldsymbol{h},\boldsymbol{T}_t,\boldsymbol{T}_t)$ is defined in Eq. (7.4.4). Note that we have added $\widehat{R}_t(\boldsymbol{h}^*\circ\boldsymbol{T}_t)$ in Eq. (7.4.6), because we need to guarantee that the labeled target samples can be classified accurately.

7.5 Kernel-based Algorithm for SsHeDA

This section presents kernel heterogeneous domain alignment (KHDA) algorithm, where the spaces $\mathcal{F}_s,\mathcal{F}_t$ and $\mathcal{H}$ in problem (7.4.5) are defined as follows:

$$\begin{aligned}\mathcal{F}_s &= \{[T_1,...,T_c,...,T_d] \mid T_c\in\mathcal{H}_{k_s}\},\\ \mathcal{F}_t &= \{[T_1,...,T_c,...,T_d] \mid T_c\in\mathcal{H}_{k_t}\},\\ \mathcal{H} &= \{[h_1,...,h_c,...,h_K] \mid h_c(\mathbf{x})=\mathbf{x}\boldsymbol{a}^T,\ \boldsymbol{a},\mathbf{x}\in\mathbb{R}^{1\times d}\},\end{aligned}$$

where d is the dimension of the latent space $\mathcal{X}$, $\mathcal{H}_{k_s}$ is the RKHS space with kernel $k_s(\cdot,\cdot)$ defined in the space $\mathcal{X}_s\times\mathcal{X}_s$ and $\mathcal{H}_{k_t}$ is the RKHS space with kernel $k_t(\cdot,\cdot)$ defined in the space $\mathcal{X}_t\times\mathcal{X}_t$. The function h_c is a linear function, thus, $h_c\circ\boldsymbol{T}_s\in\mathcal{H}_{k_s}$ and $h_c\circ\boldsymbol{T}_t\in\mathcal{H}_{k_t}$.

7.5.1 Loss Function in KHDA

As introduced in Eq. (7.4.3), the transfer error rate is $R_s(\boldsymbol{h} \circ \boldsymbol{T}_s) + \rho D^2_{\boldsymbol{h}}(P_{\boldsymbol{T}_t(X_t)}, P_{\boldsymbol{T}_s(X_s)})$, then the empirical form of the transfer error rate can be written as

$$\widehat{R}_s(\boldsymbol{h} \circ \boldsymbol{T}_s) + \rho D^2_{\boldsymbol{h}}(\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_t)}, \widehat{P}_{\boldsymbol{T}_s(\mathbf{X}_s)}).$$

According to Eq. (7.4.3), the heterogeneous space alignment $d(\boldsymbol{h}, \boldsymbol{T}_s, \boldsymbol{T}_t)$ is set as projected MMD alignment. The empirical projected MMD alignment is

$$\sum_{c=1}^{K} D^2_{\boldsymbol{h}^*}(\widehat{P}_{\boldsymbol{T}_s(\mathbf{X}_s^c)}, \widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_l^c)}).$$

Motivated by [65, 96], the pseudo labels are used to further improve the classification performance, hence, we replace above equation by

$$\sum_{c=1}^{K} D^2_{\boldsymbol{h}^*}(\widehat{P}_{\boldsymbol{T}_s(\mathbf{X}_s^c)}, \widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_t^{\bar{c}})}). \tag{7.5.1}$$

where $\mathbf{X}_t^{\bar{c}} = [\mathbf{X}_l^c, \mathbf{X}_u^{\bar{c}}]$, $\mathbf{X}_u^{\bar{c}}$ is unlabeled data matrix with pseudo label c.

Then, to preserve domains' special structures, such as manifold structure and clustering structure, manifold regularization [121] is considered in KHDA. Many kernel-based domain adaptation algorithms [96, 199] have studied the manifold regularization and shown that it does help to improve the transfer performance. One can write the manifold regularizations $\widehat{M}_1(\boldsymbol{h}^*, \boldsymbol{T}_s, \boldsymbol{T}_t), \widehat{M}_2(\boldsymbol{h}, \boldsymbol{T}_t, \boldsymbol{T}_t)$ as follows:

$$\sum_{\mathbf{x},\mathbf{x}' \in \mathbf{X}_s \text{ or } \mathbf{X}_t} \left\| \boldsymbol{h}^* \circ \boldsymbol{T}(\mathbf{x}) - \boldsymbol{h}^* \circ \boldsymbol{T}(\mathbf{x}') \right\|_2^2 \mathbf{W}^*(\mathbf{x}, \mathbf{x}'),$$
$$\sum_{\mathbf{x},\mathbf{x}' \in \mathbf{X}_t} \left\| \boldsymbol{h} \circ \boldsymbol{T}_t(\mathbf{x}) - \boldsymbol{h} \circ \boldsymbol{T}_t(\mathbf{x}') \right\|_2^2 \mathbf{W}(\mathbf{x}, \mathbf{x}'),$$

where $\boldsymbol{T}(\mathbf{x}) = \boldsymbol{T}_s(\mathbf{x})$ if $\mathbf{x} \in \mathbf{X}_s$, otherwise, $\boldsymbol{T}(\mathbf{x}) = \boldsymbol{T}_t(\mathbf{x})$; $\mathbf{W}^*(\mathbf{x}, \mathbf{x}')$ and $\mathbf{W}(\mathbf{x}, \mathbf{x}')$ are the pair-wise affinity functions and estimate the similarity of $\mathbf{x}, \mathbf{x}'$.

Summarizing the above discussion, the loss function (7.4.6) can be

rewritten as follows:

$$\begin{aligned}
&\widehat{R}_t(\boldsymbol{h}\circ\boldsymbol{T}_t)+\sigma\|\boldsymbol{h}\circ\boldsymbol{T}_s\|_s^2+\sigma\|\boldsymbol{h}\circ\boldsymbol{T}_t\|_t^2\\
&+\rho\big(\widehat{d}_t(\boldsymbol{h},\boldsymbol{T}_t,\boldsymbol{T}_t)+\widehat{M}_2(\boldsymbol{h},\boldsymbol{T}_t,\boldsymbol{T}_t)\big)\\
&+\lambda\widehat{R}_s(\boldsymbol{h}^*\circ\boldsymbol{T}_s)+\lambda\widehat{R}_t(\boldsymbol{h}^*\circ\boldsymbol{T}_t)+\lambda\sigma\|\boldsymbol{h}^*\circ\boldsymbol{T}_s\|_s^2\\
&+\lambda\rho\big(\widehat{d}_s(\boldsymbol{h}^*,\boldsymbol{T}_s,\boldsymbol{T}_t)+\widehat{M}_1(\boldsymbol{h}^*,\boldsymbol{T}_s,\boldsymbol{T}_t)\big)+\lambda\sigma\|\boldsymbol{h}^*\circ\boldsymbol{T}_t\|_t^2,\quad\text{where}
\end{aligned}\tag{7.5.2}$$

$$\begin{aligned}
\widehat{d}_s(\boldsymbol{h}^*,\boldsymbol{T}_s,\boldsymbol{T}_t)&=D^2_{\boldsymbol{h}^*}(\widehat{P}_{\boldsymbol{T}_s(\mathbf{X}_s)},\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_t)})+\sum_{c=1}^{K}D^2_{\boldsymbol{h}^*}(\widehat{P}_{\boldsymbol{T}_s(\mathbf{X}_s^c)},\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_t^{\bar{c}})}),\\
\widehat{d}_t(\boldsymbol{h},\boldsymbol{T}_t,\boldsymbol{T}_t)&=D^2_{\boldsymbol{h}}(\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_l)},\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_u)})+\sum_{c=1}^{K}D^2_{\boldsymbol{h}}(\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_l^c)},\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_u^{\bar{c}})}),
\end{aligned}\tag{7.5.3}$$

$\|\cdot\|_s^2$, $\|\cdot\|_t^2$ are the squared norms in RKHS with kernel k_s and k_t respectively; $\|\boldsymbol{h}\circ\boldsymbol{T}_s\|_s^2+\|\boldsymbol{h}\circ\boldsymbol{T}_t\|_t^2$ and $\|\boldsymbol{h}^*\circ\boldsymbol{T}_s\|_s^2+\|\boldsymbol{h}^*\circ\boldsymbol{T}_t\|_t^2$ are used to avoid over-fitting; and σ is the free parameters ($\sigma\geq 0$). In addition, we set the loss ℓ as the squared loss $\ell(\mathbf{y},\mathbf{y}')=\|\mathbf{y}-\mathbf{y}'\|_2^2$ in KHDA.

7.5.2 Reformulation of the KHDA Loss Function

This section shows how to reformulate Eq. (7.5.2). Following the representer theorem [200], $\boldsymbol{T}_s$ and $\boldsymbol{T}_t$ can be written as

$$\begin{aligned}
\boldsymbol{T}_s(\mathbf{x})&=\sum_{i=1}^{n_s}\boldsymbol{\alpha}^i k_s(\mathbf{x},\mathbf{x}^i),\quad \forall\mathbf{x}^i\in\mathbf{X}_s,\ \mathbf{x}\in\mathcal{X}_s,\\
\boldsymbol{T}_t(\mathbf{x})&=\sum_{i=1}^{n_t}\boldsymbol{\beta}^i k_t(\mathbf{x},\mathbf{x}^i),\quad \forall\mathbf{x}^i\in\mathbf{X}_t,\ \mathbf{x}\in\mathcal{X}_t,
\end{aligned}$$

where $\boldsymbol{\alpha}^i,\boldsymbol{\beta}^i\in\mathbb{R}^{1\times d}$ are the parameters. We define matrices $\boldsymbol{\alpha}\in\mathbb{R}^{n_s\times d}$, $\boldsymbol{\beta}\in\mathbb{R}^{n_t\times d}$ and $\boldsymbol{\Theta}\in\mathbb{R}^{(n_s+n_t)\times d}$ as

$$\boldsymbol{\alpha}=\begin{bmatrix}\boldsymbol{\alpha}^1\\ \cdots\\ \boldsymbol{\alpha}^i\\ \cdots\\ \boldsymbol{\alpha}^{n_s}\end{bmatrix},\quad \boldsymbol{\beta}=\begin{bmatrix}\boldsymbol{\beta}^1\\ \cdots\\ \boldsymbol{\beta}^i\\ \cdots\\ \boldsymbol{\beta}^{n_t}\end{bmatrix},\quad \boldsymbol{\Theta}=\begin{bmatrix}\boldsymbol{\alpha}\\ \boldsymbol{\beta}\end{bmatrix}.$$

We also define the kernel matrix

$$\mathbf{K} = \begin{bmatrix} \mathbf{K}_{ss} & \mathbf{O} \\ \mathbf{O} & \mathbf{K}_{tt} \end{bmatrix} \in \mathbb{R}^{(n_s+n_t)\times(n_s+n_t)}, \tag{7.5.4}$$

where $\mathbf{K}_{ss} = [k_s(\mathbf{x}_i, \mathbf{x}_j)] \in \mathbb{R}^{n_s \times n_s}$, $\mathbf{K}_{tt} = [k_t(\mathbf{x}_i, \mathbf{x}_j)] \in \mathbb{R}^{n_t \times n_t}$ are source and target kernel matrices, respectively.

Additionally, the hypothesis space $\mathcal{H}$ is the linear space, thus, we can write $\boldsymbol{h}$ and $\boldsymbol{h}^*$ as follows: $\boldsymbol{h}(\mathbf{x}) = \mathbf{x}\boldsymbol{\Gamma}, \quad \boldsymbol{h}^*(\mathbf{x}) = \mathbf{x}\boldsymbol{\Gamma}^*, \qquad \forall \mathbf{x} \in \mathcal{X}$, where $\boldsymbol{\Gamma}, \boldsymbol{\Gamma}^* \in \mathbb{R}^{d\times K}$ are the parameters. With this form of $\boldsymbol{T}_s, \boldsymbol{T}_t, \boldsymbol{h}, \boldsymbol{h}^*$, we explain the computation of terms in (7.5.2).

7.5.2.1 Empirical Risks

We use a matrix to rewrite the following equation:

$$\begin{aligned} &\lambda \widehat{R}_s(\boldsymbol{h}^* \circ \boldsymbol{T}_s) + \widehat{R}_t(\boldsymbol{h}^* \circ \boldsymbol{T}_t) + \sigma \|\boldsymbol{h}^* \circ \boldsymbol{T}_s\|_s^2 + \sigma \|\boldsymbol{h}^* \circ \boldsymbol{T}_t\|_t^2 \\ &+\widehat{R}_t(\boldsymbol{h} \circ \boldsymbol{T}_t) + \sigma \|\boldsymbol{h} \circ \boldsymbol{T}_s\|_s^2 + \sigma \|\boldsymbol{h} \circ \boldsymbol{T}_t\|_t^2. \end{aligned} \tag{7.5.5}$$

Let the label matrix be $\mathbf{Y} \in \mathbb{R}^{(n_s+n_t)\times K}$:

$$\mathbf{Y}_{ic} = \begin{cases} 1, & \mathbf{x}_i \in \mathbf{X}_s^c \text{ or } \mathbf{X}_l^c, \\ 0, & \text{otherwise.} \end{cases} \tag{7.5.6}$$

Then, the Eq. (7.5.5) can be written as

$$\begin{aligned} &\|\mathbf{A}(\mathbf{Y} - \mathbf{K}\boldsymbol{\Theta}\boldsymbol{\Gamma})\|_F^2 + \lambda \|\mathbf{A}^*(\mathbf{Y} - \mathbf{K}\boldsymbol{\Theta}\boldsymbol{\Gamma}^*)\|_F^2 \\ &+\sigma \mathrm{tr}((\boldsymbol{\Theta}\boldsymbol{\Gamma})^T \mathbf{K} \boldsymbol{\Theta}\boldsymbol{\Gamma}) + \sigma\lambda \mathrm{tr}((\boldsymbol{\Theta}\boldsymbol{\Gamma}^*)^T \mathbf{K} \boldsymbol{\Theta}\boldsymbol{\Gamma}^*) \\ &=\mathrm{tr}\big((\boldsymbol{\Theta}\boldsymbol{\Gamma})^T (\mathbf{K}\mathbf{A}^2\mathbf{K} + \sigma\mathbf{K})\boldsymbol{\Theta}\boldsymbol{\Gamma}\big) - 2\mathrm{tr}\big(\mathbf{Y}^T \mathbf{A}^2 \mathbf{K}\boldsymbol{\Theta}\boldsymbol{\Gamma}\big) \\ &+\lambda \mathrm{tr}\big((\boldsymbol{\Theta}\boldsymbol{\Gamma}^*)^T (\mathbf{K}\mathbf{A}^{*2}\mathbf{K} + \sigma\mathbf{K})\boldsymbol{\Theta}\boldsymbol{\Gamma}^*\big) - 2\lambda \mathrm{tr}\big(\mathbf{Y}^T \mathbf{A}^{*2} \mathbf{K}\boldsymbol{\Theta}\boldsymbol{\Gamma}^*\big) \\ &+\text{Constant} \end{aligned} \tag{7.5.7}$$

where $\mathbf{A}$ is a $(n_s + n_t) \times (n_s + n_t)$ diagonal matrix with $\mathbf{A}_{ii} = \sqrt{\frac{1}{n_l}}$ if $\mathbf{x}_i \in \mathbf{X}_l$, otherwise, $\mathbf{A}_{ii} = 0$; $\mathbf{A}^*$ is a $(n_s + n_t) \times (n_s + n_t)$ diagonal matrix with $\mathbf{A}^*_{ii} = \sqrt{\frac{1}{n_s}}$ if $\mathbf{x}_i \in \mathbf{X}_s$, $\mathbf{A}^*_{ii} = \sqrt{\frac{1}{n_l}}$ if $\mathbf{x}_i \in \mathbf{X}_l$, otherwise, $\mathbf{A}^*_{ii} = 0$,

and $\|\cdot\|_F$ is the Frobenius norm.

7.5.2.2 Distribution Matching

Using the representer theorem [200] and kernel trick [199], we reformulate Eq. (7.5.3) as follows:

$$\begin{aligned}\widehat{d}_s(\boldsymbol{h}^*, \boldsymbol{T}_s, \boldsymbol{T}_t) &= \mathrm{tr}\big((\boldsymbol{\Theta}\boldsymbol{\Gamma}^*)^T \mathbf{K}\mathbf{M}^*\mathbf{K}\boldsymbol{\Theta}\boldsymbol{\Gamma}^*\big),\\ \widehat{d}_t(\boldsymbol{h}, \boldsymbol{T}_t, \boldsymbol{T}_t) &= \mathrm{tr}\big((\boldsymbol{\Theta}\boldsymbol{\Gamma})^T \mathbf{K}\mathbf{M}\mathbf{K}\boldsymbol{\Theta}\boldsymbol{\Gamma}\big),\end{aligned} \tag{7.5.8}$$

where $\mathbf{M} = \mathbf{M}_0 + \sum_{c=1}^{K} \mathbf{M}_c$, $\mathbf{M}^* = \mathbf{M}_0^* + \sum_{c=1}^{K} \mathbf{M}_c^*$ are the MMD matrices.

The (i,j)-th elements of matrices $\mathbf{M}_0$, $\mathbf{M}_c$, $\mathbf{M}_0^*$, $\mathbf{M}_c^*$ are

$$(\mathbf{M}_0)_{ij} = \begin{cases} 0, & \mathbf{x}_i \text{ or } \mathbf{x}_j \in \mathbf{X}_s,\\ \dfrac{1}{(n_l)^2}, & \mathbf{x}_i, \mathbf{x}_j \in \mathbf{X}_l,\\ \dfrac{1}{(n_u)^2}, & \mathbf{x}_i, \mathbf{x}_j \in \mathbf{X}_u,\\ -\dfrac{1}{n_l n_u}, & \text{otherwise;}\end{cases}$$

$$(\mathbf{M}_c)_{ij} = \begin{cases} \dfrac{1}{(n_l^c)^2}, & \mathbf{x}_i, \mathbf{x}_j \in \mathbf{X}_l^c,\\ \dfrac{1}{(n_u^{\bar{c}})^2}, & \mathbf{x}_i, \mathbf{x}_j \in \mathbf{X}_u^{\bar{c}},\\ -\dfrac{1}{n_l^c n_u^{\bar{c}}}, & \mathbf{x}_i \in \mathbf{X}_l^c, \mathbf{x}_j \in \mathbf{X}_u^{\bar{c}},\\ -\dfrac{1}{n_l^c n_u^{\bar{c}}}, & \mathbf{x}_j \in \mathbf{X}_l^c, \mathbf{x}_j \in \mathbf{X}_u^{\bar{c}},\\ 0, & \text{otherwise;}\end{cases}$$

$$(\mathbf{M}_0^*)_{ij} = \begin{cases} \dfrac{1}{(n_s)^2}, & \mathbf{x}_i, \mathbf{x}_j \in \mathbf{X}_s,\\ \dfrac{1}{(n_t)^2}, & \mathbf{x}_i, \mathbf{x}_j \in \mathbf{X}_t,\\ -\dfrac{1}{n_s n_t}, & \text{otherwise;}\end{cases}$$

$$(\mathbf{M}_c^*)_{ij} = \begin{cases} \dfrac{1}{(n_s^c)^2}, & \mathbf{x}_i, \mathbf{x}_j \in \mathbf{X}_s^c, \\ \dfrac{1}{(n_l^c + n_u^{\bar{c}})^2}, & \mathbf{x}_i, \mathbf{x}_j \in \mathbf{X}_t^{\bar{c}}, \\ -\dfrac{1}{(n_l^c + n_u^{\bar{c}})n_s^c}, & \mathbf{x}_i \in \mathbf{X}_s^c, \mathbf{x}_j \in \mathbf{X}_t^{\bar{c}}, \\ -\dfrac{1}{(n_l^c + n_u^{\bar{c}})n_s^c}, & \mathbf{x}_j \in \mathbf{X}_s^c, \mathbf{x}_i \in \mathbf{X}_t^{\bar{c}}, \\ 0, & \text{otherwise;} \end{cases}$$

where $n_u^{\bar{c}}$ is the number of unlabeled target samples with pseudo label c.

7.5.2.3 Manifold Regularization

The pair-wise source affinity matrix $\mathbf{W}_s$ is denoted as

$$(\mathbf{W}_s)_{ij} = \begin{cases} \text{sim}(\mathbf{x}_i, \mathbf{x}_j), & \mathbf{x}_i \in \mathcal{N}_p(\mathbf{x}_j) \text{ or } \mathbf{x}_j \in \mathcal{N}_p(\mathbf{x}_i) \\ 0, & \text{otherwise;} \end{cases}$$

here $\mathbf{x}_i, \mathbf{x}_j$ are from $\mathbf{X}_s$, $\text{sim}(\mathbf{x}_i, \mathbf{x}_j)$ is the similarity function such as cosine similarity, $\mathcal{N}_p(\mathbf{x}_i)$ denotes the set of p-nearest neighbors to point $\mathbf{x}_i$ and p is a free parameter.

The pair-wise target affinity matrix $\mathbf{W}_t$ is denoted as

$$(\mathbf{W}_t)_{ij} = \begin{cases} \text{sim}(\mathbf{x}_i, \mathbf{x}_j), & \mathbf{x}_i \in \mathcal{N}_p(\mathbf{x}_j) \text{ or } \mathbf{x}_j \in \mathcal{N}_p(\mathbf{x}_i) \\ 0, & \text{otherwise;} \end{cases}$$

where $\mathbf{x}_i, \mathbf{x}_j$ are from $\mathbf{X}_t$.

Using $\mathbf{W}_s$ and $\mathbf{W}_t$, we construct two matrices

$$\mathbf{W} = \begin{bmatrix} \mathbf{O} & \mathbf{O} \\ \mathbf{O} & \mathbf{W}_t \end{bmatrix} \in \mathbb{R}^{(n_s+n_t)\times(n_s+n_t)},$$

$$\mathbf{W}^* = \begin{bmatrix} \mathbf{W}_s & \mathbf{O} \\ \mathbf{O} & \mathbf{W}_t \end{bmatrix} \in \mathbb{R}^{(n_s+n_t)\times(n_s+n_t)}.$$

Using the representer theorem and kernel trick, we can formulate

$\widehat{M}_1(\boldsymbol{h}^*, \boldsymbol{T}_s, \boldsymbol{T}_t)$ and $\widehat{M}_2(\boldsymbol{h}, \boldsymbol{T}_t, \boldsymbol{T}_t)$ as

$$\mathrm{tr}\big((\boldsymbol{\Theta\Gamma}^*)^T\mathbf{K}\mathbf{L}^*\mathbf{K}\boldsymbol{\Theta\Gamma}^*\big), \quad \mathrm{tr}\big((\boldsymbol{\Theta\Gamma})^T\mathbf{K}\mathbf{L}\mathbf{K}\boldsymbol{\Theta\Gamma}\big), \tag{7.5.9}$$

where $\mathbf{L}^*$ and $\mathbf{L}$ are the Laplacian matrices, which can be written as $\mathbf{D}^* - \mathbf{W}^*$ and $\mathbf{D} - \mathbf{W}$. $\mathbf{D}^*$, $\mathbf{D}$ are diagonal matrices: $\mathbf{D}^*_{ii} = \sum_{j=1}^{n_s+n_t} \mathbf{W}^*_{ij}$, $\mathbf{D}_{ii} = \sum_{j=1}^{n_s+n_t} \mathbf{W}_{ij}$.

7.5.2.4 Overall Reformulation

Finally, combining (7.5.2) with (7.5.7), (7.5.8), (7.5.9), the optimization problem is reformulated as

$$\boldsymbol{\Gamma}, \boldsymbol{\Gamma}^*, \boldsymbol{\Theta} = \underset{\boldsymbol{\Gamma},\boldsymbol{\Gamma}^*\in\mathbb{R}^{d\times K},\boldsymbol{\Theta}\in\mathbb{R}^{(n_s+n_t)\times d}}{\arg\min} \mathcal{L}(\boldsymbol{\Gamma}, \boldsymbol{\Gamma}^*, \boldsymbol{\Theta}), \tag{7.5.10}$$

where $\mathcal{L}(\boldsymbol{\Gamma}, \boldsymbol{\Gamma}^*, \boldsymbol{\Theta})$ is

$$\begin{aligned} &\mathrm{tr}\big((\boldsymbol{\Theta\Gamma})^T(\mathbf{K}(\mathbf{A}^2 + \rho\mathbf{M} + \rho\mathbf{L})\mathbf{K} + \sigma\mathbf{K})\boldsymbol{\Theta\Gamma}\big) \\ &+\lambda\mathrm{tr}\big((\boldsymbol{\Theta\Gamma}^*)^T(\mathbf{K}(\mathbf{A}^{*2} + \rho\mathbf{M}^* + \rho\mathbf{L}^*)\mathbf{K} + \sigma\mathbf{K})\boldsymbol{\Theta\Gamma}^*\big) \\ &-2\mathrm{tr}\big(\mathbf{Y}^T\mathbf{A}^2\mathbf{K}\boldsymbol{\Theta\Gamma}\big) - 2\lambda\mathrm{tr}\big(\mathbf{Y}^T\mathbf{A}^{*2}\mathbf{K}\boldsymbol{\Theta\Gamma}^*\big). \end{aligned} \tag{7.5.11}$$

7.5.3 Analytical Solution

We theoretically analyse the optimization problem (7.5.10) and the following theorem tells us that the optimization problem (7.5.10) has countless solutions.

Theorem 20. *If the optimization problem*

$$\min_{\boldsymbol{\Gamma},\boldsymbol{\Gamma}^*\in\mathbb{R}^{d\times K},\boldsymbol{\Theta}\in\mathbb{R}^{(n_s+n_t)\times d}} \mathcal{L}(\boldsymbol{\Gamma}, \boldsymbol{\Gamma}^*, \boldsymbol{\Theta}),$$

has a solution, then the optimization problem has countless solutions, where $\mathcal{L}(\boldsymbol{\Gamma}, \boldsymbol{\Gamma}^*, \boldsymbol{\Theta})$ *is defined in Eq. (7.5.11).*

Proof of Theorem 20. We assume a solution to the problem is $\underline{\boldsymbol{\Gamma}}, \underline{\boldsymbol{\Gamma}}^*, \underline{\boldsymbol{\Theta}}$. Given any inverse matrix $\mathbf{F} \in \mathbb{R}^{d\times d}$, then we need to prove that $\mathbf{F}\underline{\boldsymbol{\Gamma}}, \mathbf{F}\underline{\boldsymbol{\Gamma}}^*$

and $\underline{\mathbf{\Theta}}^*\mathbf{F}^{-1}$ is a different solution. This is because

$$\begin{aligned}
&\mathcal{L}(\underline{\mathbf{\Gamma}}, \underline{\mathbf{\Gamma}}^*, \underline{\mathbf{\Theta}}) \\
=&\mathrm{tr}\big((\underline{\mathbf{\Theta}\mathbf{\Gamma}})^T(\mathbf{K}(\mathbf{A}^2 + \rho\mathbf{M} + \rho\mathbf{L})\mathbf{K} + \sigma\mathbf{K})\underline{\mathbf{\Theta}\mathbf{\Gamma}}\big) \\
+&\lambda\mathrm{tr}\big((\underline{\mathbf{\Theta}\mathbf{\Gamma}}^*)^T(\mathbf{K}(\mathbf{A}^{*2} + \rho\mathbf{M}^* + \rho\mathbf{L}^*)\mathbf{K} + \sigma\mathbf{K})\underline{\mathbf{\Theta}\mathbf{\Gamma}}^*\big) \\
-&2\mathrm{tr}\big(\mathbf{Y}^T\mathbf{A}^2\mathbf{K}\underline{\mathbf{\Theta}\mathbf{\Gamma}}\big) - 2\lambda\mathrm{tr}\big(\mathbf{Y}^T\mathbf{A}^{*2}\mathbf{K}\underline{\mathbf{\Theta}\mathbf{\Gamma}}^*\big) \\
=&\mathrm{tr}\big((\underline{\mathbf{\Theta}}\mathbf{F}^{-1}\mathbf{F}\underline{\mathbf{\Gamma}})^T(\mathbf{K}(\mathbf{A}^2 + \rho\mathbf{M} + \rho\mathbf{L})\mathbf{K} + \sigma\mathbf{K})\underline{\mathbf{\Theta}}\mathbf{F}^{-1}\mathbf{F}\underline{\mathbf{\Gamma}}\big) \\
+&\lambda\mathrm{tr}\big((\underline{\mathbf{\Theta}}\mathbf{F}^{-1}\mathbf{F}\underline{\mathbf{\Gamma}}^*)^T(\mathbf{K}(\mathbf{A}^{*2} + \rho\mathbf{M}^* + \rho\mathbf{L}^*)\mathbf{K} + \sigma\mathbf{K})\underline{\mathbf{\Theta}}\mathbf{F}^{-1}\mathbf{F}\underline{\mathbf{\Gamma}}^*\big) \\
-&2\mathrm{tr}\big(\mathbf{Y}^T\mathbf{A}^2\mathbf{K}\underline{\mathbf{\Theta}}\mathbf{F}^{-1}\mathbf{F}\underline{\mathbf{\Gamma}}\big) - 2\lambda\mathrm{tr}\big(\mathbf{Y}^T\mathbf{A}^{*2}\mathbf{K}\underline{\mathbf{\Theta}}\mathbf{F}^{-1}\mathbf{F}\underline{\mathbf{\Gamma}}^*\big) \\
=&\mathcal{L}(\mathbf{F}\underline{\mathbf{\Gamma}}, \mathbf{F}\underline{\mathbf{\Gamma}}^*, \underline{\mathbf{\Theta}}\mathbf{F}^{-1})
\end{aligned} \tag{7.5.12}$$

Note that the number of inverse matrices is countless, hence, we can construct countless solutions. The proof has been completed. □

According to Theorem 20, the optimization problem (7.5.10) has countless solutions, hence, we do not know which solution is the best. However, since we are only interested in $\mathbf{\Theta\Gamma}$ and $\mathbf{\Theta\Gamma}^*$, we investigate whether the optimization problem (7.5.10) can be transformed to an optimization problem with respective to $\mathbf{\Theta\Gamma}$ and $\mathbf{\Theta\Gamma}^*$ in Theorem 21.

Theorem 21. *Given any optimal solution $\underline{\mathbf{\Gamma}}, \underline{\mathbf{\Gamma}}^*, \underline{\mathbf{\Theta}}$ of the optimization problem (7.5.10), if the source kernel k_s and target kernel k_t are universal, and the latent subspace dimension $d \geq 2K$, then*

$$\underline{\mathbf{\Theta}\mathbf{\Gamma}}, \quad \underline{\mathbf{\Theta}\mathbf{\Gamma}}^*$$

is the unique solution of the optimization problem

$$\min_{\mathbf{Z}, \mathbf{Z}^* \in \mathbb{R}^{(n_s+n_t)\times K}} \mathcal{L}(\mathbf{Z}, \mathbf{Z}^*), \quad \textit{where} \tag{7.5.13}$$

$$\begin{aligned}
\mathcal{L}(\mathbf{Z}, \mathbf{Z}^*) = &\mathrm{tr}\big(\mathbf{Z}^T(\mathbf{K}(\mathbf{A}^2 + \rho\mathbf{M} + \rho\mathbf{L})\mathbf{K} + \sigma\mathbf{K})\mathbf{Z}\big) \\
&+ \lambda\mathrm{tr}\big(\mathbf{Z}^{*T}(\mathbf{K}(\mathbf{A}^{*2} + \rho\mathbf{M}^* + \rho\mathbf{L}^*)\mathbf{K} + \sigma\mathbf{K})\mathbf{Z}^*\big) \\
&- 2\mathrm{tr}\big(\mathbf{Y}^T\mathbf{A}^2\mathbf{K}\mathbf{Z}\big) - 2\lambda\mathrm{tr}\big(\mathbf{Y}^T\mathbf{A}^{*2}\mathbf{K}\mathbf{Z}^*\big).
\end{aligned}$$

Proof of Theorem 21. Let $n = n_s + n_t$. We note that

$$\{\mathbf{\Theta} \cdot \mathbf{\Gamma} |\ \mathbf{\Theta} \in \mathbb{R}^{n\times d}, \mathbf{\Gamma} \in \mathbb{R}^{d\times K}\} \subset \{\mathbf{Z} |\ \mathbf{Z} \in \mathbb{R}^{n\times K}\}.$$

Hence,

$$\min_{\mathbf{\Gamma},\mathbf{\Gamma}^*\in\mathbb{R}^{d\times K},\mathbf{\Theta}\in\mathbb{R}^{n\times d}} \mathcal{L}(\mathbf{\Gamma},\mathbf{\Gamma}^*,\mathbf{\Theta}) \geq \min_{\mathbf{Z},\mathbf{Z}^*\in\mathbb{R}^{n\times K}} \mathcal{L}(\mathbf{Z},\mathbf{Z}^*). \tag{7.5.14}$$

It is notable that

$$\mathbf{K}(\mathbf{A}^2 + \rho\mathbf{M} + \rho\mathbf{L})\mathbf{K} + \sigma\mathbf{K}, \quad \mathbf{K}(\mathbf{A}^{*2} + \rho\mathbf{M}^* + \rho\mathbf{L}^*)\mathbf{K} + \sigma\mathbf{K}$$

are strictly positive matrices. Hence, the problem $\min_{\mathbf{Z},\mathbf{Z}^*\in\mathbb{R}^{n\times K}} \mathcal{L}(\mathbf{Z},\mathbf{Z}^*)$ has a solution and the solution is unique.

Let $\underline{\mathbf{Z}}$ and $\underline{\mathbf{Z}}^*$ be the solution of the problem $\min_{\mathbf{Z},\mathbf{Z}^*\in\mathbb{R}^{n\times K}} \mathcal{L}(\mathbf{Z},\mathbf{Z}^*)$. We define

$$\underline{\Theta} = [\underline{\mathbf{Z}}, \underline{\mathbf{Z}}^*, \mathbf{O}_{n\times(d-2K)}],$$
$$\underline{\Gamma} = \begin{bmatrix} \mathbf{I}_{K\times K} \\ \mathbf{O}_{K\times K} \\ \mathbf{O}_{(d-2K)\times K} \end{bmatrix}, \underline{\Gamma}^* = \begin{bmatrix} \mathbf{O}_{K\times K} \\ \mathbf{I}_{K\times K} \\ \mathbf{O}_{(d-2K)\times K} \end{bmatrix},$$

where $\mathbf{I}$ is the identity matrix. Then

$$\underline{\mathbf{Z}} = \underline{\mathbf{\Theta}} \cdot \underline{\mathbf{\Gamma}}, \quad \underline{\mathbf{Z}}^* = \underline{\mathbf{\Theta}} \cdot \underline{\mathbf{\Gamma}}^*,$$

which implies that

$$\min_{\mathbf{\Gamma},\mathbf{\Gamma}^*\in\mathbb{R}^{d\times K},\mathbf{\Theta}\in\mathbb{R}^{n\times d}} \mathcal{L}(\mathbf{\Gamma},\mathbf{\Gamma}^*,\mathbf{\Theta}) \leq \min_{\mathbf{Z},\mathbf{Z}^*\in\mathbb{R}^{n\times K}} \mathcal{L}(\mathbf{Z},\mathbf{Z}^*). \tag{7.5.15}$$

Combining inequalities (7.5.14), (7.5.15), we have

$$\min_{\mathbf{\Gamma},\mathbf{\Gamma}^*\in\mathbb{R}^{d\times K},\mathbf{\Theta}\in\mathbb{R}^{n\times d}} \mathcal{L}(\mathbf{\Gamma},\mathbf{\Gamma}^*,\mathbf{\Theta}) = \min_{\mathbf{Z},\mathbf{Z}^*\in\mathbb{R}^{n\times K}} \mathcal{L}(\mathbf{Z},\mathbf{Z}^*). \tag{7.5.16}$$

Lastly, let $\underline{\mathbf{\Theta}}, \underline{\mathbf{\Gamma}}, \underline{\mathbf{\Gamma}}^*$ be a solution of $\min_{\mathbf{\Gamma},\mathbf{\Gamma}^*\in\mathbb{R}^{d\times K},\mathbf{\Theta}\in\mathbb{R}^{n\times d}} \mathcal{L}(\mathbf{\Gamma},\mathbf{\Gamma}^*,\mathbf{\Theta})$.

Then, according to Eq. (7.5.16), we have

$$\mathcal{L}(\underline{\mathbf{\Theta}}, \underline{\mathbf{\Gamma}}, \underline{\mathbf{\Gamma}}^*) = \min_{\mathbf{Z},\mathbf{Z}^* \in \mathbb{R}^{n\times K}} \mathcal{L}(\mathbf{Z}, \mathbf{Z}^*) = \mathcal{L}(\underline{\mathbf{\Theta}} \cdot \underline{\mathbf{\Gamma}}, \underline{\mathbf{\Theta}} \cdot \underline{\mathbf{\Gamma}}^*). \tag{7.5.17}$$

Hence, $\underline{\mathbf{\Theta}} \cdot \underline{\mathbf{\Gamma}}, \underline{\mathbf{\Theta}} \cdot \underline{\mathbf{\Gamma}}^*$ is the solution of the problem $\min_{\mathbf{Z},\mathbf{Z}^* \in \mathbb{R}^{n\times K}} \mathcal{L}(\mathbf{Z}, \mathbf{Z}^*)$. In the second step, we have proven that the solution of the problem $\min_{\mathbf{Z},\mathbf{Z}^* \in \mathbb{R}^{n\times K}} \mathcal{L}(\mathbf{Z}, \mathbf{Z}^*)$ is unique. Hence, for any solution $\underline{\mathbf{\Theta}}, \underline{\mathbf{\Gamma}}, \underline{\mathbf{\Gamma}}^*$, we have proven that $\underline{\mathbf{\Theta}} \cdot \underline{\mathbf{\Gamma}}, \underline{\mathbf{\Theta}} \cdot \underline{\mathbf{\Gamma}}^*$ are fixed and equal to $\underline{\mathbf{Z}}, \underline{\mathbf{Z}}^*$.

We have completed the proof. □

Theorem 21 implies that an important result though the solutions of problem (7.5.10) are not unique, $\underline{\mathbf{\Theta\Gamma}}, \underline{\mathbf{\Theta\Gamma}}^*$ are fixed and are the unique solution of the problem (7.5.13). Then, the analytical solution to problem (7.5.13) is presented in the following theorem.

Theorem 22. *If the kernels k_s and k_t are universal, then the optimization problem (7.5.13) has a unique solution, which can be formulated as follows:*

$$\underline{\mathbf{Z}} = \left((\mathbf{A}^2 + \rho\mathbf{M} + \rho\mathbf{L})\mathbf{K} + \sigma\mathbf{I}\right)^{-1}\mathbf{A}^2\mathbf{Y}, \tag{7.5.18}$$

$$\underline{\mathbf{Z}}^* = \left((\mathbf{A}^{*2} + \rho\mathbf{M}^* + \rho\mathbf{L}^*)\mathbf{K} + \sigma\mathbf{I}\right)^{-1}\mathbf{A}^{*2}\mathbf{Y}. \tag{7.5.19}$$

Proof of Theorem 22. Note that $\mathcal{L}(\mathbf{Z}, \mathbf{Z}^*)$ is a convex function about $\mathbf{Z}, \mathbf{Z}^*$.

Hence, we compute $\frac{\partial \mathcal{L}}{\partial \mathbf{Z}}(\underline{\mathbf{Z}}, \underline{\mathbf{Z}}^*) = \mathbf{0}, \quad \frac{\partial \mathcal{L}}{\partial \mathbf{Z}^*}(\underline{\mathbf{Z}}, \underline{\mathbf{Z}}^*) = \mathbf{0}$:

$$\begin{aligned} \mathbf{0} &= -2\mathbf{K}\mathbf{A}^2\mathbf{Y} + 2\sigma\mathbf{K}\mathbf{Z} + 2\mathbf{K}\mathbf{A}^2\mathbf{K}\mathbf{Z} + \mathbf{K}(\rho\mathbf{M} + \rho\mathbf{L})\mathbf{K}\mathbf{Z}, \\ \mathbf{0} &= -2\mathbf{K}\mathbf{A}^{*2}\mathbf{Y} + 2\sigma\mathbf{K}\mathbf{Z}^* + 2\mathbf{K}\mathbf{A}^{*2}\mathbf{K}\mathbf{Z}^* + \mathbf{K}(\rho\mathbf{M}^* + \rho\mathbf{L}^*)\mathbf{K}\mathbf{Z}^* \end{aligned} \tag{7.5.20}$$

The solution of (7.5.20) can be written as:

$$\underline{\mathbf{Z}} = \left((\mathbf{A}^2 + \rho\mathbf{M} + \rho\mathbf{L})\mathbf{K} + \sigma\mathbf{I}\right)^{-1}\mathbf{A}^2\mathbf{Y},$$

$$\underline{\mathbf{Z}}^* = \left((\mathbf{A}^{*2} + \rho\mathbf{M}^* + \rho\mathbf{L}^*)\mathbf{K} + \sigma\mathbf{I}\right)^{-1}\mathbf{A}^{*2}\mathbf{Y}.$$

□

Algorithm 4 KHDA algorithm for SsHeDA

1: Input Data $\mathcal{S}, \mathcal{T}_u, \mathcal{T}_l$; #Iterations T;
Parameters σ, ρ; #Neighbor p; Kernel k_s, k_t;
2: Obtain $Y_u = \text{1NN}(\mathbf{X}_u)$; //1-Nearest Neighbor
3: Assign $Y_u^* = Y_u$;
for $i = 1, 2, ..., T$ **do**;
4: Compute $\underline{\mathbf{Z}}, \underline{\mathbf{Z}}^*$ by Eqs. (7.5.18), (7.5.19) with pseudo labels Y_u, Y_u^*;
5: Obtain $\boldsymbol{h} \circ \boldsymbol{T}_t, \boldsymbol{h}^* \circ \boldsymbol{T}_t$ by $\underline{\mathbf{Z}}, \underline{\mathbf{Z}}^*$ (Eqs. (7.5.21), (7.5.22));
6: Update Y_u^* by $\boldsymbol{h}^* \circ \boldsymbol{T}_t(\mathbf{X}_u)$;
7: Compute $\underline{\mathbf{Z}}$ by Eq. (7.5.18) with pseudo labels Y_u^*;
8: Obtain $\widetilde{\boldsymbol{h}} \circ \boldsymbol{T}_t$ by $\underline{\mathbf{Z}}$ and Eq. (7.5.21);
9: Obtain the classifier $\boldsymbol{f}$ by Eq. (7.5.23);
10: Update pseudo labels Y_u by $\boldsymbol{f}(\mathbf{X}_u)$;
end for;
11: Output predicted target labels Y_u.

Based on Theorem 22, $\boldsymbol{h} \circ \boldsymbol{T}_t$ and $\boldsymbol{h}^* \circ \boldsymbol{T}_t$ can be written as

$$\boldsymbol{h} \circ \boldsymbol{T}_t(\mathbf{x}) = \sum_{i=1}^{n_t} \underline{\mathbf{Z}}_i k_t(\mathbf{x}, \mathbf{x}_i), \tag{7.5.21}$$

$$\boldsymbol{h}^* \circ \boldsymbol{T}_t(\mathbf{x}) = \sum_{i=1}^{n_t} \underline{\mathbf{Z}}_i^* k_t(\mathbf{x}, \mathbf{x}_i), \tag{7.5.22}$$

where $\mathbf{x} \in \mathcal{X}_t$, $\mathbf{x}_i \in \mathbf{X}_t$, and $\underline{\mathbf{Z}}_i$, $\underline{\mathbf{Z}}_i^*$ are $(i + n_s)$-th rows of matrices $\underline{\mathbf{Z}}$ and $\underline{\mathbf{Z}}^*$, respectively.

7.5.4 KHDA Algorithm

To compute Eq. (7.5.18) and Eq. (7.5.19) accurately, the labels of the unlabeled target samples are required. However, we have no label information related to these samples. A simple and effective method is to use the pseudo-label iterative strategy proposed by JDA [65]. Motivated by pseudo-label iterative strategy, Algorithm 4 presents how to iteratively improve the quality of Eq. (7.5.18) and Eq. (7.5.19) and give a final kernel-based solution to the SsHeDA problem. In the following, we explain three main steps in Algorithm 4.

Step 1 (Initialize pseudo labels, lines 2-3). Using 1NN with labeled target samples, we predict the pseudo target labels Y_u. Then, we set $Y_u^* = Y_u$.

Step 2 (Construct combination classifier, lines 4-9). Using Eqs. (7.5.19), (7.5.18) with the pseudo labels Y_u^* and Y_u, we obtain $\boldsymbol{h}^* \circ \boldsymbol{T}_t, \boldsymbol{h} \circ \boldsymbol{T}_t$.

To link $\boldsymbol{h}^* \circ \boldsymbol{T}_t, \boldsymbol{h} \circ \boldsymbol{T}_t$, we update the pseudo label Y_u^* by classifier $\boldsymbol{h}^* \circ \boldsymbol{T}_t$, then use Eq. (7.5.18) with the pseudo label Y_u^* to learn the third classifier $\widetilde{\boldsymbol{h}} \circ \boldsymbol{T}_t$.

Next, we advocate taking advantage of the complementary of $\boldsymbol{h} \circ \boldsymbol{T}_t$, $\boldsymbol{h}^* \circ \boldsymbol{T}_t$ and $\widetilde{\boldsymbol{h}} \circ \boldsymbol{T}_t$ via a simple combination.

$$f_c(\mathbf{x}) = \max\{h_c \circ \boldsymbol{T}_t(\mathbf{x}), h_c^* \circ \boldsymbol{T}_t(\mathbf{x}), \widetilde{h}_c \circ \boldsymbol{T}_t(\mathbf{x})\}, \tag{7.5.23}$$

where h_c , h_c^* and $\widetilde{h}_c$ are the c-th coordinate of $\boldsymbol{h}$, $\boldsymbol{h}^*$, $\widetilde{\boldsymbol{h}}$.

As a result, the pseudo label of a given target sample $\mathbf{x}$ can be predicted by

$$\arg\max_{c \in \mathcal{Y}} \ [f_1(\mathbf{x}), ..., f_c(\mathbf{x}), ..., f_K(\mathbf{x})].$$

Using $\boldsymbol{f}$, we obtain the pseudo labels Y_u, where $\boldsymbol{f} = [f_1, ..., f_c, .., f_K]$.

Step 3. We repeat Step 2 until convergence. Lastly, we choose $\boldsymbol{f}$ as the final classifier.

7.6 Experiments and Evaluations

In the section, we empirically evaluate the proposed algorithms KHDA and JMEA on different transfer tasks. We then conducted the experiments to examine the behavior of the parameters.

7.6.1 Datasets and Baseline Algorithms

Image↔Image. Office+Caltech256 [130, 149] consists of 2,533 images in 10 classes, forming four different domains: **A** (images from amazon.com), **W** (low-resolution images captured by webcams), **D** (high-resolution images captured by a digital SLR camera), and C (images from the Caltech256 dataset). Following the settings in [122, 124], two types of features are used: SURF [149] and DeCAF6 [130]. Three do-

mains **A**, **W**, **C** with SURF features are used as the source domains and four domains **A**, **W**, **D**, **C** with DeCAF6 features are used as the target domain (SURF→DeCAF6). Hence, there are $3 \times 4 = 12$ SsHeDA tasks. Following the setting reported in [122], 20 images per class are chosen randomly as the source samples, 3 images per class are chosen randomly as the labeled target samples, and the remaining samples in the target domain are regarded as the unlabeled samples. The average accuracy and standard error of 20 random trials are shown in Table 7.2.

Text↔Text. Reuters-21578 [25] contains three different datasets: OrgsPeople (OPe) ($4,772$ dimensional features and 2 classes: Orgs, People), OrgsPlace (OPl) ($4,416$ dimensional features and 2 classes: Orgs and Place), and PeoplePlaces (PPl) ($4,563$ dimensional features and 2 classes: People and Place). Each dataset consists of two domains, which are from same feature space, but different distributions. OPe consists of **OrgsPeople_src** (**OPes**) and **OrgsPeople_tar** (**OPet**), OPl contains **OrgsPlace_src** (**OPls**) and **OrgsPlace_tar** (**OPlt**), and PPl contains **PeoplePlaces_src** (**PPls**) and **PeoplePlaces_tar** (**PPlt**). Following the setting in [25], we construct 6 heterogeneous domain adaptation tasks: **OPes** → **OPlt**, **OPes** → **PPlt**, **OPls** → **OPet**, **OPls** → **PPlt**, **PPls** → **OPet** and **PPls** → **OPlt**. 5 samples per class in the target domain are chosen randomly as the labeled target samples, and the remaining samples in the target domain are regarded as the unlabeled samples. The average accuracy and standard error of 20 random trials are shown in Table 7.4.

Text↔Image. Wikipedia dataset [201,202] is extracted from Wikipedia feature articles and consists of $2,866$ image-text pairs with 10 semantic classes. We use the VGG-16 features [105] ($4,096$ dimensional features) to represent the images (**I**). For text features (**T**), we adopt *word2vec* model to learn the 100 dimensional skip-gram word vectors [203]. 50 samples per class in the source domain are selected randomly as the labeled source samples. For the target domain, we choose numbers of $5, 10, 15, 20$ per class as labeled target samples, and randomly choose 50

samples per class in the remaining samples as the unlabeled target samples. There are $4 \times 2 = 8$ SsHeDA tasks. The average accuracy and standard error of 20 random trials are shown in Table 7.5.

Text↔Text. Multilingual Reuters Collection (MRC) [150, 204] is a text dataset applied for multi-lingual text categorization and consists of $11,000$ articles from six categories in five languages, i.e., **English, French, German, Italian, and Spanish**. Following the same settings in previous work [122], we use BOW with TF-ITF to describe each article. Then we use PCA in the BoW features to preserve 60% energy. After the dimensionality reduction, the dimensions of the five languages English, French, Italian, German, and Spanish are $1,131$, $1,230$, $1,417$, $1,041$, and 807, respectively. Following the previous work [122, 190], we set **English, French, Italian** and **German** as the source domains and **Spanish** as the target domain. 100 samples per class in the source domain are selected randomly as the labeled source samples. For the target domain, we choose numbers of $5, 10, 15, 20$ per class as the labeled target samples and randomly choose 500 samples per class in the remaining samples as the unlabeled target samples. There are $4 \times 4 = 16$ SsHeDA tasks. The average accuracy and standard error of 20 random trials are shown in Table 7.6.

Image↔Image. ImageCLEF-DA [47] is a benchmark dataset for ImageCLEF 2014 domain adaptation challenge, which aims to classify 12 categories shared in three datasets: Caltech-256 (**C**), ImageNet ILSVRC 2012 (**I**), Bing (**B**) and PascalVOC 2012 (**P**). We use the ResNet-50 features as the source domain and VGG-16 features as the target domain. We also use the VGG-16 features as the source domain and ResNet-50 features as the target domain. By considering each as a domain, we build 32 transfer tasks: **I** (ResNet-50) → **I** (VGG-16), **I** (ResNet-50) → **P** (VGG-16), **I** (ResNet-50) → **B** (VGG-16),..., **C** (VGG-16) → **C** (ResNet-50). We randomly select 20 labeled source samples per class and 3 labeled target samples per class for training, and use the rest target samples as unlabeled samples. The average accuracy and standard error

Table 7.1: Parameters for different tasks on KHDA.

Tasks	ρ	p	σ	T
Office+Caltech256	1.0	10.0	0.01	10.0
Retuters-21578	10.0	10.0	0.01	10.0
Wikipedia	1.0	10.0	0.01	10.0
ImageCLEF-DA	1.0	10.0	0.01	10.0
MRC	10.0	10.0	0.01	10.0

of 20 random trials are shown in Tables 7.7 and 7.8.

• **Baseline Algorithms.** The baseline algorithms are 1NN, SVMt, DAMA [120], SHFA [24], G-JDA [122], CDLS [30], DACoM [123].

7.6.2 Experimental Setup

Before detailing the evaluation results, we explain how the parameters of KHDA are set. There are several parameters: 1) the kernel k_s and k_t; 2) #iterations T; 3) σ, ρ, #neighbor p.

As suggested in [62, 96], we choose the Gaussian kernel

$$k_s(\mathbf{x}_s, \mathbf{x}_s') = \exp\Big(-\frac{\|\mathbf{x}_s - \mathbf{x}_s'\|_2^2}{2r_s^2}\Big),$$
$$k_t(\mathbf{x}_t, \mathbf{x}_t') = \exp\Big(-\frac{\|\mathbf{x}_t - \mathbf{x}_t'\|_2^2}{2r_t^2}\Big),$$

where $\mathbf{x}_s, \mathbf{x}_s' \in \mathbf{X}_s$, $\mathbf{x}_t, \mathbf{x}_t' \in \mathbf{X}_t$ and the kernel bandwidth r_s is median($\|\mathbf{x}_s - \mathbf{x}_s'\|_2$), $\forall \mathbf{x}_s, \mathbf{x}_s' \in \mathbf{X}_s$ and the kernel bandwidth r_t is median($\|\mathbf{x}_t - \mathbf{x}_t'\|_2$), $\forall \mathbf{x}_t, \mathbf{x}_t' \in \mathbf{X}_t$.

The details of the parameters #iterations T, σ, ρ, p are shown in Table 7.1.

The classification accuracy [65] on the test samples is

$$Accuracy = \frac{\sum_{c=1}^{K} |\mathbf{x} \in \mathbf{X}_u^c : g(\mathbf{x}) = c|}{|\mathbf{x} : \mathbf{x} \in \mathbf{X}_u|},$$

where g is the predicted classifier and $\mathbf{X}_u^c$ is the unlabeled target data matrix with true label c.

7.6.3 Experimental Results

The classification accuracy and standard error on different tasks are shown in Tables 7.2, 7.3, 7.4, 7.5, 7.6, 7.7 and 7.8. In Tables 7.2-7.8,

the *bold* color indicates the best accuracy.

• **Office+Caltech256** [130, 149]**.** 1) In Tables 7.2 and 7.3, compared to all baselines, KHDA works the best for almost all tasks (22/24) and the mean accuracy achieves an improvement at least 1.5%. 2) Compared to the best baseline algorithm CDLS, KHDA achieves 2.2% and 1.7% mean improvements, respectively. 3) Except for DAMA, baselines G-JDA, CDLS and DACoM achieve better mean performance than 1NN and SVMt (at least 1.5%). This indicates that the baselines SHFA, G-JDA, CDLS and DACoM can transfer knowledge from source samples to target samples.

• **Retuters-21578** [25]**.** The results for Retuters-21578 dataset are reported in Table 7.4. 1) Compared to all baselines, KHDA works the best for almost all tasks (5/6) and has achieved a mean improvement at least 3.6%. 2) Among all baselines, DACoM achieves best accuracy on task OPls→PPlt, even better than KHDA. However, on other task, its accuracy is lower than that of KHDA.

• **Wikipedia** [201,202]**.** Table 7.5 presents results for Wikipedia dataset. 1) KHDA has achieved at least 0.7% mean improvement (different number of labeled target samples per class), compared to the best baseline SHFA. 2) It is notable that the accuracy of all algorithms increases when using more labeled target samples per class. 3) Among all baselines, SHFA works the best for almost all tasks (6/8). This indicates that the feature augmentation used in SHFA is effective on Wikipedia tasks.

• **Multilingual Reuters Collection** [150, 204]**.** Table 7.6 shows the means and standard errors of classification accuracy for all algorithms on the Multilingual Reuters Collection when there are 5, 10, 15 and 20 labeled target samples per class. 1) Compared to all baselines, KHDA performs the best on 15 tasks (15/16). 2) According to the results from Table 7.6, the accuracy of all algorithms increases when using more labeled target samples per class.

• **ImageCLEF-DA** [47]**.** Table 7.7 and 7.8 show the means and standard errors of classification accuracy for all algorithms on the Multilingual

Reuters Collection when there are $5, 10, 15$ and 20 labeled target samples per class. 1) Compared to all baselines, KHDA performs the best on 15 tasks (15/16). 2) According to the results from Table 7.6, the accuracy of all algorithms increases when using more labeled target samples per class.

• **Summary.** 1) Generally, KHDA outperforms baselines on most tasks (79/86). 2) According to the results from Tables 7.5, 7.6, 7.7 and 7.8, the accuracy of all baselines increases when using more labeled target samples per class.

7.6.4 Ablation Study

Here we present the ablation study for KHDA. This study is conducted on different types of datasets: Office+Caltech256, Reuters-21587 and Wikipedia (10 labeled target samples per class), which involves 20 tasks. We report average accuracy for different dataset.

7.6.4.1 Ablation Study for KHDA

We conduct thorough experiments to show the contribution of individual components in KHDA in Table 7.9. We consider the following baselines:

- w/o D: In KHDA, train classifiers without distribution alignment Eq. (7.5.3).
- w/o M: In KHDA, train classifiers without manifold regularization Eq. (7.5.9).
- $\rho = 0$: In KHDA, train classifiers without the combination of manifold regularization and distribution alignment.
- $\sigma = 0$: In KHDA, train classifiers without kernel regularization.
- $\boldsymbol{h}$, $\boldsymbol{h}^*$, $\widetilde{\boldsymbol{h}}$: In KHDA, the performance of $\boldsymbol{h}$, $\boldsymbol{h}^*$ and $\widetilde{\boldsymbol{h}}$.
- w/o $\boldsymbol{h}$: In KHDA, train classifiers without $\boldsymbol{h}$.
- w/o $\boldsymbol{h}^*$: In KHDA, train classifiers without $\boldsymbol{h}^*$.

- w/o $\widetilde{\boldsymbol{h}}$: In KHDA, train classifiers without $\widetilde{\boldsymbol{h}}$.

1) When we omit the distribution alignment (i.e., w/o D), the mean accuracy drops from 81.8% to 77.7%. This indicates that the distribution alignment is important for KHDA. If we omit the manifold regularization (i.e., w/o M), the mean accuracy 78.4% is worse than the mean accuracy 81.8% of KHDA. This indicates that the manifold regularization is necessary for KHDA.

2) If we set $\rho = 0$, then the mean accuracy 68.0% is much lower than the mean accuracy 81.8% of KHDA. This indicates that the combination of manifold regularization and distribution alignment can greatly improve the performance of KHDA. If we set $\sigma = 0$, then the mean accuracy 62.3% is much lower than the mean accuracy 81.8% of KHDA with $\sigma = 0.01$. One possible reason for this is that the overfitting occurs when $\sigma = 0$.

3) We observe that the performance of $\boldsymbol{h}^*$ (73.5%) and $\widetilde{\boldsymbol{h}}$ (77.4%) is much worse than the performance of $\boldsymbol{h}$ (81.8%) and $\boldsymbol{f}$ (81.8%). The performance of $\boldsymbol{h}$ and $\boldsymbol{f}$ is the same, which shows that $\boldsymbol{h}$ can also be used as the final classifier.

4) If we omit $\boldsymbol{h}$ (i.e., w/o $\boldsymbol{h}$), $\boldsymbol{h}^*$ (i.e., w/o $\boldsymbol{h}^*$) and $\widetilde{\boldsymbol{h}}$ (i.e., w/o $\widetilde{\boldsymbol{h}}$), respectively, the mean performance of KHDA (81.8%) decreases to 77.3%, 81.1% and 81.2%, respectively. This implies that every classifier is important for KHDA.

7.7 Summary

This chapter provides an in-depth analysis of SsHeDA that gives rise to a comprehensive theory of the SsHeDA problem, a novel perspective for analyzing domain adaptation problems, and new mathematical tools for solving the SsHeDA problem. The theoretical result is the first-ever explanation of why labeled source samples combined with unlabeled target samples helps to reduce the need for labeled samples in the target domain. Then we propose a novel SsHeDA algorithm to bring the proposed theory into reality: KHDA. KHDA is kernel-based and is well-suited to

situations with small amounts of samples. In sweeping experiments with 7 baselines and 86 different SsHeDA tasks, KHDA is better than all baselines.

Table 7.2: Accuracy (%) with standard error on Office+Caltech256 dataset from SURF → DeCAF6.

Methods	1NN	SVMt	DAMA	SHFA	G-JDA	CDLS	DACoM	KHDA
A→A	85.7±0.5	86.4±0.5	87.5±0.3	89.0±0.3	92.3±0.2	92.9±0.1	92.2±0.3	**93.3**±0.2
A→W	87.8±0.8	86.4±0.8	87.0±0.8	89.7±0.7	91.1±1.0	92.8±0.8	91.5±0.8	**96.3**±0.4
A→D	90.5±0.6	90.4±0.7	91.4±0.6	92.4±0.7	94.3±0.7	92.6±0.8	92.9±0.8	**95.0**±0.4
A→C	73.7±0.7	76.7±0.7	71.8±0.9	79.8±0.5	86.9±0.4	84.3±0.6	84.5±0.5	**87.5**±0.4
W→A	85.7±0.5	86.4±0.5	85.6±0.3	89.0±0.2	92.8±0.2	93.1±0.1	88.6±0.2	**93.2**±0.2
W→W	87.8±0.8	86.4±0.8	87.9±0.8	89.8±0.7	89.4±0.9	93.8±0.7	93.9±0.7	**96.3**±0.4
W→D	90.5±0.6	90.4±0.7	94.1±0.4	92.4±0.7	**95.1**±0.4	92.7±0.8	91.7±0.7	95.0±0.5
W→C	73.7±0.7	76.7±0.7	71.5±0.8	79.8±0.5	86.3±0.5	84.1±0.6	84.7±0.5	**87.6**±0.4
C→A	85.7±0.5	86.4±0.5	87.7±0.8	89.0±0.3	92.6±0.1	92.7±0.1	92.2±0.2	**93.3**±0.2
C→W	87.8±0.8	86.4±0.8	89.1±0.7	89.8±0.7	92.1±1.0	93.0±0.9	92.2±0.9	**96.3**±0.4
C→D	90.5±0.6	90.4±0.7	91.1±0.6	92.3±0.7	93.0±0.8	92.3±1.0	93.5±0.7	**94.7**±0.5
C→C	73.7±0.7	76.7±0.7	76.6±0.8	79.8±0.5	86.7±0.5	84.9±0.5	84.0±0.6	**87.5**±0.4
Avg	84.4	84.9	85.1	87.7	91.0	90.8	90.6	**93.0**

Table 7.3: Accuracy (%) with standard error on Office+Caltech256 dataset from DeCAF6 → SURF.

Methods	1NN	SVMt	DAMA	SHFA	G-JDA	CDLS	DACoM	KHDA
A→A	41.8±0.5	45.2±0.7	45.9±0.6	50.4±0.5	50.3±0.7	**53.9**±0.7	46.3±0.8	53.6±0.7
A→W	61.8±0.8	62.3±0.7	58.6±0.7	64.4±0.8	64.0±1.0	67.3±0.9	69.9±0.9	**70.6**±0.9
A→D	56.3±1.0	58.2±1.0	50.5±1.0	59.1±0.8	56.9±0.8	60.5±1.0	59.3±0.8	**60.8**±0.8
A→C	29.8±0.6	32.0±0.5	31.1±0.6	33.6±0.6	32.7±0.8	33.3±0.7	31.4±0.7	**35.2**±0.9
W→A	41.8±0.5	45.2±0.7	44.3±0.7	46.2±0.5	47.9±0.8	53.0±0.7	49.2±0.9	**53.4**±0.7
W→W	61.8±0.8	62.3±0.7	57.6±0.8	64.5±0.8	63.8±0.9	66.6±0.9	69.9±1.0	**70.4**±0.8
W→D	56.3±1.0	58.2±1.0	54.6±0.9	59.2±0.8	55.5±0.8	60.5±1.2	58.3±0.9	**60.6**±0.8
W→C	29.8±0.6	32.0±0.5	31.1±0.9	33.6±0.6	29.7±0.7	32.6±0.7	30.9±0.8	**35.1**±0.9
C→A	41.8±0.5	45.2±0.7	45.4±0.6	46.2±0.5	51.0±1.0	52.4±0.7	50.0±0.7	**53.2**±0.7
C→W	61.8±0.8	62.3±0.7	58.8±0.7	64.5±0.8	66.6±0.9	65.8±1.0	69.9±0.9	**70.7**±0.9
C→D	56.3±1.0	58.2±1.0	49.3±0.8	59.2±0.7	57.2±1.0	59.5±1.1	59.4±0.8	**60.3**±0.8
C→C	29.8±0.6	32.0±0.5	31.6±0.6	33.5±0.6	33.7±0.8	33.5±0.8	32.2±0.8	**35.4**±0.8
Avg	47.4	49.4	46.6	50.9	50.8	53.2	52.6	**54.9**

Table 7.4: Accuracy (%) with standard error on Reuters-21578 dataset.

Methods	1NN	SVMt	DAMA	SHFA	G-JDA	CDLS	DACoM	KHDA
OPes→OPlt	58.6±1.6	62.7±1.4	63.7±1.3	70.2±0.9	72.1±2.3	65.8±1.4	70.4±1.5	**75.0**±1.0
OPes→PPlt	59.8±1.7	64.8±1.4	58.1±1.0	68.6±0.7	64.9±1.4	55.9±2.2	64.5±1.5	**69.2**±1.6
OPls→OPet	53.8±0.8	57.2±1.0	64.9±1.0	69.1±1.1	72.2±2.3	72.1±1.5	71.1±1.2	**78.4**±0.9
OPls→PPlt	60.0±1.7	64.8±1.4	69.1±0.7	68.7±1.0	65.8±2.4	61.0±1.6	**70.2**±1.2	68.5±1.1
PPls→OPet	53.8±0.8	57.2±1.0	68.0±1.0	68.9±0.9	72.4±2.1	71.6±1.6	72.0±1.2	**76.6**±1.1
PPls→OPlt	58.6±1.6	62.7±1.4	71.6±1.1	72.0±0.9	70.8±1.5	68.3±0.7	72.9±1.3	**74.6**±1.2
Avg	57.4	61.6	65.9	69.6	69.7	65.8	70.2	**73.8**

Table 7.5: Accuracy (%) with standard error on Wikipedia dataset.

Methods	1NN	SVMt	DAMA	SHFA	G-JDA	CDLS	DACoM	KHDA
I→T L5	63.3±0.9	67.2±0.7	69.9±0.8	73.7±0.6	70.5±0.5	71.6±0.7	66.4±0.6	**76.5**±0.5
T→I L5	61.0±0.8	67.8±0.6	69.3±0.8	72.7±0.5	70.6±0.5	70.9±0.6	65.7±0.7	**75.6**±0.4
Avg	62.2	67.5	69.6	73.2	70.6	71.3	66.1	**76.0**
I→T L10	67.5±0.6	71.6±0.5	74.8±0.4	77.1±0.4	73.1±0.5	74.8±0.4	71.6±0.6	**78.5**±0.4
T→I L10	68.1±0.6	71.6±0.5	74.9±0.4	78.3±0.5	73.4±0.5	75.4±0.6	71.8±0.6	**79.9**±0.5
Avg	67.8	71.6	74.9	77.7	73.3	75.1	71.7	**79.2**
I→T L15	70.8±0.5	73.3±0.5	78.5±0.5	80.2±0.4	75.8±0.5	77.6±0.4	74.3±0.5	**80.8**±0.5
T→I L15	70.9±0.5	74.0±0.5	78.6±0.6	80.0±0.5	75.9±0.3	77.8±0.6	75.2±0.4	**81.0**±0.4
Avg	70.9	73.7	78.6	80.1	75.9	77.7	74.8	**80.9**
I→T L20	72.0±0.6	74.2±0.5	79.3±0.5	81.0±0.4	76.9±0.4	78.7±0.5	74.6±0.5	**81.6**±0.4
T→I L20	72.6±0.4	74.8±0.4	79.6±0.4	81.3±0.3	77.6±0.4	79.2±0.4	74.7±0.4	**82.2**±0.3
Avg	72.3	74.5	79.5	81.2	77.3	79.0	74.4	**81.9**

Table 7.6: Accuracy (%) with standard error on Multilingual Reuters Collection (MRC) dataset.

Methods	1NN	SVMt	DAMA	SHFA	G-JDA	CDLS	DACoM	KHDA
English L5	56.1±0.8	47.6±1.5	47.9±1.0	64.3±0.7	63.4±0.7	63.2±0.7	66.6±0.7	**68.2**±0.6
France L5	56.1±0.8	47.6±1.5	46.4±1.0	65.3±0.7	62.5±0.9	62.9±0.7	67.4±0.6	**68.3**±0.6
German L5	56.1±0.8	47.6±1.5	46.9±1.0	64.1±0.7	63.6±1.3	61.3±0.7	**68.2**±0.7	67.6±0.7
Italian L5	56.1±0.8	47.6±1.5	45.8±1.0	65.6±0.7	64.7±1.1	62.6±0.8	67.7±0.8	**68.5**±0.6
Avg	56.1	47.6	46.7	64.0	63.5	62.5	67.5	**68.1**
English L10	63.0±0.5	61.6±0.9	63.7±0.9	70.4±0.5	69.4±0.8	70.0±0.9	72.3±0.6	**73.1**±0.6
France L10	63.0±0.5	61.6±0.9	62.9±0.7	70.5±0.5	70.5±0.7	70.0±0.9	72.7±0.6	**73.3**±0.6
German L10	63.0±0.5	61.6±0.9	62.6±0.9	71.5±0.5	69.6±1.0	69.0±0.8	72.5±0.7	**73.4**±0.6
Italian L10	63.0±0.5	61.6±0.9	63.2±1.0	71.3±0.5	70.1±1.0	69.8±0.9	72.7±0.7	**73.3**±0.6
Avg	63.0	61.6	63.1	70.9	69.9	69.7	72.6	**73.3**
English L15	66.9±0.4	68.0±0.7	69.1±0.5	74.7±0.4	73.8±0.6	73.6±0.6	74.9±0.3	**75.2**±0.4
France L15	66.9±0.4	68.0±0.7	68.3±0.6	73.6±0.4	73.6±0.5	73.4±0.7	75.3±0.5	**75.8**±0.4
German L15	66.9±0.4	68.0±0.7	68.5±0.5	74.3±0.4	74.0±0.5	72.4±0.6	75.7±0.4	**75.8**±0.4
Italian L15	66.9±0.4	68.0±0.7	69.4±0.6	73.9±0.4	73.4±0.5	73.2±0.7	**75.7**±0.4	75.5±0.4
Avg	66.9	68.0	65.4	74.1	73.7	73.2	75.4	**75.6**
English L20	69.3±0.4	71.2±0.5	71.9±0.4	76.4±0.4	76.0±0.7	75.2±0.6	77.2±0.4	**77.8**±0.5
France L20	69.3±0.4	71.2±0.5	71.4±0.5	76.8±0.4	76.8±0.8	75.3±0.5	77.4±0.5	**78.1**±0.4
German L20	69.3±0.4	71.2±0.5	71.5±0.4	77.1±0.5	76.8±0.7	74.3±0.6	77.3±0.5	**78.4**±0.4
Italian L20	69.3±0.4	71.2±0.5	71.8±0.4	76.9±0.4	76.6±0.7	75.3±0.6	77.3±0.5	**78.1**±0.4
Avg	69.3	71.2	71.7	76.8	76.6	75.0	77.3	**78.1**

Table 7.7: Accuracy (%) with standard error on ImageCLEF dataset from ResNet-50 → VGG-16.

Methods	1NN	SVMt	DAMA	SHFA	G-JDA	CDLS	DACoM	KHDA
I→I	66.7±1.4	76.0±0.6	67.4±1.4	74.4±0.9	79.2±0.5	79.2±0.6	71.3±1.2	**80.0**±0.7
I→P	68.8±1.0	75.5±0.8	68.3±1.3	74.7±0.9	79.2±0.5	79.9±0.1	70.9±0.8	**80.8**±0.2
I→B	68.0±1.1	76.2±0.9	68.4±1.2	74.9±0.9	79.4±0.5	79.2±0.7	69.6±1.1	**80.3**±0.6
I→C	68.8±0.9	75.8±0.8	68.7±1.2	75.4±0.9	79.0±0.4	78.6±0.7	71.6±0.8	**80.0**±0.6
P→I	66.7±1.4	76.0±0.6	68.4±1.2	74.4±0.9	79.2±0.3	79.5±0.4	71.0±1.2	**80.0**±0.7
P→P	68.8±1.0	75.5±0.8	68.6±1.2	74.7±0.9	79.4±0.5	79.9±0.2	70.7±0.9	**80.0**±0.9
P→B	68.0±1.1	76.2±0.6	68.7±1.2	75.0±0.9	79.8±0.5	79.2±0.6	69.8±1.1	**80.4**±0.6
P→C	68.8±0.9	75.8±0.8	68.5±1.2	76.0±0.8	79.4±0.5	78.7±0.7	71.4±0.9	**80.4**±0.5
B→I	66.7±1.4	76.0±0.6	68.6±1.2	75.2±0.9	79.4±0.5	79.2±0.6	70.8±1.3	**80.0**±0.8
B→P	68.8±1.0	75.5±0.8	68.6±1.2	75.8±0.8	79.2±0.5	79.4±0.2	70.6±0.8	**80.4**±0.5
B→B	68.0±1.1	76.2±0.6	68.7±1.2	75.0±0.8	79.3±0.5	79.2±0.6	69.9±1.1	**80.2**±0.6
B→C	69.9±0.9	75.8±0.8	68.6±1.2	74.6±0.8	79.3±0.5	78.2±0.7	71.4±0.8	**80.2**±0.6
C→I	66.7±1.4	76.0±0.6	68.6±1.2	75.1±0.7	79.1±0.6	79.1±0.6	71.5±1.2	**80.0**±0.7
C→P	68.8±1.0	75.5±0.8	68.6±1.2	75.7±0.8	79.2±0.6	79.7±0.1	70.8±0.8	**80.9**±0.2
C→B	68.0±1.1	76.2±0.6	68.6±1.2	75.0±0.8	79.8±0.2	79.4±0.3	69.5±1.1	**80.2**±0.6
C→C	68.8±0.9	75.8±0.8	68.7±1.2	74.7±1.0	79.0±0.8	78.5±0.7	71.8±0.7	**80.1**±0.5
Avg	68.1	75.9	68.5	75.0	79.3	79.2	70.8	**80.2**

Table 7.8: Accuracy (%) with standard error on ImageCLEF dataset from VGG-16 → ResNet-50.

Methods	1NN	SVMt	DAMA	SHFA	G-JDA	CDLS	DACoM	KHDA
I→I	67.0±1.0	73.9±0.6	67.1±1.0	73.1±0.8	76.6±0.6	**79.8**±0.2	71.2±0.8	79.1±0.7
I→P	66.4±0.9	73.9±0.5	67.1±1.0	73.3±0.8	76.0±0.9	79.6±0.3	70.2±0.7	**80.0**±0.1
I→B	66.7±0.9	73.4±0.9	67.2±1.0	71.1±1.2	75.6±0.6	**79.6**±0.1	70.5±0.3	79.5±0.5
I→C	67.9±1.0	73.7±0.9	67.4±1.0	71.7±1.3	76.5±0.6	79.0±0.7	71.3±0.8	**79.6**±0.5
P→I	67.0±1.0	73.9±0.6	67.3±1.0	72.9±0.9	76.0±1.0	77.5±0.6	71.1±0.7	**79.0**±0.7
P→P	66.4±0.9	73.9±0.5	67.3±1.0	73.2±1.0	76.7±0.5	78.4±0.8	70.0±0.6	**80.0**±0.1
P→B	66.7±0.9	73.4±0.9	67.3±1.0	71.5±0.9	76.2±0.6	78.7±0.3	70.4±0.8	**79.5**±0.5
P→C	67.9±1.0	73.7±0.9	67.3±1.0	71.9±1.1	76.7±0.6	78.0±0.7	70.8±0.6	**79.6**±0.5
B→I	67.0±1.0	73.9±0.6	67.4±1.0	72.4±1.1	75.7±1.0	78.6±0.3	71.3±0.8	**79.1**±0.7
B→P	66.4±0.9	73.9±0.5	67.4±1.0	72.8±0.8	77.0±0.5	78.7±0.5	70.4±1.0	**80.0**±0.1
B→B	66.7±0.9	73.4±0.9	67.3±1.0	72.1±0.8	76.1±0.9	78.3±0.5	70.8±0.7	**79.5**±0.5
B→C	67.9±1.0	73.7±0.9	67.3±1.0	71.6±1.0	76.0±0.8	77.9±0.8	71.3±0.7	**79.6**±0.5
C→I	67.0±1.0	73.9±0.6	67.4±1.0	72.1±0.9	76.7±0.6	78.6±0.4	70.4±0.8	**79.1**±0.7
C→P	66.4±0.9	73.9±0.5	67.3±1.0	72.7±0.8	76.8±0.7	79.0±0.3	70.6±0.7	**80.1**±0.1
C→B	66.7±0.9	73.4±0.9	67.3±1.0	72.0±0.8	75.8±0.9	**79.3**±0.1	70.7±0.8	**79.3**±0.5
C→C	67.9±1.0	73.7±0.9	67.4±1.0	71.7±1.0	76.3±0.8	78.3±0.7	71.2±0.8	**79.3**±0.5
Avg	67.0	73.7	67.3	72.1	76.3	78.7	70.8	**79.5**

Table 7.9: Accuracy (%) of ablation study of KHDA on Office+Caltech256, Wikipedia and Retuters-21578 datasets.

Dataset	w/o D	w/o M	$\rho = 0$	$\sigma = 0$	$\boldsymbol{h}^*$	$\boldsymbol{h}$	$\tilde{\boldsymbol{h}}$	w/o $\boldsymbol{h}^*$	w/o $\boldsymbol{h}$	w/o $\tilde{\boldsymbol{h}}$	KHDA ($\boldsymbol{f}$)
Office+Caltech256	90.3	90.2	60.0	62.7	87.2	**93.1**	91.9	92.4	91.7	92.4	**93.0**
Wikipedia	77.9	77.3	75.6	73.6	76.2	**79.2**	**79.1**	78.8	79.0	78.8	**79.2**
Retuters-21578	64.9	67.8	68.3	50.6	57.2	**73.3**	61.4	72.1	61.4	72.3	**73.4**
Avg	77.7	78.4	68.0	62.3	73.5	**81.8**	77.4	81.1	77.3	81.2	**81.8**

Chapter 8

Conclusions and Future Directions

This chapter concludes the entire thesis and provides some further research directions for transfer learning theory and algorithms.

8.1 Conclusions

The research of transfer learning has presented how to transfer knowledge from a source domain to a given target domain, however, most existing researches need additional assumptions to ensure the effectiveness of their algorithms. Two basic assumptions in most transfer learning algorithms are that

1. the source and target label sets are same, i.e., $\mathcal{Y}_s = \mathcal{Y}_t$.

2. the source and target feature spaces are same, i.e., $\mathcal{X}_s = \mathcal{X}_t$;

Under these two assumptions, transfer learning is called homogeneous domain adaptation. To relax the first assumption, this thesis shows how to transfer knowledge from different label sets that the target label set contains the source label set. To relax the second assumption, this thesis presents how to transfer knowledge from heterogeneous feature spaces, i.e., heterogeneous domain adaptation.

To solve the above-mentioned challenges, this thesis proposes three research questions and corresponding research objectives.

1. *To understand how to transfer knowledge from source domain to target domain under the simplest setting (i.e., $\mathcal{Y}_s = \mathcal{Y}_t$ and $\mathcal{X}_s = \mathcal{X}_t$), chapter 3 presents a novel algorithm to address unsupervised homogeneous domain adaptation, and chapter 4 considers a new problem setting for unsupervised homogeneous domain adaptation.* (to address objective 1)

- Based on the perspectives of geometric correlation and fuzzy the-

ory, we propose the first algorithm Manifold Fuzzy Clustering. Manifold Fuzzy Clustering (MFC) jointly explores the geometric structure between the source and target domains. After leveraging the geometric information between domains, MFC compresses the geometric relationship into a membership matrix (fuzzy matrix). With the membership matrix which implies the transfer knowledge between domains, MFC classifies the target samples accurately.

• Chapter 4 proposes a new problem setting for unsupervised homogeneous domain adaptation, called source domain reconstruction (SDR), which constructs a new source domain by the given source domain, target domain and UHoDA algorithm. To explore the feasibility of SDR, we analyze it theoretically. Further, we propose a novel algorithm with utilizing MixUp technique. Nine famous UHoDA algorithms are selected to evaluate the ability of SDR algorithm in reconstructing source sequence and enhance the effectiveness of UHoDA.

2. *To study how to transfer knowledge in the open world (relax assumption 1), i.e., $\mathcal{Y}_s \subset \mathcal{Y}_t$, we investigate the unsupervised open-set domain adaptation. A novel theory for open-set domain adaptation is proposed. Based on our open-set learning bound, a novel UOSDA algorithm is proposed in chapter 5.* (to address objective 2)

To explore the generalization ability and the feasibility of open-set domain adaptation, we establish the open-set domain adaptation learning bound. The bound shows that the risk of target domain can be controlled by three main terms: the source risk, the distribution discrepancy and the open-set difference. Specially, the open-set difference is deeply related to the risk of unknown classes (i.e., class from $\mathcal{Y}_t - \mathcal{Y}_s$). We discover that we can recognize the unknown samples (i.e., samples with labels from $\mathcal{Y}_t - \mathcal{Y}_s$) by minimizing the open-set difference. With the open-set difference as the regularization, we design a novel algorithm Distribution Alignment with Open Difference (DAOD) to address the unsupervised open-set domain adaptation.

3. As an application of our open-set domain adaptation theory, we explore the PAC-learning theory for open-set learning, where there exist test samples from the classes that are unseen during training. We make a bold attempt to study open-set learning by proving its generalization error, i.e., given training samples with size n, the estimation error will get close to order $O_p(1/\sqrt{n})$. This is the first study to provide a generalization bound for open-set learning, which we do by theoretically investigating the risk of the target classifier on unknown classes. According to our theory, a novel algorithm, called auxiliary open-set risk is proposed in chapter 6.

4. *To study how to transfer knowledge between heterogeneous domains (relax assumption 2), i.e., $\mathcal{X}_s \neq \mathcal{X}_t$, we investigate the semi-supervised heterogeneous domain adaptation. The first heterogeneous domain adaptation theory is investigated. To bring the theory into reality, a novel semi-supervised heterogeneous domain adaptation algorithm is proposed in chapter 7.* (to address objective 3)

To explain why the labeled source samples and labeled target samples help to reduce the need for unlabeled target samples, we introduce the concepts of compatibility, transfer error rates and uniform sample complexity as new tools for estimating the need for labeled target data. Then, using these novel concepts, a novel theory for semi-supervised heterogeneous domain adaptation is proposed. Combining our theory and kernel methods, we design a novel algorithm Kernel Heterogeneous Domain Alignment (KHDA) to address the semi-supervised heterogeneous domain adaptation.

8.2 Future Study

This thesis presents the following directions as future work:

• *Partial Domain Adaptation*

In this thesis, we have investigated the open-set domain adaptation, which assumes that the target label set contains the source label set. However, existing some transfer learning scenarios require that the source label set contains the target label sets. This is known as the partial domain adaptation [91]. How to develop learning theory to explore the feasibility of partial domain adaptation is still an open problem. In the future, we aim to explore the learning theory and the theoretical-guided algorithm for partial domain adaptation.

• *Universal Domain Adaptation*

After addressing the open-set domain adaptation and partial domain adaptation, we are interested in addressing universal domain adaptation [92], which assumes that the source and target label sets both contain unknown classes, i.e., $\mathcal{Y}_s - \mathcal{Y}_t \neq \emptyset$ and $\mathcal{Y}_t - \mathcal{Y}_s \neq \emptyset$. It is clear that the open-set domain adaptation and partial domain adaptation are special cases of universal domain adaptation. The learning theory to explain the feasibility of universal domain adaptation is important and challenging.

• *Universal Domain Adaptation*

After addressing the open-set domain adaptation and partial domain adaptation, we are interested in addressing universal domain adaptation [92], which assumes that the source and target label sets both contain unknown classes, i.e., $\mathcal{Y}_s - \mathcal{Y}_t \neq \emptyset$ and $\mathcal{Y}_t - \mathcal{Y}_s \neq \emptyset$. It is clear that the open-set domain adaptation and partial domain adaptation are special cases of universal domain adaptation. The learning theory to explain the feasibility of universal domain adaptation is important and challenging.

• *Remaining Problems in Open-Set Learning Theory*

We list several interesting and important problems for open-set learning theory as follows.

1. How to construct weaker assumption to replace assumption 3 for achieving similar results?

2. Without assumption 3, what will happen?

3. Is it possible for OSL to achieve agnostic PAC learnability and achieve fast learning rate $O_p(1/n^a)$, for $a > 0.5$?

4. Is it possible to construct OSL theory by stability theory [184]?

• *Unsupervised Heterogeneous Domain Adaptation*

In this thesis, the semi-supervised heterogeneous domain adaptation has been studied well. To further understand how to transfer between two heterogeneous domains, the unsupervised heterogeneous domain adaptation is a challenging task. Only few works [153] have investigated this task. These works only focus on the binary classification setting. How to address the unsupervised heterogeneous domain adaptation with multi-classification task is extremely challenging.

Appendix A

Necessary Lemmas

Before proving main theoretical results introduced in this thesis, some necessary lemmas are introduced in this appendix.

Definition 28 (**Rademacher Complexity**). *Let $\mathcal{F}$ be a class of real-valued functions defined in a space $\mathcal{Z}$. Given sample $S = \{\boldsymbol{z}_1, ..., \boldsymbol{z}_n\} \in \mathcal{Z}$, then the Empirical Rademacher Complexity of $\mathcal{F}$ with respect to the sample S is*

$$\widehat{\Re}_S(\mathcal{F}) = \mathbb{E}_\sigma[\sup_{f\in\mathcal{F}} \frac{1}{n}\sum_{i=1}^{n} \sigma_i f(\boldsymbol{z}_i)], \tag{A.0.1}$$

where $\sigma = (\sigma_1, ..., \sigma_n)$ are Rademacher variables, with σ_is independent uniform random variables taking values in $-1, +1$.

Lemma 23. ***(Theorem 26.5 in [57].)*** *Given a space $\mathcal{Z}$, a function $l : R \times \mathcal{Z} \to \mathbb{R}_+$ and a hypothesis set $\mathcal{H} \subset \{f : \mathcal{Z} \to R\}$, let*

$$\mathcal{F} = l \circ \mathcal{H} = \{l(f(\boldsymbol{z}), \boldsymbol{z}) : f \in \mathcal{H}\},$$

where $l \leq B$. Then for a distribution Q on space $\mathcal{Z}$, samples $S = \{\boldsymbol{z}_1, ..., \boldsymbol{z}_n\} \sim Q$ i.i.d, we have with probability of at least $1 - \delta > 0$, for all $f \in \mathcal{F}$:

$$\widehat{R}_S(f) - R_Q(f) \leq 2\mathbb{E}_{S\sim Q^n}\widehat{\Re}_S(\mathcal{F}) + B\sqrt{\frac{2\log(2/\delta)}{n}}, \tag{A.0.2}$$

where $R_Q(f) = \int_{\mathcal{Z}} l(f(\boldsymbol{z}), \boldsymbol{z})\mathrm{d}Q(\boldsymbol{z})$ and $\widehat{R}_S(f) = \frac{1}{n}\sum_{i=1}^{n} l(f(\boldsymbol{z}_i), \boldsymbol{z}_i)$.

Using the same technique of Lemma 23, we can also have: with probability of at least $1 - \delta > 0$, for all $f \in \mathcal{F}$:

$$R_Q(f) - \widehat{R}_S(f) \leq 2\mathbb{E}_{S\sim Q^n}\widehat{\Re}_S(\mathcal{F}) + B\sqrt{\frac{2\log(2/\delta)}{n}},$$

Next we introduce the Natarajan dimension [57], which is a generalization of the VC dimension to classes of multi-class predictors.

Definition 29 (Shattering [57]). *Given a feature space $\mathcal{X}$, we say that a set $U \subset \mathcal{X}$ is shattered by $\mathcal{H}$ if there exist two functions $\boldsymbol{h}_0, \boldsymbol{h}_1 : U \to \mathcal{Y}$, such that*

- *For every $\mathbf{x} \in U$, $\boldsymbol{h}_0(\mathbf{x}) \neq \boldsymbol{h}_1(\mathbf{x})$.*
- *For every $V \subset U$, there exists a function $\boldsymbol{h} \in \mathcal{H}$ such that $\forall \mathbf{x} \in V, \boldsymbol{h}(\mathbf{x}) = \boldsymbol{h}_0(\mathbf{x})$ and $\forall \mathbf{x} \in U \backslash V, \boldsymbol{h}(\mathbf{x}) = \boldsymbol{h}_1(\mathbf{x})$.*

Hence, we can define the Natarajan dimension as follows:

Definition 30 (Natarajan Dimension [57]). *The Natarajan dimension of $\mathcal{H}$, denoted Ndim($\mathcal{H}$), is the maximal size of a shattered set $U \subset X$.*

It is not hard to see that in the case that there are exactly two classes, $\text{Ndim}(\mathcal{H}) = \text{VCdim}(\mathcal{H})$. Therefore, the Natarajan dimension generalizes the VC dimension.

Lemma 24. *Assume that $|\mathcal{Y}| = K$, the loss ℓ is upper bounded by B and the hypothesis space $\mathcal{H}$ has Natarajan dimension d. Given labeled samples $\mathcal{S} \sim P_{XY}$ with size n, there exists a uniform constant $C > 0$ depending on K, B such that for any $\epsilon > 0$, if*

$$n \geq C \frac{d \log(d/\epsilon) + \log(1/\delta)}{\epsilon^2},$$

then with a probability of at least $1 - \delta > 0$, for any $\boldsymbol{h} \in \mathcal{H}$, $\boldsymbol{T}_s \in \mathcal{F}_s$, we have

$$|R_P(\boldsymbol{h}) - \widehat{R}_{\mathcal{S}}(\boldsymbol{h})| \leq \epsilon.$$

Proof. We prove that with a probability of at least $1 - 2\delta > 0$, for all $\boldsymbol{h} \in \mathcal{H}$:

$$|R_P(\boldsymbol{h}) - \widehat{R}_{\mathcal{S}}(\boldsymbol{h})| \leq 2B \sqrt{\frac{8d \log n + 16d \log(K) + 2 \log(2/\delta)}{n}}.$$

Let the samples be $\mathcal{S} = \{(\mathbf{x}^1, \mathbf{y}^1), ..., (\mathbf{x}^n, \mathbf{y}^n)\}$. Recall that the Natarajan

lemma (Lemma 29.4 of [57]) tells us that

$$|\{\boldsymbol{h}(\mathbf{x}^1), ..., \boldsymbol{h}(\mathbf{x}^n) | \boldsymbol{h} \in \mathcal{H}\}| \leq (n)^d K^{2d}.$$

Denote $A = \{(\ell(\boldsymbol{h}(\mathbf{x}^1), \mathbf{y}^1), ..., \ell(\boldsymbol{h}(\mathbf{x}^n), \mathbf{y}^n) | \boldsymbol{h} \in \mathcal{H}\}$. This clearly implies that

$$|A| \leq |\{\boldsymbol{h}(\mathbf{x}^1), ..., \boldsymbol{h}(\mathbf{x}^n) | \boldsymbol{h} \in \mathcal{H}\}| \leq (n)^d K^{2d}.$$

Combining this with Lemma 26.8 of [57], we obtain the following bound:

$$\frac{1}{n}\mathbb{E}_\sigma[\sup_{\mathbf{a}\in A}\sum_{i=1}^{n}\sigma_i a_i] \leq B\sqrt{\frac{2d\log n + 4d\log(K)}{n}},$$

where a_i is the i-th coordinate value of $\mathbf{a}$ and $\sigma = (\sigma_1, ..., \sigma_n)$ are Rademacher variables, with σ_is independent uniform random variables taking values in $-1, +1$.

Let $\mathcal{F}$ be

$$\{\ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}) | \boldsymbol{h} \in \mathcal{H}, (\mathbf{x}, \mathbf{y}) \in \mathcal{X} \times \mathcal{Y}\},$$

then

$$\widehat{\Re}_{\mathcal{S}}(\mathcal{F}) = \frac{1}{n}\mathbb{E}_\sigma[\sup_{\mathbf{a}\in A}\sum_{i=1}^{n}\sigma_i a_i] \leq B\sqrt{\frac{2d\log n + 4d\log(K)}{n}}.$$

Following Lemma 23, we have with a probability of at least $1 - 2\delta > 0$, for all $\boldsymbol{h} \in \mathcal{H}$:

$$\begin{aligned} |R_P(\boldsymbol{h}) - \widehat{R}_{\mathcal{S}}(\boldsymbol{h})| &\leq 2B\sqrt{\frac{2d\log n + 4d\log(K)}{n}} + B\sqrt{\frac{2\log(2/\delta)}{n}} \\ &\leq 2B\sqrt{\frac{8d\log n + 16d\log(K) + 2\log(2/\delta)}{n}}. \end{aligned}$$

Using Lemma A.2 of [57], we prove the result. □

Lemma 25. *[172, 205]. Assume the feature space $\mathcal{X}$ is compact. Let the RKHS $\mathcal{H}_K$ be the Hilbert space with Gaussian kernel. Suppose that the real density $p/q \in \mathcal{H}_k$ and set the regularization parameter $\lambda = \lambda_{n,m}$*

in KuLSIF *such that*

$$\lim_{n,m\to 0} \lambda_{n,m} = 0, \quad \lambda_{n,m}^{-1} = O(\min\{n,m\}^{1-\delta}),$$

where $0 < \delta < 1$ *is any constant, then*

$$\sqrt{\int_{\mathcal{X}} (\widehat{w}(\mathbf{x}) - r(\mathbf{x}))^2 \mathrm{d}U(\mathbf{x})} = O_p(\lambda_{n,m}^{\frac{1}{2}}),$$

and

$$\|\widehat{w}\|_{\mathcal{H}_k} = O_p(1),$$

where $\widehat{w}$ *is the solution of* KuLSIF.

Proof. The result

$$\sqrt{\int_{\mathcal{X}} (\widehat{w}(\mathbf{x}) - r(\mathbf{x}))^2 \mathrm{d}U(\mathbf{x})} = O_p(\lambda_{n,m}^{\frac{1}{2}}),$$

can be found in Theorem 1 of [205] and Theorem 2 of [172].
The result

$$\|\widehat{w}\|_{\mathcal{H}_k} = O_p(1)$$

can be found in the proving process (pages 27-28) of Theorem 1 of [205] and the proving process (pages 354-365) of Theorem 2 of [172]. □

Appendix B

Proofs for Chapter 5

This appendix introduces how to complete the proofs of the main theoretical results in chapter 5.

Proposition 1. *If $\ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y})$ is a $P_{X_tY_t}$-measurable function, then*

$$R_t(\boldsymbol{h}) = (1 - \pi_t^{K+1})R_t^*(\boldsymbol{h}) + \pi_t^{K+1}R_t^{K+1}(\boldsymbol{h}).$$

Proof for Proposition 1. First consider $\pi_t^{K+1}\mathrm{d}P_{X_t|\mathbf{y}_{K+1}}(B)$.

$$\pi_t^{K+1}\mathrm{d}P_{X_t|\mathbf{y}_{K+1}}(\mathbf{x}) = P_{Y_t}(\mathbf{y}_{K+1})\mathrm{d}P_{X_t|\mathbf{y}_{K+1}}(\mathbf{x}) = \mathrm{d}P_{X_tY_t}(\mathbf{x}, \mathbf{y}_{K+1}).$$

Hence, $\pi_t^{K+1}\mathrm{d}P_{X_t|\mathbf{y}_{K+1}}(\mathbf{x}) = \mathrm{d}P_{X_tY_t}(\mathbf{x}, \mathbf{y}_{K+1})$.
Now consider $(1 - \pi_t^{K+1})R_t^*(\boldsymbol{h}) + \pi_t^{K+1}R_t^{K+1}(\boldsymbol{h})$,

$$\begin{aligned}
&(1 - \pi_t^{K+1})R_t^*(\boldsymbol{h}) + \pi_t^{K+1}R_t^{K+1}(\boldsymbol{h}) \\
=&\int_{\mathcal{X}\times\mathcal{Y}_s} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y})\mathrm{d}P_{X_tY_t}(\mathbf{x}, \mathbf{y}) + \pi_t^{K+1}\int_{\mathcal{X}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1})\mathrm{d}P_{X_t|\mathbf{y}_{K+1}}(\mathbf{x}) \\
=&\sum_{\mathbf{y}\in\mathcal{Y}_s}\int_{\mathcal{X}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y})\mathrm{d}P_{X_tY_t}(\mathbf{x}, \mathbf{y}) + \pi_t^{K+1}\int_{\mathcal{X}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1})\mathrm{d}P_{X_t|\mathbf{y}_{K+1}}(\mathbf{x}) \\
=&\sum_{\mathbf{y}\in\mathcal{Y}_s}\int_{\mathcal{X}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y})\mathrm{d}P_{X_tY_t}(\mathbf{x}, \mathbf{y}) + \int_{\mathcal{X}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1})\mathrm{d}P_{X_tY_t}(\mathbf{x}, \mathbf{y}_{K+1}) \\
=&\sum_{\mathbf{y}\in\mathcal{Y}_t}\int_{\mathcal{X}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y})\mathrm{d}P_{X_tY_t}(\mathbf{x}, \mathbf{y}) = \int_{\mathcal{X}\times\mathcal{Y}_t} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y})\mathrm{d}P_{X_tY_t}(\mathbf{x}, \mathbf{y}) \\
=&R_t(\boldsymbol{h}).
\end{aligned}$$

□

Next we provide a proof for Theorem 5.

B.1 Proof of Theorem 5

We firstly present the main idea of the proof. According to Proposition 1, we have that $R_t(\boldsymbol{h}) = (1-\pi_t^{K+1})R_t^*(\boldsymbol{h}) + \pi_t^{K+1}R_t^{K+1}(\boldsymbol{h})$, then

$$\frac{R_t(\boldsymbol{h})}{1-\pi_t^{K+1}} - R_s(\boldsymbol{h}) = R_t^*(\boldsymbol{h}) - R_s(\boldsymbol{h}) + \frac{\pi_t^{K+1}}{1-\pi_t^{K+1}}R_t^{K+1}(\boldsymbol{h}), \tag{B.1.1}$$

thus we separate the proof into two main steps. For the first step, we mainly consider that $R_t^*(\boldsymbol{h}) - R_s(\boldsymbol{h})$. For the second step, we investigate $R_t^{K+1}(\boldsymbol{h})$.

If we can prove that

$$R_t^*(\boldsymbol{h}) - R_s(\boldsymbol{h}) \leq \Lambda + d_{\mathcal{H}}^{\ell}(P_{X_t|\mathcal{Y}_s}, P_{X_s}), \tag{B.1.2}$$

and

$$\frac{\pi_t^{K+1}}{1-\pi_t^{K+1}}R_t^{K+1}(\boldsymbol{h}) \leq d_{\mathcal{H}}^{\ell}(P_{X_t|\mathcal{Y}_s}, P_{X_s}) + \frac{R_t^{u,K+1}(\boldsymbol{h})}{1-\pi_t^{K+1}} - R_t^{u,K+1}(\boldsymbol{h}), \tag{B.1.3}$$

then combining (B.1.2), (B.1.3) with (B.1.1), we have

$$\frac{R_t(\boldsymbol{h})}{1-\pi_t^{K+1}} \leq R_s(\boldsymbol{h}) + 2d_{\mathcal{H}}^{\ell}(P_{X_t|\mathcal{Y}_s}, P_{X_s}) + \Lambda + \Delta_o.$$

Step 1. We claim that $R_t^*(\boldsymbol{h}) - R_s(\boldsymbol{h}) \leq \Lambda + d_{\mathcal{H}}^{\ell}(P_{X_t|\mathcal{Y}_s}, P_{X_s})$.

Firstly, we note that

$$\begin{aligned}
&R_t^*(\boldsymbol{h}) - R_s(\boldsymbol{h}) \\
=&\int_{\mathcal{X}\times\mathcal{Y}_t} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}) \mathrm{d}P_{X_tY_t|\mathcal{Y}_s}(\mathbf{x}, \mathbf{y}) - \int_{\mathcal{X}\times\mathcal{Y}_s} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}) \mathrm{d}P_{X_sY_s}(\mathbf{x}, \mathbf{y}) \\
\underset{\text{inequality1}}{\leq}& R_t^*(\tilde{\boldsymbol{h}}) + \int_{\mathcal{X}\times\mathcal{Y}_t} \ell(\boldsymbol{h}(\mathbf{x}), \tilde{\boldsymbol{h}}(\mathbf{x})) \mathrm{d}P_{X_tY_t|\mathcal{Y}_s}(\mathbf{x}, \mathbf{y}) \\
&+ R_s(\tilde{\boldsymbol{h}}) - \int_{\mathcal{X}\times\mathcal{Y}_s} \ell(\boldsymbol{h}(\mathbf{x}), \tilde{\boldsymbol{h}}(\mathbf{x})) \mathrm{d}P_{X_sY_s}(\mathbf{x}, \mathbf{y}),
\end{aligned} \tag{B.1.4}$$

where $\tilde{\boldsymbol{h}}$ is any scoring function in $\mathcal{H}$. In inequality 1, we have used following triangle inequality: $\left|\ell(\boldsymbol{h}(\mathbf{x}),\mathbf{y})-\ell(\tilde{\boldsymbol{h}}(\mathbf{x}),\mathbf{y})\right|\leq\ell(\boldsymbol{h}(\mathbf{x}),\tilde{\boldsymbol{h}}(\mathbf{x}))$.

Then, according to the definition of $P_{X_tY_t|\mathcal{Y}_s}$, we can check that

$$\int_{\mathcal{X}\times\mathcal{Y}_t}\ell(\boldsymbol{h}(\mathbf{x}),\tilde{\boldsymbol{h}}(\mathbf{x}))\mathrm{d}P_{X_tY_t|\mathcal{Y}_s}(\mathbf{x},\mathbf{y})=\int_{\mathcal{X}}\ell(\boldsymbol{h}(\mathbf{x}),\tilde{\boldsymbol{h}}(\mathbf{x}))\mathrm{d}P_{X_t|\mathcal{Y}_s}(\mathbf{x}). \tag{B.1.5}$$

By Fubini's theorem, it is also easy to check that

$$\int_{\mathcal{X}\times\mathcal{Y}_s}\ell(\boldsymbol{h}(\mathbf{x}),\tilde{\boldsymbol{h}}(\mathbf{x}))\mathrm{d}P_{X_sY_s}(\mathbf{x},\mathbf{y})=\int_{\mathcal{X}}\ell(\boldsymbol{h}(\mathbf{x}),\tilde{\boldsymbol{h}}(\mathbf{x}))\mathrm{d}P_{X_s}(\mathbf{x}). \tag{B.1.6}$$

Based on (B.1.4), (B.1.5), (B.1.6) and the definition of the $\mathcal{H}\Delta\mathcal{H}$-divergence, we have

$$\begin{aligned}
&R_t^*(\boldsymbol{h})-R_s(\boldsymbol{h})\\
\leq\ &R_t^*(\tilde{\boldsymbol{h}})+R_s(\tilde{\boldsymbol{h}})+\left|\int_{\mathcal{X}}\ell(\boldsymbol{h}(\mathbf{x}),\tilde{\boldsymbol{h}}(\mathbf{x}))\mathrm{d}P_{X_t|\mathcal{Y}_s}(\mathbf{x})-\int_{\mathcal{X}}\ell(\boldsymbol{h}(\mathbf{x}),\tilde{\boldsymbol{h}}(\mathbf{x}))\mathrm{d}P_{X_s}(\mathbf{x})\right|\\
\leq\ &R_t^*(\tilde{\boldsymbol{h}})+R_s(\tilde{\boldsymbol{h}})+d_{\mathcal{H}}^{\ell}(P_{X_t|\mathcal{Y}_s},P_{X_s}).
\end{aligned}$$

Hence,

$$\begin{aligned}
&R_t^*(\boldsymbol{h})-R_s(\boldsymbol{h})\\
\leq&\min_{\tilde{h}\in\mathcal{H}}\left(R_t^*(\tilde{\boldsymbol{h}})+R_s(\tilde{\boldsymbol{h}})\right)+d_{\mathcal{H}}^{\ell}(P_{X_t|\mathcal{Y}_s},P_{X_s})\\
=&\Lambda+d_{\mathcal{H}}^{\ell}(P_{X_t|\mathcal{Y}_s},P_{X_s}).
\end{aligned} \tag{B.1.7}$$

Step 2. We claim that $\frac{\pi_t^{K+1}}{1-\pi_t^{K+1}}R_t^{K+1}(\boldsymbol{h})\leq d_{\mathcal{H}}^{\ell}(P_{X_t|\mathcal{Y}_s},P_{X_s})+\frac{R_t^{u,K+1}(\boldsymbol{h})}{1-\pi_t^{K+1}}-R_t^{u,K+1}(\boldsymbol{h})$.

We note that

$$\begin{aligned}
&R_t^{u,K+1}(\boldsymbol{h})-(1-\pi_t^{K+1})R_s^{u,K+1}(\boldsymbol{h})\\
=&\pi_t^{K+1}R_t^{K+1}(\boldsymbol{h})+(1-\pi_t^{K+1})\int_{\mathcal{X}}\ell(\boldsymbol{h}(\mathbf{x}),\mathbf{y}_{K+1})\mathrm{d}P_{X_t|\mathcal{Y}_s}(\mathbf{x})\\
&-(1-\pi_t^{K+1})\int_{\mathcal{X}}\ell(\boldsymbol{h}(\mathbf{x}),\mathbf{y}_{K+1})\mathrm{d}P_{X_s}(\mathbf{x}),
\end{aligned} \tag{B.1.8}$$

this is because the following equation

$$\begin{aligned}
&R_t^{u,K+1}(\boldsymbol{h}) = \int_{\mathcal{X}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}) \mathrm{d}P_{X_t}(\mathbf{x}) \\
=&\pi_t^{K+1} \int_{\mathcal{X}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}) \mathrm{d}P_{X_t|\mathbf{y}_{K+1}}(\mathbf{x}) \\
+&(1-\pi_t^{K+1}) \int_{\mathcal{X}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}) \mathrm{d}P_{X_t|\mathcal{Y}_s}(\mathbf{x}) \\
=&\pi_t^{K+1} R_{K+1}^t(\boldsymbol{h}) + (1-\pi_t^{K+1}) \int_{\mathcal{X}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}) \mathrm{d}P_{X_t|\mathcal{Y}_s}(\mathbf{x}).
\end{aligned} \tag{B.1.9}$$

According to the definition of the discrepancy distance and the condition that the constant vector value function $\boldsymbol{g} = \mathbf{y}_{K+1} \in \mathcal{H}$, we have

$$\begin{aligned}
&\left| \int_{\mathcal{X}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}) \mathrm{d}P_{X_t|\mathcal{Y}_s}(\mathbf{x}) - \int_{\mathcal{X}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}) \mathrm{d}P_{X_s}(\mathbf{x}) \right| \\
=&\left| \int_{\mathcal{X}} \ell(\boldsymbol{h}(\mathbf{x}), \boldsymbol{g}(\mathbf{x})) \mathrm{d}P_{X_t|\mathcal{Y}_s}(\mathbf{x}) - \int_{\mathcal{X}} \ell(\boldsymbol{h}(\mathbf{x}), \boldsymbol{g}(\mathbf{x})) \mathrm{d}P_{X_s}(\mathbf{x}) \right| \\
\leq& d_{\mathcal{H}}^{\ell}(P_{X_t|\mathcal{Y}_s}, P_{X_s}).
\end{aligned} \tag{B.1.10}$$

Combining (B.1.8) and (B.1.10), we show that

$$\begin{aligned}
&\pi_t^{K+1} R_t^{K+1}(\boldsymbol{h}) \\
\leq&(1-\pi_t^{K+1}) d_{\mathcal{H}}^{\ell}(P_{X_t|\mathcal{Y}_s}, P_{X_s}) + R_t^{u,K+1}(\boldsymbol{h}) - (1-\pi_t^{K+1}) R_t^{u,K+1}(\boldsymbol{h}).
\end{aligned}$$

Hence, we have proved $\frac{\pi_t^{K+1}}{1-\pi_t^{K+1}} R_t^{K+1}(\boldsymbol{h}) \leq d_{\mathcal{H}}^{\ell}(P_{X_t|\mathcal{Y}_s}, P_{X_s}) + \frac{R_t^{u,K+1}(\boldsymbol{h})}{1-\pi_t^{K+1}} - R_t^{u,K+1}(\boldsymbol{h})$.

As mentioned above, combining (B.1.2), (B.1.3) with (B.1.1), we have

$$\frac{R_t(\boldsymbol{h})}{1-\pi_t^{K+1}} \leq R_s(\boldsymbol{h}) + 2d_{\mathcal{H}}^{\ell}(P_{X_t|\mathcal{Y}_s}, P_{X_s}) + \Lambda + \Delta_o.$$

The proof has been completed.

B.2 Proof of Corollary 5.1

Step 1. Using Lemma 24, we can prove that with probability of at least $1 - 2\delta > 0$, for all $\boldsymbol{h} \in \mathcal{H}$:

$$|R_s(\boldsymbol{h}) - \widehat{R}_s(\boldsymbol{h})| \leq 2B\sqrt{\frac{8d\log n_s + 16d\log(K+1) + 2\log(2/\delta)}{n_s}}.$$

Step 2. We prove that with probability of at least $1 - \delta > 0$, for all $\boldsymbol{h} \in \mathcal{H}$:

$$-\widehat{R}_s^{u,K+1}(\boldsymbol{h}) \leq -R_s^{u,K+1}(\boldsymbol{h}) + 2B\sqrt{\frac{8d\log n_s + 16d\log(K+1) + 2\log(2/\delta)}{n_s}}.$$

Let the source samples be $\mathcal{S}_X = \{\mathbf{x}_s^1, ..., \mathbf{x}_s^{n_s}\}$. Recall that the Natarajan lemma (Lemma 29.4 of [57]) tells us that if $\text{Ndim}(\mathcal{H})$ is d, then

$$|\{\boldsymbol{h}(\mathbf{x}_s^1), ..., \boldsymbol{h}(\mathbf{x}_s^{n_s}) : \boldsymbol{h} \in \mathcal{H}\}| \leq (n_s)^d (K+1)^{2d}.$$

Denote $A = \{(\ell(\boldsymbol{h}(\mathbf{x}_s^1), \mathbf{y}_{K+1}), ..., \ell(\boldsymbol{h}(\mathbf{x}_s^{n_s}), \mathbf{y}_{K+1})) : \boldsymbol{h} \in \mathcal{H}\}$. This clearly implies that

$$|A| \leq |\{\boldsymbol{h}(\mathbf{x}_s^1), ..., \boldsymbol{h}(\mathbf{x}_s^{n_s}) : \boldsymbol{h} \in \mathcal{H}\}| \leq (n_s)^d (K+1)^{2d}.$$

Combining this with Lemma 26.8 of [57], we obtain the following bound:

$$\frac{1}{n_s}\mathbb{E}_\sigma[\sup_{\mathbf{a}\in A}\sum_{i=1}^{n_s}\sigma_i a_i] \leq \sqrt{\frac{2d\log n_s + 4d\log(K+1)}{n_s}},$$

where a_i is the i-th coordinate value of $\mathbf{a}$ and $\sigma = (\sigma_1, ..., \sigma_n)$ are Rademacher variables, with σ_is independent uniform random variables taking values in $-1, +1$.

Let $\mathcal{F}$ be

$$\{\ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}) : \boldsymbol{h} \in \mathcal{H}, \mathbf{x} \in \mathcal{X}\},$$

then

$$\widehat{\Re}_{\mathcal{S}_X}(\mathcal{F}) = \frac{1}{n_s}\mathbb{E}_\sigma[\sup_{\mathbf{a}\in A}\sum_{i=1}^{n_s}\sigma_i a_i] \leq \sqrt{\frac{2d\log n_s + 4d\log(K+1)}{n_s}}.$$

Following Lemma 23, we have with probability of at least $1-\delta > 0$, for all $\boldsymbol{h} \in \mathcal{H}$:

$$-\widehat{R}_s^{u,K+1}(\boldsymbol{h}) \leq -R_s^{u,K+1}(\boldsymbol{h}) + 2B\sqrt{\frac{8d\log n_s + 16d\log(K+1) + 2\log(2/\delta)}{n_s}}.$$

Step 3. We prove that with probability of at least $1-\delta > 0$, for all $\boldsymbol{h} \in \mathcal{H}$:

$$\widehat{R}_t^{u,K+1}(\boldsymbol{h}) \leq R_t^{u,K+1}(\boldsymbol{h}) + 2B\sqrt{\frac{8d\log n_u + 16d\log(K+1) + 2\log(2/\delta)}{n_u}}.$$

Let thetarget samples be $\mathcal{T}_X = \{\mathbf{x}_t^1, ..., \mathbf{x}_t^{n_u}\}$. Recall that the Natarajan lemma (Lemma 29.4 of [57]) tells us that if $\text{Ndim}(\mathcal{H})$ is d, then

$$|\{\boldsymbol{h}(\mathbf{x}_t^1), ..., \boldsymbol{h}(\mathbf{x}_t^{n_u}) : \boldsymbol{h} \in \mathcal{H}\}| \leq (n_u)^d (K+1)^{2d}.$$

Denote $A = \{(\ell(\boldsymbol{h}(\mathbf{x}_t^1), \mathbf{y}_{K+1}), ..., \ell(\boldsymbol{h}(\mathbf{x}_t^{n_u})), \mathbf{y}_{K+1}) : \boldsymbol{h} \in \mathcal{H}\}$. This clearly implies that

$$|A| \leq |\{\boldsymbol{h}(\mathbf{x}_t^1), ..., \boldsymbol{h}(\mathbf{x}_t^{n_u}) : \boldsymbol{h} \in \mathcal{H}\}| \leq (n_u)^d (K+1)^{2d}.$$

Combining this with Lemma 26.8 of [57], we obtain the following bound:

$$\frac{1}{n_u}\mathbb{E}_\sigma[\sup_{\mathbf{a}\in A}\sum_{i=1}^{n_u}\sigma_i a_i] \leq \sqrt{\frac{2d\log n_u + 4d\log(K+1)}{n_u}},$$

where a_i is the i-th coordinate value of $\mathbf{a}$ and $\sigma = (\sigma_1, ..., \sigma_n)$ are Rademacher variables, with σ_is independent uniform random variables taking values in $-1, +1$.

Let $\mathcal{F}$ be

$$\{\ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}) : \boldsymbol{h} \in \mathcal{H}, \mathbf{x} \in \mathcal{X}\},$$

then

$$\widehat{\Re}_{\mathcal{T}_u}(\mathcal{F}) = \frac{1}{n_u}\mathbb{E}_\sigma[\sup_{\mathbf{a}\in A}\sum_{i=1}^{n_u}\sigma_i a_i] \leq \sqrt{\frac{2d\log n_u + 4d\log(K+1)}{n_u}}.$$

Following Lemma 23, we have with probability of at least $\delta > 0$, for all $\boldsymbol{h} \in \mathcal{H}$:

$$\widehat{R}_t^{u,K+1}(\boldsymbol{h}) \leq R_t^{u,K+1}(\boldsymbol{h}) + 2B\sqrt{\frac{8d\log n_u + 16d\log(K+1) + 2\log(2/\delta)}{n_u}}.$$

Combining Steps 1, 2 and 3 with Theorem 5, we obtain the result.

Appendix C

Proofs for Chapter 6

This appendix introduces how to complete the proofs of the main theoretical results in chapter 6.

Lemma 26. *For any* $\boldsymbol{h} \in \mathcal{H}$,

$$R_Q^\alpha(\boldsymbol{h}) = (1-\alpha)R_{P,k}(\boldsymbol{h}) + \max\{R_Q(\boldsymbol{h}, \mathbf{y}_{K+1}) - (1-\alpha)R_{P,k}(\boldsymbol{h}, \mathbf{y}_{K+1}), 0\},$$

where $\alpha = Q(Y = \mathbf{y}_{K+1})$,

$$R_Q(\boldsymbol{h}, \mathbf{y}_{K+1}) = \int_{\mathcal{X}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}) \mathrm{d}Q_X(\mathbf{x}),$$

and

$$R_{P,k}(\boldsymbol{h}, \mathbf{y}_{K+1}) = \int_{\mathcal{X}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}) \mathrm{d}P_{X|Y\in\mathcal{Y}_k}(\mathbf{x}),$$

Proof. **Step 1.** We claim that $R_Q^\alpha(\boldsymbol{h}) = (1-\alpha)R_{P,k}(\boldsymbol{h}) + \alpha R_{Q,u}(\boldsymbol{h})$. First, it is clear that

$$R_Q^\alpha(\boldsymbol{h}) = (1-\alpha)R_{Q,k}(\boldsymbol{h}) + \alpha R_{Q,u}(\boldsymbol{h}). \tag{C.0.1}$$

Because $Q_{XY|Y\in\mathcal{Y}_k} = P_{XY|Y\in\mathcal{Y}_k}$, hence,

$$\begin{aligned} R_{Q,k}(\boldsymbol{h}) =& \int_{\mathcal{X}\times\mathcal{Y}_k} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}) \mathrm{d}Q_{XY|Y\in\mathcal{Y}_k}(\mathbf{x}, \mathbf{y}) \\ =& \int_{\mathcal{X}\times\mathcal{Y}_k} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}) \mathrm{d}P_{XY|Y\in\mathcal{Y}_k}(\mathbf{x}, \mathbf{y}) \\ =& R_{P,k}(\boldsymbol{h}). \end{aligned} \tag{C.0.2}$$

Combining Eq. (C.0.1) with Eq. (C.0.2), we have that

$$R_Q^\alpha(\boldsymbol{h}) = (1-\alpha)R_{P,k}(\boldsymbol{h}) + \alpha R_{Q,u}(\boldsymbol{h}).$$

Step 2. We claim that $\alpha R_{Q,u}(\boldsymbol{h}) = \max\{R_Q(\boldsymbol{h}, \mathbf{y}_{K+1}) - (1-\alpha)R_{P,k}(\boldsymbol{h}, \mathbf{y}_{K+1}), 0\}$.

First, it is clear that

$$\begin{aligned}
&R_Q(\boldsymbol{h}, \mathbf{y}_{K+1}) \\
=&(1-\alpha)\int_{\mathcal{X}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}) \mathrm{d}Q_{X|Y\in\mathcal{Y}_k} + \alpha \int_{\mathcal{X}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}) \mathrm{d}Q_{X|Y=\mathbf{y}_{K+1}} \\
=&(1-\alpha)\int_{\mathcal{X}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}) \mathrm{d}P_{X|Y\in\mathcal{Y}_k} + \alpha \int_{\mathcal{X}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}) \mathrm{d}Q_{X|Y=\mathbf{y}_{K+1}} \\
=&(1-\alpha)R_{P,k}(\boldsymbol{h}, \mathbf{y}_{K+1}) + \alpha \int_{\mathcal{X}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}) \mathrm{d}Q_{X|Y=\mathbf{y}_{K+1}} \\
=&(1-\alpha)R_{P,k}(\boldsymbol{h}, \mathbf{y}_{K+1}) + \alpha R_{Q,u}(\boldsymbol{h}).
\end{aligned} \tag{C.0.3}$$

Hence,

$$\alpha R_{Q,u}(\boldsymbol{h}) = R_Q(\boldsymbol{h}, \mathbf{y}_{K+1}) - (1-\alpha)R_{P,k}(\boldsymbol{h}, \mathbf{y}_{K+1}).$$

Because $\alpha R_{Q,u}(\boldsymbol{h}) \geq 0$, we obtain that

$$\alpha R_{Q,u}(\boldsymbol{h}) = \max\{R_Q(\boldsymbol{h}, \mathbf{y}_{K+1}) - (1-\alpha)R_{P,k}(\boldsymbol{h}, \mathbf{y}_{K+1}), 0\}.$$

Step 3. Combining the results of Steps 1 and 2, we have that

$$R_Q^\alpha(\boldsymbol{h}) = (1-\alpha)R_{P,k}(\boldsymbol{h}) + \max\{R_Q(\boldsymbol{h}, \mathbf{y}_{K+1}) - (1-\alpha)R_{P,k}(\boldsymbol{h}, \mathbf{y}_{K+1}), 0\}.$$

□

Lemma 27. *Assume that $\mathcal{H} \subset \{\boldsymbol{h} : \mathcal{X} \to \mathcal{Y}\}$ has finite Natarajan dimension and the loss function ℓ has upper bound c, then for any $0 < \delta < 1$,*

$$\begin{aligned}
\sup_{\boldsymbol{h}\in\mathcal{H}} |R_{P,k}(\boldsymbol{h}) - \widehat{R}_S(\boldsymbol{h})| &= cO_p(1/n^{\frac{1-\delta}{2}}), \\
\sup_{\boldsymbol{h}\in\mathcal{H}} |R_{P,k}(\boldsymbol{h}, \mathbf{y}_{K+1}) - \widehat{R}_S(\boldsymbol{h}, \mathbf{y}_{K+1})| &= cO_p(1/n^{\frac{1-\delta}{2}}),
\end{aligned}$$

where

$$\widehat{R}_S(\boldsymbol{h}) = \frac{1}{n}\sum_{(\mathbf{x},\mathbf{y})\in S} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}), \quad \widehat{R}_S(\boldsymbol{h}, \mathbf{y}_{K+1}) = \frac{1}{n}\sum_{(\mathbf{x},\mathbf{y})\in S} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}).$$

Proof. The proof is similar to the proof of Lemma 24. We omit it. □

Lemma 28. *Assume the feature space $\mathcal{X}$ is compact and the loss function has an upper bound c. Let the RKHS $\mathcal{H}_k$ is the Hilbert space with Gaussian kernel. Suppose that the real density $p/q \in \mathcal{H}_k$ and set the regularization parameter $\lambda = \lambda_{n,m}$ in* KuLSIF *such that*

$$\lim_{n,m\to 0} \lambda_{n,m} = 0, \quad \lambda_{n,m}^{-1} = O(\min\{n,m\}^{1-\delta}),$$

where $0 < \delta < 1$ is any constant, then

$$\begin{aligned}&\sup_{\boldsymbol{h}\in\mathcal{H}} |R_Q(\boldsymbol{h}, \mathbf{y}_{K+1}) - \gamma \widehat{R}_T^{\tau,\beta}(\boldsymbol{h}, \mathbf{y}_{K+1})| \\ \leq &\gamma\beta c U(\{\mathbf{x}:\ 0 < r(\mathbf{x}) \leq 2\tau\}) + c\big(\max\{1, \frac{\beta}{\tau}\} + \beta\big) O_p(\lambda_{n,m}^{\frac{1}{2}}),\end{aligned}$$

where

$$\begin{aligned}R_Q(\boldsymbol{h}, \mathbf{y}_{K+1}) &= \int_{\mathcal{X}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}) \mathrm{d}Q_X(\mathbf{x}), \\ \widehat{R}_T^{\tau,\beta}(\boldsymbol{h}, \mathbf{y}_{K+1}) &= \frac{1}{m}\sum_{\mathbf{x}\in T} L_{\tau,\beta}(\widehat{w}(\mathbf{x})) \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}),\end{aligned}$$

here $Q_X = Q_U^{0,\beta}$, and $\widehat{w}$ is the solution of KuLSIF*.*

Proof. **Step 1.** We claim that

$$\sup_{\boldsymbol{h}\in\mathcal{H}} |R_Q(\boldsymbol{h}, \mathbf{y}_{K+1}) - \gamma R_U^{\tau,\beta}(\boldsymbol{h}, \mathbf{y}_{K+1})| \leq \gamma\beta c U(\{\mathbf{x}:\ 0 < r(\mathbf{x}) \leq 2\tau\}),$$

where

$$R_U^{\tau,\beta}(\boldsymbol{h}, \mathbf{y}_{K+1}) = \int_{\mathcal{X}} L_{\tau,\beta}(r(\mathbf{x})) \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}) \mathrm{d}U(\mathbf{x}),$$

here $r(\mathbf{x}) = p(\mathbf{x})/q(\mathbf{x})$.

First, we note that

$$
\begin{aligned}
&\Big| \int_{\mathcal{X}} L_{0,\beta}(r(\mathbf{x}))\ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1})\mathrm{d}U(\mathbf{x}) - \int_{\mathcal{X}} L_{\tau,\beta}(r(\mathbf{x}))\ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1})\mathrm{d}U(\mathbf{x})| \\
\leq &\Big| \int_{\mathcal{X}} L_{0,\beta}(r(\mathbf{x}))\ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1})\mathrm{d}U(\mathbf{x}) - \int_{\mathcal{X}} L_{\tau,\beta}(r(\mathbf{x}))\ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1})\mathrm{d}U(\mathbf{x})| \\
\leq &c \int_{\mathcal{X}} \big|L_{0,\beta}(r(\mathbf{x})) - L_{\tau,\beta}(r(\mathbf{x}))\big|\mathrm{d}U(\mathbf{x}) \\
\leq &c \int_{\{\mathbf{x}:\ 0<r(\mathbf{x})\leq 2\tau\}} \beta \mathrm{d}U(\mathbf{x}) = \beta c U(\{\mathbf{x}:\ 0 < r(\mathbf{x}) \leq 2\tau\}).
\end{aligned}
\tag{C.0.4}
$$

Because $Q_{XY} = Q_U^{0,\beta} P_{Y|X}$, then according to the definition of $Q_U^{0,\beta}$, we know

$$
R_Q(\boldsymbol{h}, \mathbf{y}_{K+1}) = \gamma \int_{\mathcal{X}} L_{0,\beta}(r(\mathbf{x}))\ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1})\mathrm{d}U(\mathbf{x}),
$$

which implies

$$
\sup_{\boldsymbol{h}\in\mathcal{H}} |R_Q(\boldsymbol{h}, \mathbf{y}_{K+1}) - \gamma R_U^{\tau,\beta}(\boldsymbol{h}, \mathbf{y}_{K+1})| \leq \gamma\beta c U(\{\mathbf{x}:\ 0 < r(\mathbf{x}) \leq 2\tau\}).
$$

Step 2. We claim that

$$
\begin{aligned}
&\sup_{\boldsymbol{h}\in\mathcal{H}} |R_U^{\tau,\beta}(\boldsymbol{h}, \mathbf{y}_{K+1}) - \int_{\mathcal{X}} L_{\tau,\beta}(\widehat{w}(\mathbf{x}))\ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1})\mathrm{d}U(\mathbf{x})| \\
\leq &\max\{c, \frac{c\beta}{\tau}\} O_p(\lambda_{n,m}^{\frac{1}{2}}).
\end{aligned}
$$

First, the Lipschitz constant for $L_{\tau,\beta}$ is smaller than $\max\{1, \frac{\beta}{\tau}\}$.

Then,

$$
\begin{aligned}
&\sup_{\boldsymbol{h}\in\mathcal{H}} |R_U^{\tau,\beta}(\boldsymbol{h}, \mathbf{y}_{K+1}) - \int_{\mathcal{X}} L_{\tau,\beta}(\widehat{w}(\mathbf{x}))\ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1})\mathrm{d}U(\mathbf{x})| \\
= &\sup_{\boldsymbol{h}\in\mathcal{H}} | \int_{\mathcal{X}} L_{\tau,\beta}(r(\mathbf{x}))\ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1})\mathrm{d}U(\mathbf{x}) \\
&- \int_{\mathcal{X}} L_{\tau,\beta}(\widehat{w}(\mathbf{x}))\ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1})\mathrm{d}U(\mathbf{x}) \\
\leq &\sup_{\boldsymbol{h}\in\mathcal{H}} \int_{\mathcal{X}} |L_{\tau,\beta}(r(\mathbf{x})) - L_{\tau,\beta}(\widehat{w}(\mathbf{x}))|\ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1})\mathrm{d}U(\mathbf{x})|
\end{aligned}
$$

$$
\begin{aligned}
&\leq \sup_{\boldsymbol{h}\in\mathcal{H}} \sqrt{\int_{\mathcal{X}} |L_{\tau,\beta}(r(\mathbf{x})) - L_{\tau,\beta}(\widehat{w}(\mathbf{x}))|^2 \mathrm{d}U(\mathbf{x})} \sqrt{\int_{\mathcal{X}} \ell^2(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}) \mathrm{d}U(\mathbf{x})} \\
&\leq c \sup_{\boldsymbol{h}\in\mathcal{H}} \sqrt{\int_{\mathcal{X}} |L_{\tau,\beta}(r(\mathbf{x})) - L_{\tau,\beta}(\widehat{w}(\mathbf{x}))|^2 \mathrm{d}U(\mathbf{x})} \\
&\leq \max\{c, \frac{c\beta}{\tau}\} \sup_{\boldsymbol{h}\in\mathcal{H}} \sqrt{\int_{\mathcal{X}} |r(\mathbf{x}) - \widehat{w}(\mathbf{x})|^2 \mathrm{d}U(\mathbf{x})}.
\end{aligned}
$$

Lastly, using Lemma 25,

$$
\begin{aligned}
&\sup_{\boldsymbol{h}\in\mathcal{H}} |R_U^{\tau,\beta}(\boldsymbol{h}, \mathbf{y}_{K+1}) - \int_{\mathcal{X}} L_{\tau,\beta}(\widehat{w}(\mathbf{x})) \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}) \mathrm{d}U(\mathbf{x})| \\
\leq &\max\{c, \frac{c\beta}{\tau}\} O_p(\lambda_{n,m}^{\frac{1}{2}}).
\end{aligned}
$$

Step 3. We claim that

$$
\begin{aligned}
&\sup_{\boldsymbol{h}\in\mathcal{H}} \big|\frac{1}{m} \sum_{\mathbf{x}\in T} L_{\tau,\beta}(\widehat{w}(\mathbf{x})) \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}) - \int_{\mathcal{X}} L_{\tau,\beta}(\widehat{w}(\mathbf{x})) \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}) \mathrm{d}U(\mathbf{x})\big| \\
\leq &c\big(\max\{1, \frac{\beta}{\tau}\} + \beta\big) O_p(\lambda_{n,m}^{\frac{1}{2}}).
\end{aligned}
$$

First, we set $\mathcal{F}_B = \{L_{\tau,\beta}(w)\ell(\boldsymbol{h}, \mathbf{y}_{K+1}) : w \in \mathcal{H}_k, \|w\|_k \leq B, \boldsymbol{h} \in \mathcal{H}\}$. We consider

$$
\sup_{f\in\mathcal{F}_B} \big|\frac{1}{m} \sum_{\mathbf{x}\in T} f(\mathbf{x}) - \int_{\mathcal{X}} f(\mathbf{x}) \mathrm{d}U(\mathbf{x})\big|
$$

Using Lemma 23, it is easy to check that for $1 - 2\delta > 0$, we have

$$
\sup_{f\in\mathcal{F}_B} \big|\frac{1}{m} \sum_{\mathbf{x}\in T} f(\mathbf{x}) - \int_{\mathcal{X}} f(\mathbf{x}) \mathrm{d}U(\mathbf{x})\big| \leq 2\widehat{\Re}_T(\mathcal{F}_B) + 4(B+\beta)c\sqrt{\frac{2\log(4/\delta)}{m}}, \tag{C.0.5}
$$

here we have used $|f| \leq (B+\beta)c$, for any $f \in \mathcal{F}_B$.

Then, we consider $\widehat{\Re}_T(\mathcal{F}_B)$.

$$
\begin{aligned}
&m\widehat{\Re}_T(\mathcal{F}_B) \\
=&\mathbb{E}_\sigma[\sup_{\|w\|_k\leq B, \boldsymbol{h}\in\mathcal{H}} \sum_{i=1}^m \sigma_i L_{\tau,\beta}(w_i) \ell(\boldsymbol{h}_i, \mathbf{y}_{K+1})] \\
=&\mathbb{E}_\sigma[\sup_{\|w\|_k\leq B, \boldsymbol{h}\in\mathcal{H}} \sigma_1 L_{\tau,\beta}(w_1) \ell(\boldsymbol{h}_1, \mathbf{y}_{K+1}) + \sum_{i=2}^m \sigma_i L_{\tau,\beta}(w_i) \ell(\boldsymbol{h}_i, \mathbf{y}_{K+1})]
\end{aligned}
$$

$$
\begin{aligned}
=&\frac{1}{2}\mathbb{E}_{\sigma_2,\ldots,\sigma_m}[\sup_{\|w\|_k\le B,\boldsymbol{h}\in\mathcal{H}} L_{\tau,\beta}(w_1)\ell(\boldsymbol{h}_1,\mathbf{y}_{K+1})+\sum_{i=2}^{m}\sigma_i L_{\tau,\beta}(w_i)\ell(\boldsymbol{h}_i,\mathbf{y}_{K+1})\\
&+\sup_{\|w\|_k\le B,\boldsymbol{h}\in\mathcal{H}} -L_{\tau,\beta}(w_1)\ell(\boldsymbol{h}_1,\mathbf{y}_{K+1})+\sum_{i=2}^{m}\sigma_i L_{\tau,\beta}(w_i)\ell(\boldsymbol{h}_i,\mathbf{y}_{K+1})]\\
=&\frac{1}{2}\mathbb{E}_{\sigma_2,\ldots,\sigma_m}[\sup_{\|w\|_k\le B,\boldsymbol{h}\in\mathcal{H}}\sup_{\|w'\|_k\le B,\boldsymbol{h}'\in\mathcal{H}} L_{\tau,\beta}(w_1)\ell(\boldsymbol{h}_1,\mathbf{y}_{K+1})\\
&+\sum_{i=2}^{m}\sigma_i L_{\tau,\beta}(w_i)\ell(\boldsymbol{h}_i,\mathbf{y}_{K+1})-L_{\tau,\beta}(w_1')\ell(\boldsymbol{h}_1',\mathbf{y}_{K+1})\\
&+\sum_{i=2}^{m}\sigma_i L_{\tau,\beta}(w_i')\ell(\boldsymbol{h}_i',\mathbf{y}_{K+1})]\\
\le&\frac{1}{2}\mathbb{E}_{\sigma_2,\ldots,\sigma_m}[\sup_{\|w\|_k\le B,\boldsymbol{h}\in\mathcal{H}}\sup_{\|w'\|_k\le B,\boldsymbol{h}'\in\mathcal{H}} |L_{\tau,\beta}(w_1)-L_{\tau,\beta}(w_1')|\ell(\boldsymbol{h}_1',\mathbf{y}_{K+1})\\
&+L_{\tau,\beta}(w_1)|\ell(\boldsymbol{h}_1',\mathbf{y}_{K+1})-\ell(\boldsymbol{h}_1,\mathbf{y}_{K+1})|\\
&+\sum_{i=2}^{m}\sigma_i L_{\tau,\beta}(w_i)\ell(\boldsymbol{h}_i,\mathbf{y}_{K+1})+\sum_{i=2}^{m}\sigma_i L_{\tau,\beta}(w_i')\ell(\boldsymbol{h}_i',\mathbf{y}_{K+1})]\\
\le&\frac{1}{2}\mathbb{E}_{\sigma_2,\ldots,\sigma_m}[\sup_{\|w\|_k\le B,\boldsymbol{h}\in\mathcal{H}}\sup_{\|w'\|_k\le B,\boldsymbol{h}'\in\mathcal{H}} Lc|w_1-w_1'|\\
&+(B+\beta)|\ell(\boldsymbol{h}_1',\mathbf{y}_{K+1})-\ell(\boldsymbol{h}_1,\mathbf{y}_{K+1})|\\
&+\sum_{i=2}^{m}\sigma_i L_{\tau,\beta}(w_i)\ell(\boldsymbol{h}_i,\mathbf{y}_{K+1})+\sum_{i=2}^{m}\sigma_i L_{\tau,\beta}(w_i')\ell(\boldsymbol{h}_i',\mathbf{y}_{K+1})]\\
=&\mathbb{E}_{\sigma}[\sup_{\|w\|_k\le B,\boldsymbol{h}\in\mathcal{H}} Lc\sigma_1 w_1+(B+\beta)\sigma_1\ell(\boldsymbol{h}_1,\mathbf{y}_{K+1})+\sum_{i=2}^{m}\sigma_i L_{\tau,\beta}(w_i)\ell(\boldsymbol{h}_i,\mathbf{y}_{K+1})]
\end{aligned}
$$

Repeat the process $m-1$ times for $i=2,\ldots,m$.

$$
\begin{aligned}
\le&\mathbb{E}_{\sigma}[\sup_{\|w\|_k\le B,\boldsymbol{h}\in\mathcal{H}}\sum_{i=1}^{m} Lc\sigma_i w_i+\sum_{i=1}^{m}(B+\beta)\sigma_i\ell(\boldsymbol{h}_i,\mathbf{y}_{K+1})]\\
\le&mLc\widehat{\mathfrak{R}}_T(\mathcal{H}_{K,B})+m(B+\beta)\widehat{\mathfrak{R}}_T(\mathcal{F}),
\end{aligned}
$$

where $w_i = w(\tilde{\mathbf{x}}_i), \boldsymbol{h}_i = \boldsymbol{h}(\tilde{\mathbf{x}}_i)$, $L = \max\{1, \frac{\beta}{\tau}\}$, $\mathcal{H}_{K,B} = \{w : w \in \mathcal{H}_k, \|w\|_k \le B\}$ and $\mathcal{F} = \{\ell(\boldsymbol{h}, \mathbf{y}_{K+1}) : \boldsymbol{h} \in \mathcal{H}\}$.

According to Theorem 5.5 of [171], we obtain that

$$
\widehat{\mathfrak{R}}_T(\mathcal{H}_{K,B}) \le B\sqrt{\frac{1}{m}}.
$$

It is easy to check that

$$
\widehat{\mathfrak{R}}_T(\mathcal{F}) \le c\sqrt{\frac{4d\log m + 8d\log(K+1)}{m}},
$$

where d is the Natarajan Dimension of $\mathcal{H}$. Hence,

$$\widehat{\mathfrak{R}}_T(\mathcal{F}_B) \leq BLc\sqrt{\frac{1}{m}} + (B+\beta)c\sqrt{\frac{4d\log m + 8d\log(K+1)}{m}}.$$

This implies that for $1-2\delta > 0$, we have

$$\begin{aligned}
&\sup_{w\in\mathcal{H}_k, \|w\|_k \leq B} \sup_{\boldsymbol{h}\in\mathcal{H}} \Big| \frac{1}{m}\sum_{\mathbf{x}\in T} L_{\tau,\beta}(w(\mathbf{x}))\ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}) \\
&\qquad\qquad - \int_{\mathcal{X}} L_{\tau,\beta}(w(\mathbf{x}))\ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1})\mathrm{d}U(\mathbf{x}) \Big| \\
\leq & 2B\max\{1, \frac{\beta}{\tau}\}c\sqrt{\frac{1}{m}} + 2(B+\beta)c\sqrt{\frac{4d\log m + 8d\log(K+1)}{m}} \\
& + 4(B+\beta)c\sqrt{\frac{2\log(4/\delta)}{m}}.
\end{aligned} \tag{C.0.6}$$

Because $\|\widehat{w}\|_k = O_p(1)$, then combining inequality C.0.6, we know that

$$\begin{aligned}
&\sup_{\boldsymbol{h}\in\mathcal{H}} \Big| \frac{1}{m}\sum_{\mathbf{x}\in T} L_{\tau,\beta}(\widehat{w}(\mathbf{x}))\ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}) \\
&\qquad - \int_{\mathcal{X}} L_{\tau,\beta}(w(\mathbf{x}))\ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1})\mathrm{d}U(\mathbf{x}) \Big| \\
\leq & 2O_p(1)\max\{1, \frac{\beta}{\tau}\}cO_p(\sqrt{\frac{1}{m}}) \\
& + 2(O_p(1)+\beta)cO_p(\sqrt{\frac{4d\log m + 8d\log(K+1)}{m}}) \\
& + 4(O_p(1)+\beta)cO_p(\sqrt{\frac{2\log(4/\delta)}{m}}).
\end{aligned}$$

This implies that

$$\begin{aligned}
&\sup_{\boldsymbol{h}\in\mathcal{H}} \Big| \frac{1}{m}\sum_{\mathbf{x}\in T} L_{\tau,\beta}(\widehat{w}(\mathbf{x}))\ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}) \\
&\qquad - \int_{\mathcal{X}} L_{\tau,\beta}(w(\mathbf{x}))\ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1})\mathrm{d}U(\mathbf{x}) \Big| \\
= & c\big(\max\{1, \frac{\beta}{\tau}\} + \beta\big)O_p(\lambda_{n,m}^{\frac{1}{2}}).
\end{aligned}$$

Step 4. Using the results of Steps 1, 2 and 3, we have

$$
\begin{aligned}
&\sup_{\boldsymbol{h}\in\mathcal{H}}|R_Q(\boldsymbol{h},\mathbf{y}_{K+1})-\gamma\widehat{R}_T^{\tau,\beta}(\boldsymbol{h},\mathbf{y}_{K+1})|\\
\leq&\sup_{\boldsymbol{h}\in\mathcal{H}}|R_Q(\boldsymbol{h},\mathbf{y}_{K+1})-\gamma R_U^{\tau,\beta}(\boldsymbol{h},\mathbf{y}_{K+1})|+\sup_{\boldsymbol{h}\in\mathcal{H}}|\gamma R_U^{\tau,\beta}(\boldsymbol{h},\mathbf{y}_{K+1})\\
&-\gamma\int_{\mathcal{X}}L_{\tau,\beta}(\widehat{w}(\mathbf{x}))\ell(\boldsymbol{h}(\mathbf{x}),\mathbf{y}_{K+1})\mathrm{d}U(\mathbf{x})|\\
&+\sup_{\boldsymbol{h}\in\mathcal{H}}|\gamma\int_{\mathcal{X}}L_{\tau,\beta}(\widehat{w}(\mathbf{x}))\ell(\boldsymbol{h}(\mathbf{x}),\mathbf{y}_{K+1})\mathrm{d}U(\mathbf{x})-\gamma\widehat{R}_T^{\tau,\beta}(\boldsymbol{h},\mathbf{y}_{K+1})|\\
\leq&\gamma\beta cU(\{\mathbf{x}:\ 0<r(\mathbf{x})\leq 2\tau\})\\
&+\gamma\max\{c,\frac{c\beta}{\tau}\}O_p(\lambda_{n,m}^{\frac{1}{2}})+c\big(\max\{1,\frac{\beta}{\tau}\}+\beta\big)O_p(\lambda_{n,m}^{\frac{1}{2}}).
\end{aligned}
$$

Note that $\gamma<1$, we can write

$$
\begin{aligned}
&\sup_{\boldsymbol{h}\in\mathcal{H}}|R_Q(\boldsymbol{h},\mathbf{y}_{K+1})-\gamma\widehat{R}_T^{\tau,\beta}(\boldsymbol{h},\mathbf{y}_{K+1})|\\
\leq&\gamma\beta cU(\{\mathbf{x}:\ 0<r(\mathbf{x})\leq 2\tau\})+c\big(\max\{1,\frac{\beta}{\tau}\}+\beta\big)O_p(\lambda_{n,m}^{\frac{1}{2}}).
\end{aligned}
$$

□

C.1 Proof of Theorem 9

We separate the proof into three steps.

Step 1. We claim that

$$
\begin{aligned}
&\sup_{\boldsymbol{h}\in\mathcal{H}}|(1-\alpha)\Delta_{S,T}^{\tau,\beta}(\boldsymbol{h})-\max\{R_Q(\boldsymbol{h},\mathbf{y}_{K+1})-(1-\alpha)R_{P,k}(\boldsymbol{h},\mathbf{y}_{K+1}),0\}|\\
\leq&\gamma\beta cU(\{\mathbf{x}:\ 0<r(\mathbf{x})\leq 2\tau\})+c\big(\max\{1,\frac{\beta}{\tau}\}+\beta\big)O_p(\lambda_{n,m}^{\frac{1}{2}}).
\end{aligned}
$$

First, it is easy to check that

$$
\begin{aligned}
&\sup_{\boldsymbol{h}\in\mathcal{H}} |(1-\alpha)\Delta_{S,T}^{\tau,\beta}(\boldsymbol{h}) - \max\{R_Q(\boldsymbol{h},\mathbf{y}_{K+1}) - (1-\alpha)R_{P,k}(\boldsymbol{h},\mathbf{y}_{K+1}),0\}| \\
\leq &\sup_{\boldsymbol{h}\in\mathcal{H}} |(1-\alpha)\widehat{R}_T^{\tau,\beta}(\boldsymbol{h},\mathbf{y}_{K+1}) - (1-\alpha)\widehat{R}_S(\boldsymbol{h},\mathbf{y}_{K+1}) \\
&\qquad\qquad - R_Q(\boldsymbol{h},\mathbf{y}_{K+1}) + (1-\alpha)R_{P,k}(\boldsymbol{h},\mathbf{y}_{K+1})| \\
\leq &\sup_{\boldsymbol{h}\in\mathcal{H}} |(1-\alpha)\widehat{R}_T^{\tau,\beta}(\boldsymbol{h},\mathbf{y}_{K+1}) - R_Q(\boldsymbol{h},\mathbf{y}_{K+1})| \\
+&(1-\alpha)\sup_{\boldsymbol{h}\in\mathcal{H}} |\widehat{R}_S(\boldsymbol{h},\mathbf{y}_{K+1}) - R_{P,k}(\boldsymbol{h},\mathbf{y}_{K+1})| \\
\leq &\gamma\beta cU(\{\mathbf{x}:\ 0<r(\mathbf{x})\leq 2\tau\}) + c\big(\max\{1,\frac{\beta}{\tau}\}+\beta\big)O_p(\lambda_{n,m}^{\frac{1}{2}}) \text{ Lemma 28} \\
+&(1-\alpha)\sup_{\boldsymbol{h}\in\mathcal{H}} |\widehat{R}_S(\boldsymbol{h},\mathbf{y}_{K+1}) - R_{P,k}(\boldsymbol{h},\mathbf{y}_{K+1})| \\
\leq &\gamma\beta cU(\{\mathbf{x}:\ 0<r(\mathbf{x})\leq 2\tau\}) + c\big(\max\{1,\frac{\beta}{\tau}\}+\beta\big)O_p(\lambda_{n,m}^{\frac{1}{2}}) \\
+&(1-\alpha)cO_p(\lambda_{n,m}^{\frac{1}{2}}) \text{ Lemma 27.}
\end{aligned}
$$

Hence, we can write

$$
\begin{aligned}
&\sup_{\boldsymbol{h}\in\mathcal{H}} |(1-\alpha)\Delta_{S,T}^{\tau,\beta}(\boldsymbol{h}) \\
&\qquad - \max\{R_Q(\boldsymbol{h},\mathbf{y}_{K+1}) - (1-\alpha)R_{P,k}(\boldsymbol{h},\mathbf{y}_{K+1}),0\}| \\
\leq &\gamma\beta cU(\{\mathbf{x}:\ 0<r(\mathbf{x})\leq 2\tau\}) + c\big(\max\{1,\frac{\beta}{\tau}\}+\beta\big)O_p(\lambda_{n,m}^{\frac{1}{2}}).
\end{aligned}
$$

Step 2.

$$
\begin{aligned}
&\sup_{\boldsymbol{h}\in\mathcal{H}} |(1-\alpha)\widehat{R}_S(\boldsymbol{h}) + (1-\alpha)\Delta_{S,T}^{\tau,\beta}(\boldsymbol{h}) - (1-\alpha)R_{P,k}(\boldsymbol{h}) \\
&\qquad - \max\{R_Q(\boldsymbol{h},\mathbf{y}_{K+1}) - (1-\alpha)R_{P,k}(\boldsymbol{h},\mathbf{y}_{K+1}),0\}| \\
\leq &\sup_{\boldsymbol{h}\in\mathcal{H}} |(1-\alpha)\widehat{R}_S(\boldsymbol{h}) - (1-\alpha)R_{P,k}(\boldsymbol{h})| \\
+&\sup_{\boldsymbol{h}\in\mathcal{H}} |(1-\alpha)\Delta_{S,T}^{\tau,\beta}(\boldsymbol{h}) - \max\{R_Q(\boldsymbol{h},\mathbf{y}_{K+1}) - (1-\alpha)R_{P,k}(\boldsymbol{h},\mathbf{y}_{K+1}),0\}| \\
\leq &(1-\alpha)cO_p(\lambda_{n,m}^{\frac{1}{2}}) \quad \text{Use Lemma 27} \\
+&\gamma\beta cU(\{\mathbf{x}:\ 0<r(\mathbf{x})\leq 2\tau\}) + c\big(\max\{1,\frac{\beta}{\tau}\}+\beta\big)O_p(\lambda_{n,m}^{\frac{1}{2}})
\end{aligned}
$$

Use the result of Step 1.

Hence, we can write

$$\begin{aligned}
&\sup_{\boldsymbol{h}\in\mathcal{H}} |(1-\alpha)\widehat{R}_S(\boldsymbol{h}) + (1-\alpha)\Delta_{S,T}^{\tau,\beta}(\boldsymbol{h}) - (1-\alpha)R_{P,k}(\boldsymbol{h}) \\
&\qquad - \max\{R_Q(\boldsymbol{h}, \mathbf{y}_{K+1}) - (1-\alpha)R_{P,k}(\boldsymbol{h}, \mathbf{y}_{K+1}), 0\}| \\
\leq &\gamma\beta c U(\{\mathbf{x}:\ 0 < r(\mathbf{x}) \leq 2\tau\}) + c\big(\max\{1, \frac{\beta}{\tau}\} + \beta\big) O_p(\lambda_{n,m}^{\frac{1}{2}}).
\end{aligned}$$

Step 3. Note that, using Lemma 26,

$$R_Q^\alpha(\boldsymbol{h}) = (1-\alpha)R_{P,k}(\boldsymbol{h}) + \max\{R_Q(\boldsymbol{h}, \mathbf{y}_{K+1}) - (1-\alpha)R_{P,k}(\boldsymbol{h}, \mathbf{y}_{K+1}), 0\}.$$

Hence,

$$\begin{aligned}
&\sup_{\boldsymbol{h}\in\mathcal{H}} |(1-\alpha)\widehat{R}_S(\boldsymbol{h}) + (1-\alpha)\Delta_{S,T}^{\tau,\beta}(\boldsymbol{h}) - R_Q^\alpha(\boldsymbol{h})| \\
\leq &\gamma\beta c U(\{\mathbf{x}:\ 0 < r(\mathbf{x}) \leq 2\tau\}) + c\big(\max\{1, \frac{\beta}{\tau}\} + \beta\big) O_p(\lambda_{n,m}^{\frac{1}{2}}).
\end{aligned}$$

C.2 Proof of Theorem 10

According to Theorem 7, we know that for any $\boldsymbol{h} \in \mathcal{H}$,

$$|R_P^\alpha(\boldsymbol{h}) - R_Q^\alpha(\boldsymbol{h})| \leq \alpha d_{\boldsymbol{h},\mathcal{H}}^{\ell}(P_{X|Y=\mathbf{y}_{K+1}}, Q_{X|Y=\mathbf{y}_{K+1}}) + \alpha\Lambda. \tag{C.2.1}$$

According to Theorem 9, we know that for any $\boldsymbol{h} \in \mathcal{H}$,

$$\begin{aligned}
&|(1-\alpha)\widehat{R}_S(\boldsymbol{h}) + (1-\alpha)\Delta_{S,T}^{\tau,\beta}(\boldsymbol{h}) - R_Q^\alpha(\boldsymbol{h})| \\
\leq &\gamma\beta c U(\{\mathbf{x}:\ 0 < r(\mathbf{x}) \leq 2\tau\}) + c\big(\max\{1, \frac{\beta}{\tau}\} + \beta\big) O_p(\lambda_{n,m}^{\frac{1}{2}}).
\end{aligned} \tag{C.2.2}$$

Combining inequalities (C.2.1) and (C.2.2), we know that for any $\boldsymbol{h} \in \mathcal{H}$,

$$\begin{aligned}
&|(1-\alpha)\widehat{R}_S(\boldsymbol{h}) + (1-\alpha)\Delta_{S,T}^{\tau,\beta}(\boldsymbol{h}) - R_P^\alpha(\boldsymbol{h})| \\
\leq &\gamma\beta c U(\{\mathbf{x}:\ 0 < r(\mathbf{x}) \leq 2\tau\}) + c\big(\max\{1, \frac{\beta}{\tau}\} + \beta\big) O_p(\lambda_{n,m}^{\frac{1}{2}}) \\
+ &\alpha d_{\boldsymbol{h},\mathcal{H}}^{\ell}(P_{X|Y=\mathbf{y}_{K+1}}, Q_{X|Y=\mathbf{y}_{K+1}}) + \alpha\Lambda.
\end{aligned}$$

C.3 Proof of Theorem 11

Assume that

$$\widehat{\boldsymbol{h}} \in \arg\min_{\boldsymbol{h}\in\mathcal{H}} \widehat{R}^{\tau,\beta}_{S,T}(\boldsymbol{h}), \quad \boldsymbol{h}_Q \in \arg\min_{\boldsymbol{h}\in\mathcal{H}} R^{\alpha}_{Q}(\boldsymbol{h}).$$

Step 1. It is easy to check that

$$\begin{aligned}
R^{\alpha}_{Q}(\widehat{\boldsymbol{h}}) - R^{\alpha}_{Q}(\boldsymbol{h}_Q) =& R^{\alpha}_{Q}(\widehat{\boldsymbol{h}}) - (1-\alpha)\widehat{R}^{\tau,\beta}_{S,T}(\widehat{\boldsymbol{h}}) \\
&+(1-\alpha)\widehat{R}^{\tau,\beta}_{S,T}(\widehat{\boldsymbol{h}}) - R^{\alpha}_{Q}(\boldsymbol{h}_Q) \\
\leq& R^{\alpha}_{Q}(\widehat{\boldsymbol{h}}) - (1-\alpha)\widehat{R}^{\tau,\beta}_{S,T}(\widehat{\boldsymbol{h}}) \\
&+(1-\alpha)\widehat{R}^{\tau,\beta}_{S,T}(\boldsymbol{h}_Q) - R^{\alpha}_{Q}(\boldsymbol{h}_Q) \\
\leq& 2\sup_{\boldsymbol{h}\in\mathcal{H}} |(1-\alpha)\widehat{R}^{\tau,\beta}_{S,T}(\boldsymbol{h}) - R^{\alpha}_{Q}(\boldsymbol{h})|,
\end{aligned}$$

and

$$\begin{aligned}
R^{\alpha}_{Q}(\widehat{\boldsymbol{h}}) - R^{\alpha}_{Q}(\boldsymbol{h}_Q) =& R^{\alpha}_{Q}(\widehat{\boldsymbol{h}}) - (1-\alpha)\widehat{R}^{\tau,\beta}_{S,T}(\boldsymbol{h}_Q) \\
&+(1-\alpha)\widehat{R}^{\tau,\beta}_{S,T}(\boldsymbol{h}_Q) - R^{\alpha}_{Q}(\boldsymbol{h}_Q) \\
\geq& R^{\alpha}_{Q}(\widehat{\boldsymbol{h}}) - (1-\alpha)\widehat{R}^{\tau,\beta}_{S,T}(\widehat{\boldsymbol{h}}) \\
&+(1-\alpha)\widehat{R}^{\tau,\beta}_{S,T}(\boldsymbol{h}_Q) - R^{\alpha}_{Q}(\boldsymbol{h}_Q) \\
\geq& -2\sup_{\boldsymbol{h}\in\mathcal{H}} |(1-\alpha)\widehat{R}^{\tau,\beta}_{S,T}(\boldsymbol{h}) - R^{\alpha}_{Q}(\boldsymbol{h})|,
\end{aligned}$$

which implies that

$$|R^{\alpha}_{Q}(\widehat{\boldsymbol{h}}) - R^{\alpha}_{Q}(\boldsymbol{h}_Q)| \leq 2\sup_{\boldsymbol{h}\in\mathcal{H}} |(1-\alpha)\widehat{R}^{\tau,\beta}_{S,T}(\boldsymbol{h}) - R^{\alpha}_{Q}(\boldsymbol{h})|.$$

Using the result of Theorem 9, we obtain that

$$\begin{aligned}
|R^{\alpha}_{Q}(\widehat{\boldsymbol{h}}) - R^{\alpha}_{Q}(\boldsymbol{h}_Q)| \leq& 2c\big(\max\{1, \frac{\beta}{\tau}\} + \beta\big)O_p(\lambda^{\frac{1}{2}}_{n,m}) \\
&+2\gamma c\beta U(0 < p/q \leq 2\tau).
\end{aligned} \tag{C.3.1}$$

Then, using the result of Step 3 in the proof of Theorem 8, we obtain

that

$$
\begin{aligned}
|R_Q^\alpha(\widehat{\boldsymbol{h}}) - R_P^\alpha(\boldsymbol{h}_Q)| \leq & 2c\big(\max\{1, \frac{\beta}{\tau}\} + \beta\big) O_p(\lambda_{n,m}^{\frac{1}{2}}) \\
& + 2\gamma c\beta U(0 < p/q \leq 2\tau).
\end{aligned} \tag{C.3.2}
$$

Step 2.

$$
\begin{aligned}
\widehat{R}_{S,T}^{\tau,\beta}(\widehat{\boldsymbol{h}}) &= (1-\alpha)\widehat{R}_S(\widehat{\boldsymbol{h}}) + (1-\alpha)\Delta_{S,T}^{\tau,\beta}(\widehat{\boldsymbol{h}}) \\
&\leq (1-\alpha)\widehat{R}_S(\boldsymbol{h}_Q) + (1-\alpha)\Delta_{S,T}^{\tau,\beta}(\boldsymbol{h}_Q) \\
&\leq R_Q^\alpha(\boldsymbol{h}_Q) + c\big(\max\{1, \frac{\beta}{\tau}\} + \beta\big) O_p(\lambda_{n,m}^{\frac{1}{2}}) \\
&\quad + \gamma c\beta U(0 < p/q \leq 2\tau) \text{ Theorem 9} \\
&= (1-\alpha) \min_{\boldsymbol{h}\in\mathcal{H}} R_{Q,k}(\boldsymbol{h}) \\
&\quad + c\big(\max\{1, \frac{\beta}{\tau}\} + \beta\big) O_p(\lambda_{n,m}^{\frac{1}{2}}) + \gamma c\beta U(0 < p/q \leq 2\tau) \\
&\quad \text{Using the result of Step 3 in proof of Theorem 8} \\
&\leq (1-\alpha) R_{Q,k}(\widehat{\boldsymbol{h}}) \\
&\quad + c\big(\max\{1, \frac{\beta}{\tau}\} + \beta\big) O_p(\lambda_{n,m}^{\frac{1}{2}}) + \gamma c\beta U(0 < p/q \leq 2\tau).
\end{aligned}
$$

Hence,

$$
\begin{aligned}
& (1-\alpha)\Delta_{S,T}^{\tau,\beta}(\widehat{\boldsymbol{h}}) \\
\leq & (1-\alpha) R_{Q,k}(\widehat{\boldsymbol{h}}) - (1-\alpha)\widehat{R}_S(\widehat{\boldsymbol{h}}) \\
& + c\big(\max\{1, \frac{\beta}{\tau}\} + \beta\big) O_p(\lambda_{n,m}^{\frac{1}{2}}) + \gamma c\beta U(0 < p/q \leq 2\tau) \\
= & (1-\alpha) R_{P,k}(\widehat{\boldsymbol{h}}) - (1-\alpha)\widehat{R}_S(\widehat{\boldsymbol{h}}) \\
& + c\big(\max\{1, \frac{\beta}{\tau}\} + \beta\big) O_p(\lambda_{n,m}^{\frac{1}{2}}) + \gamma c\beta U(0 < p/q \leq 2\tau) \\
\leq & (1-\alpha) c O_p(\lambda_{n,m}^{\frac{1}{2}}) + c\big(\max\{1, \frac{\beta}{\tau}\} + \beta\big) O_p(\lambda_{n,m}^{\frac{1}{2}}) \\
& + \gamma c\beta U(0 < p/q \leq 2\tau) \text{ Using Lemma 27} \\
\leq & 2c\big(\max\{1, \frac{\beta}{\tau}\} + \beta\big) O_p(\lambda_{n,m}^{\frac{1}{2}}) + \gamma c\beta U(0 < p/q \leq 2\tau).
\end{aligned}
$$

Then, combining above inequality with the result of Step 1 in the proof

of Theorem 9, we obtain that

$$\begin{aligned}&\max\{R_Q(\widehat{\boldsymbol{h}}, \mathbf{y}_{K+1}) - (1-\alpha)R_{P,k}(\widehat{\boldsymbol{h}}, \mathbf{y}_{K+1}), 0\}\\ \leq& 3c\big(\max\{1, \frac{\beta}{\tau}\} + \beta\big)O_p(\lambda_{n,m}^{\frac{1}{2}}) + 2\gamma c\beta U(0 < p/q \leq 2\tau).\end{aligned}$$

Because $\max\{R_Q(\widehat{\boldsymbol{h}}, \mathbf{y}_{K+1}) - (1-\alpha)R_{P,k}(\widehat{\boldsymbol{h}}, \mathbf{y}_{K+1}), 0\} = \alpha R_{Q,u}(\widehat{\boldsymbol{h}})$, we obtain that

$$\alpha R_{Q,u}(\widehat{\boldsymbol{h}}) \leq 3c\big(\max\{1, \frac{\beta}{\tau}\} + \beta\big)O_p(\lambda_{n,m}^{\frac{1}{2}}) + 2\gamma c\beta U(0 < p/q \leq 2\tau).$$

Step 3.

$$\begin{aligned}&|R_P^\alpha(\widehat{\boldsymbol{h}}) - R_P^\alpha(\boldsymbol{h}_Q)|\\ \leq& |R_P^\alpha(\widehat{\boldsymbol{h}}) - R_Q^\alpha(\widehat{\boldsymbol{h}})| + |R_Q^\alpha(\widehat{\boldsymbol{h}}) - R_P^\alpha(\boldsymbol{h}_Q)|\\ =& \alpha|R_{P,u}(\widehat{\boldsymbol{h}}) - R_{Q,u}^\alpha(\widehat{\boldsymbol{h}})| + |R_Q^\alpha(\widehat{\boldsymbol{h}}) - R_P^\alpha(\boldsymbol{h}_Q)|\\ \leq& \alpha R_{Q,u}(\widehat{\boldsymbol{h}}) + |R_Q^\alpha(\widehat{\boldsymbol{h}}) - R_P^\alpha(\boldsymbol{h}_Q)|\\ \leq& 5c\big(\max\{1, \frac{\beta}{\tau}\} + \beta\big)O_p(\lambda_{n,m}^{\frac{1}{2}})\\ &+4\gamma c\beta U(0 < p/q \leq 2\tau) \text{ Using the results of Step 1 and Step 2.}\end{aligned}$$

Briefly, we can write (absorbing coefficient 5 into O_p)

$$|R_P^\alpha(\widehat{\boldsymbol{h}}) - R_P^\alpha(\boldsymbol{h}_Q)| \leq c\big(\max\{1, \frac{\beta}{\tau}\} + \beta\big)O_p(\lambda_{n,m}^{\frac{1}{2}}) + 4\gamma c\beta U(0 < p/q \leq 2\tau).$$

Combining above inequality with Theorem 8, we obtain that

$$\begin{aligned}|R_P^\alpha(\widehat{\boldsymbol{h}}) - \min_{\boldsymbol{h}\in\mathcal{H}} R_P^\alpha(\boldsymbol{h})| \leq& c\big(\max\{1, \frac{\beta}{\tau}\} + \beta\big)O_p(\lambda_{n,m}^{\frac{1}{2}})\\ &+4\gamma c\beta U(0 < p/q \leq 2\tau).\end{aligned}$$

C.4 Proof for Theorem 12

Lemma 29. *Assume the feature space $\mathcal{X}$ is compact and the loss function has an upper bound c. Let the RKHS $\mathcal{H}_K$ is the Hilbert space with Gaussian kernel. Suppose that the real density $p/q \in \mathcal{H}_K$ and set the*

regularization parameter $\lambda = \lambda_{n,m}$ *in* KuLSIF *such that*

$$\lim_{n,m\to 0} \lambda_{n,m} = 0, \quad \lambda_{n,m}^{-1} = O(\min\{n,m\}^{1-\delta}),$$

where $0 < \delta < 1$ *is any constant, then*

$$\sup_{\boldsymbol{h}\in\mathcal{H}} |R_{Q,u}(\boldsymbol{h}) - \gamma' \widehat{R}^{\tau,\beta}_{S,T,u}(\boldsymbol{h})| \le \gamma'\beta c U(\{\mathbf{x}:\ 0 < r(\mathbf{x}) \le 2\tau\}) + c\big(\max\{1, \frac{\beta}{\tau}\} + \beta\big) O_p(\lambda_{n,m}^{\frac{1}{2}}),$$

where $\gamma' = 1/\big(\beta U(\{\mathbf{x}:\ r(\mathbf{x}) = 0\})\big)$, *and*

$$R_{Q,u}(\boldsymbol{h}) = \int_{\mathcal{X}} \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}) \mathrm{d}Q_{X|Y=\mathbf{y}_{K+1}}(\mathbf{x}),$$
$$\widehat{R}^{\tau,\beta}_{S,T,u}(\boldsymbol{h}) = \frac{1}{m} \sum_{\mathbf{x}\in T} L_{\tau,\beta}(\widehat{w}(\mathbf{x})) \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}),$$

here $Q_{XY} = Q_U^{0,\beta} P_{Y|X}$, $\widehat{w}$ *is the solution of* KuLSIF, *and*

$$L^-_{\tau,\beta}(x) = \begin{cases} x + \beta, & x \le \tau; \\ 0, & 2\tau \le x; \\ -\dfrac{\tau+\beta}{\tau} x + 2\tau + 2\beta, & \tau < x < 2\tau. \end{cases} \tag{C.4.1}$$

Proof. **Step 1.** We claim that

$$\sup_{\boldsymbol{h}\in\mathcal{H}} |R_{Q,u}(\boldsymbol{h}) - \gamma' R^{\tau,\beta}_{U,u}(\boldsymbol{h})| \le \gamma'\beta c U(\{\mathbf{x}:\ 0 < r(\mathbf{x}) \le 2\tau\}),$$

where

$$R^{\tau,\beta}_{U,u}(\boldsymbol{h}) = \int_{\mathcal{X}} L^-_{\tau,\beta}(r(\mathbf{x})) \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}) \mathrm{d}U(\mathbf{x}),$$

here $r(\mathbf{x}) = p(\mathbf{x})/q(\mathbf{x})$.

First, we note that

$$\begin{aligned} &\Big| \int_{\mathcal{X}} L^-_{0,\beta}(r(\mathbf{x})) \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}) \mathrm{d}U(\mathbf{x}) - \int_{\mathcal{X}} L^-_{\tau,\beta}(r(\mathbf{x})) \ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}) \mathrm{d}U(\mathbf{x}) \Big| \\ \le & c \int_{\{\mathbf{x}:\ 0<r(\mathbf{x})\le 2\tau\}} (\tau+\beta) \mathrm{d}U(\mathbf{x}) = (\tau+\beta) c U(\{\mathbf{x}:\ 0 < r(\mathbf{x}) \le 2\tau\}). \end{aligned} \tag{C.4.2}$$

Because $Q_{XY} = Q_U^{0,\beta} P_{Y|X}$, then according to the definition of $Q_U^{0,\beta}$, we know

$$R_{Q,u}(\boldsymbol{h}) = \gamma' \int_{\mathcal{X}} L_{0,\beta}^{-}(r(\mathbf{x}))\ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1})\mathrm{d}U(\mathbf{x}),$$

which implies

$$\sup_{\boldsymbol{h}\in\mathcal{H}} |R_{Q,u}(\boldsymbol{h}) - \gamma' R_{U,u}^{\tau,\beta}(\boldsymbol{h})| \leq \gamma'(\tau+\beta)cU(\{\mathbf{x}:\ 0 < r(\mathbf{x}) \leq 2\tau\}).$$

Step 2. We claim that

$$\sup_{\boldsymbol{h}\in\mathcal{H}} |R_{U,u}^{\tau,\beta}(\boldsymbol{h}) - \int_{\mathcal{X}} L_{\tau,\beta}^{-}(\widehat{w}(\mathbf{x}))\ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1})\mathrm{d}U(\mathbf{x})| \leq (c + \frac{c\beta}{\tau})O_p(\lambda_{n,m}^{\frac{1}{2}}).$$

First, the Lipschitz constant for $L_{\tau,\beta}^{-}$ is smaller than $1 + \frac{\beta}{\tau}$. Then,

$$\begin{aligned}
&\sup_{\boldsymbol{h}\in\mathcal{H}} |R_{U,u}^{\tau,\beta}(\boldsymbol{h}) - \int_{\mathcal{X}} L_{\tau,\beta}^{-}(\widehat{w}(\mathbf{x}))\ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1})\mathrm{d}U(\mathbf{x})| \\
=&\sup_{\boldsymbol{h}\in\mathcal{H}} |\int_{\mathcal{X}} L_{\tau,\beta}^{-}(r(\mathbf{x}))\ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1})\mathrm{d}U(\mathbf{x}) \\
&\qquad - \int_{\mathcal{X}} L_{\tau,\beta}^{-}(\widehat{w}(\mathbf{x}))\ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1})\mathrm{d}U(\mathbf{x})| \\
\leq&\sup_{\boldsymbol{h}\in\mathcal{H}} \int_{\mathcal{X}} |L_{\tau,\beta}^{-}(r(\mathbf{x})) - L_{\tau,\beta}^{-}(\widehat{w}(\mathbf{x}))|\ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1})\mathrm{d}U(\mathbf{x}) \\
\leq&\sup_{\boldsymbol{h}\in\mathcal{H}} \sqrt{\int_{\mathcal{X}} |L_{\tau,\beta}^{-}(r(\mathbf{x})) - L_{\tau,\beta}^{-}(\widehat{w}(\mathbf{x}))|^2\mathrm{d}U(\mathbf{x})}\sqrt{\int_{\mathcal{X}} \ell^2(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1})\mathrm{d}U(\mathbf{x})} \\
\leq&c\sup_{\boldsymbol{h}\in\mathcal{H}} \sqrt{\int_{\mathcal{X}} |L_{\tau,\beta}^{-}(r(\mathbf{x})) - L_{\tau,\beta}^{-}(\widehat{w}(\mathbf{x}))|^2\mathrm{d}U(\mathbf{x})} \\
\leq&(c + \frac{c\beta}{\tau})\sup_{\boldsymbol{h}\in\mathcal{H}} \sqrt{\int_{\mathcal{X}} |r(\mathbf{x}) - \widehat{w}(\mathbf{x})|^2\mathrm{d}U(\mathbf{x})}.
\end{aligned}$$

Lastly, using Lemma 25,

$$\sup_{\boldsymbol{h}\in\mathcal{H}} |R_{U,u}^{\tau,\beta}(\boldsymbol{h}) - \int_{\mathcal{X}} L_{\tau,\beta}^{-}(\widehat{w}(\mathbf{x}))\ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1})\mathrm{d}U(\mathbf{x})| \leq (c + \frac{c\beta}{\tau})O_p(\lambda_{n,m}^{\frac{1}{2}}).$$

Step 3. We claim that

$$\begin{aligned}
&\sup_{\boldsymbol{h}\in\mathcal{H}} \big|\frac{1}{m}\sum_{\mathbf{x}\in T} L_{\tau,\beta}^{-}(\widehat{w}(\mathbf{x}))\ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1}) \\
&\qquad - \int_{\mathcal{X}} L_{\tau,\beta}^{-}(\widehat{w}(\mathbf{x}))\ell(\boldsymbol{h}(\mathbf{x}), \mathbf{y}_{K+1})\mathrm{d}U(\mathbf{x})\big| \leq c\big(1 + \frac{\beta}{\tau} + \beta\big)O_p(\lambda_{n,m}^{\frac{1}{2}}).
\end{aligned}$$

First, we set $\mathcal{F}_B = \{L^-_{\tau,\beta}(w)\ell(\boldsymbol{h}, \mathbf{y}_{K+1}) : w \in \mathcal{H}_K, \|w\|_k \leq B, \boldsymbol{h} \in \mathcal{H}\}$. We consider

$$\sup_{f\in\mathcal{F}_B} \big|\frac{1}{m}\sum_{\mathbf{x}\in T} f(\mathbf{x}) - \int_{\mathcal{X}} f(\mathbf{x})\mathrm{d}U(\mathbf{x})\big|$$

Using Lemma 23, it is easy to check that for $1 - 2\delta > 0$, we have

$$\sup_{f\in\mathcal{F}_B} \big|\frac{1}{m}\sum_{\mathbf{x}\in T} f(\mathbf{x}) - \int_{\mathcal{X}} f(\mathbf{x})\mathrm{d}U(\mathbf{x})\big| \leq 2\widehat{\Re}_T(\mathcal{F}_B) + 4(\tau+\beta)c\sqrt{\frac{2\log(4/\delta)}{m}}, \tag{C.4.3}$$

here we have used $|f| \leq (\tau + \beta)c$, for any $f \in \mathcal{F}_B$. Then, we consider $\widehat{\Re}_T(\mathcal{F}_B)$.

$$\begin{aligned}
&m\widehat{\Re}_T(\mathcal{F}_B)\\
=&\mathbb{E}_\sigma[\sup_{\|w\|_k\leq B,\boldsymbol{h}\in\mathcal{H}} \sum_{i=1}^m \sigma_i L^-_{\tau,\beta}(w_i)\ell(\boldsymbol{h}_i, \mathbf{y}_{K+1})]\\
=&\mathbb{E}_\sigma[\sup_{\|w\|_k\leq B,\boldsymbol{h}\in\mathcal{H}} \sigma_1 L^-_{\tau,\beta}(w_1)\ell(\boldsymbol{h}_1, \mathbf{y}_{K+1}) + \sum_{i=2}^m \sigma_i L^-_{\tau,\beta}(w_i)\ell(\boldsymbol{h}_i, \mathbf{y}_{K+1})]\\
=&\frac{1}{2}\mathbb{E}_{\sigma_2,\ldots,\sigma_m}[\sup_{\|w\|_k\leq B,\boldsymbol{h}\in\mathcal{H}} L^-_{\tau,\beta}(w_1)\ell(\boldsymbol{h}_1, \mathbf{y}_{K+1}) + \sum_{i=2}^m \sigma_i L^-_{\tau,\beta}(w_i)\ell(\boldsymbol{h}_i, \mathbf{y}_{K+1})\\
&+ \sup_{\|w\|_k\leq B,\boldsymbol{h}\in\mathcal{H}} -L^-_{\tau,\beta}(w_1)\ell(\boldsymbol{h}_1, \mathbf{y}_{K+1}) + \sum_{i=2}^m \sigma_i L^-_{\tau,\beta}(w_i)\ell(\boldsymbol{h}_i, \mathbf{y}_{K+1})]\\
=&\frac{1}{2}\mathbb{E}_{\sigma_2,\ldots,\sigma_m}[\sup_{\|w\|_k\leq B,\boldsymbol{h}\in\mathcal{H}} \sup_{\|w'\|_k\leq B,\boldsymbol{h}'\in\mathcal{H}} L^-_{\tau,\beta}(w_1)\ell(\boldsymbol{h}_1, \mathbf{y}_{K+1})\\
&+\sum_{i=2}^m \sigma_i L^-_{\tau,\beta}(w_i)\ell(\boldsymbol{h}_i, \mathbf{y}_{K+1}) - L^-_{\tau,\beta}(w'_1)\ell(\boldsymbol{h}'_1, \mathbf{y}_{K+1})\\
&+\sum_{i=2}^m \sigma_i L^-_{\tau,\beta}(w'_i)\ell(\boldsymbol{h}'_i, \mathbf{y}_{K+1})]\\
\leq&\frac{1}{2}\mathbb{E}_{\sigma_2,\ldots,\sigma_m}[\sup_{\|w\|_k\leq B,\boldsymbol{h}\in\mathcal{H}} \sup_{\|w'\|_k\leq B,\boldsymbol{h}'\in\mathcal{H}} |L^-_{\tau,\beta}(w_1) - L^-_{\tau,\beta}(w'_1)|\ell(\boldsymbol{h}'_1, \mathbf{y}_{K+1})\\
&+L^-_{\tau,\beta}(w_1)|\ell(\boldsymbol{h}'_1, \mathbf{y}_{K+1}) - \ell(\boldsymbol{h}_1, \mathbf{y}_{K+1})|\\
&+\sum_{i=2}^m \sigma_i L^-_{\tau,\beta}(w_i)\ell(\boldsymbol{h}_i, \mathbf{y}_{K+1}) + \sum_{i=2}^m \sigma_i L^-_{\tau,\beta}(w'_i)\ell(\boldsymbol{h}'_i, \mathbf{y}_{K+1})]\\
\leq&\frac{1}{2}\mathbb{E}_{\sigma_2,\ldots,\sigma_m}[\sup_{\|w\|_k\leq B,\boldsymbol{h}\in\mathcal{H}} \sup_{\|w'\|_k\leq B,\boldsymbol{h}'\in\mathcal{H}} Lc|w_1 - w'_1|\\
&+(B+\beta)|\ell(\boldsymbol{h}'_1, \mathbf{y}_{K+1}) - \ell(\boldsymbol{h}_1, \mathbf{y}_{K+1})|\\
&+\sum_{i=2}^m \sigma_i L^-_{\tau,\beta}(w_i)\ell(\boldsymbol{h}_i, \mathbf{y}_{K+1}) + \sum_{i=2}^m \sigma_i L^-_{\tau,\beta}(w'_i)\ell(\boldsymbol{h}'_i, \mathbf{y}_{K+1})]
\end{aligned}$$

$$
\begin{aligned}
=&\mathbb{E}_\sigma[\sup_{\|w\|_k\leq B,\boldsymbol{h}\in\mathcal{H}} Lc\sigma_1 w_1+(B+\beta)\sigma_1\ell(\boldsymbol{h}_1,\mathbf{y}_{K+1})\\
&+\sum_{i=2}^{m}\sigma_i L^-_{\tau,\beta}(w_i)\ell(\boldsymbol{h}_i,\mathbf{y}_{K+1})]\\
&\text{Repeat the process } m-1 \text{ times for } i=2,...,m.\\
\leq&\mathbb{E}_\sigma[\sup_{\|w\|_k\leq B,\boldsymbol{h}\in\mathcal{H}}\sum_{i=1}^{m} Lc\sigma_i w_i+\sum_{i=1}^{m}(B+\beta)\sigma_i\ell(\boldsymbol{h}_i,\mathbf{y}_{K+1})]\\
\leq&mLc\widehat{\Re}_T(\mathcal{H}_{K,B})+m(B+\beta)\widehat{\Re}_T(\mathcal{F}),
\end{aligned}
$$

where $w_i=w(\tilde{\mathbf{x}}_i),\boldsymbol{h}_i=\boldsymbol{h}(\tilde{\mathbf{x}}_i)$, $L=1+\frac{\beta}{\tau}$, $\mathcal{H}_{K,B}=\{w:w\in\mathcal{H}_K,\|w\|_k\leq B\}$ and $\mathcal{F}=\{\ell(\boldsymbol{h},\mathbf{y}_{K+1}):\boldsymbol{h}\in\mathcal{H}\}$.

According to Theorem 5.5 of [171], we obtain that

$$
\widehat{\Re}_T(\mathcal{H}_{K,B})\leq B\sqrt{\frac{1}{m}}.
$$

According to the proving process of Lemma 27, we obtain that

$$
\widehat{\Re}_T(\mathcal{F})\leq c\sqrt{\frac{4d\log m+8d\log(K+1)}{m}},
$$

where d is the Natarajan Dimension of $\mathcal{H}$.

Hence,

$$
\widehat{\Re}_T(\mathcal{F}_B)\leq BLc\sqrt{\frac{1}{m}}+(B+\beta)c\sqrt{\frac{4d\log m+8d\log(K+1)}{m}}.
$$

This implies that for $1-2\delta>0$, we have

$$
\begin{aligned}
&\sup_{w\in\mathcal{H}_K,\|w\|_k\leq B}\sup_{\boldsymbol{h}\in\mathcal{H}}\Big|\frac{1}{m}\sum_{\mathbf{x}\in T}L^-_{\tau,\beta}(w(\mathbf{x}))\ell(\boldsymbol{h}(\mathbf{x}),\mathbf{y}_{K+1})\\
&\qquad\qquad-\int_{\mathcal{X}}L^-_{\tau,\beta}(w(\mathbf{x}))\ell(\boldsymbol{h}(\mathbf{x}),\mathbf{y}_{K+1})\mathrm{d}U(\mathbf{x})\Big|\\
\leq&2B(1+\frac{\beta}{\tau})c\sqrt{\frac{1}{m}}+4(\tau+\beta)c\sqrt{\frac{2\log(4/\delta)}{m}}\\
&+2(B+\beta)c\sqrt{\frac{4d\log m+8d\log(K+1)}{m}}.
\end{aligned}
\tag{C.4.4}
$$

Because $\|\widehat{w}\|_k = O_p(1)$, then combining inequality C.4.4, we know that

$$\begin{aligned}
&\sup_{\boldsymbol{h}\in\mathcal{H}} \Big|\frac{1}{m}\sum_{\mathbf{x}\in T} L^-_{\tau,\beta}(\widehat{w}(\mathbf{x}))\ell(\boldsymbol{h}(\mathbf{x}),\mathbf{y}_{K+1}) \\
&\qquad - \int_{\mathcal{X}} L^-_{\tau,\beta}(w(\mathbf{x}))\ell(\boldsymbol{h}(\mathbf{x}),\mathbf{y}_{K+1})\mathrm{d}U(\mathbf{x})\Big| \\
\leq& 4(\tau+\beta)cO_p(\sqrt{\frac{2\log(4/\delta)}{m}}) + 2O_p(1)(1+\frac{\beta}{\tau})cO_p(\sqrt{\frac{1}{m}}) \\
&+2(O_p(1)+\beta)cO_p(\sqrt{\frac{4d\log m + 8d\log(K+1)}{m}}).
\end{aligned}$$

This implies that

$$\begin{aligned}
&\sup_{\boldsymbol{h}\in\mathcal{H}} \Big|\frac{1}{m}\sum_{\mathbf{x}\in T} L^-_{\tau,\beta}(\widehat{w}(\mathbf{x}))\ell(\boldsymbol{h}(\mathbf{x}),\mathbf{y}_{K+1}) \\
&\qquad - \int_{\mathcal{X}} L^-_{\tau,\beta}(w(\mathbf{x}))\ell(\boldsymbol{h}(\mathbf{x}),\mathbf{y}_{K+1})\mathrm{d}U(\mathbf{x})\Big| \\
\leq& c\big(1+\frac{\beta}{\tau}+\tau+\beta\big)O_p(\lambda_{n,m}^{\frac{1}{2}}).
\end{aligned}$$

Step 4. Using the results of Steps 1, 2 and 3, we have

$$\begin{aligned}
&\sup_{\boldsymbol{h}\in\mathcal{H}} |R_{Q,u}(\boldsymbol{h}) - \gamma'\widehat{R}^{\tau,\beta}_{S,T,u}(\boldsymbol{h})| \\
\leq& \sup_{\boldsymbol{h}\in\mathcal{H}} |R_{Q,u}(\boldsymbol{h}) - \gamma' R^{\tau,\beta}_{U,u}(\boldsymbol{h})| \\
&+ \sup_{\boldsymbol{h}\in\mathcal{H}} |\gamma' R^{\tau,\beta}_{U,u}(\boldsymbol{h}) - \gamma'\int_{\mathcal{X}} L^-_{\tau,\beta}(\widehat{w}(\mathbf{x}))\ell(\boldsymbol{h}(\mathbf{x}))\mathrm{d}U(\mathbf{x})| \\
&+ \sup_{\boldsymbol{h}\in\mathcal{H}} |\gamma'\int_{\mathcal{X}} L^-_{\tau,\beta}(\widehat{w}(\mathbf{x}))\ell(\boldsymbol{h}(\mathbf{x}),\mathbf{y}_{K+1})\mathrm{d}U(\mathbf{x}) - \gamma'\widehat{R}^{\tau,\beta}_{S,T,u}(\boldsymbol{h},\mathbf{y}_{K+1})| \\
\leq& \gamma'\beta cU(\{\mathbf{x}:\ 0<r(\mathbf{x})\leq 2\tau\}) + \gamma'(c+\frac{c\beta}{\tau})O_p(\lambda_{n,m}^{\frac{1}{2}}) \\
&+c\gamma'\big(1+\frac{\beta}{\tau}+\tau+\beta\big)O_p(\lambda_{n,m}^{\frac{1}{2}}).
\end{aligned}$$

We can write

$$\begin{aligned}
\sup_{\boldsymbol{h}\in\mathcal{H}} |R_{Q,u}(\boldsymbol{h}) - \gamma'\widehat{R}^{\tau,\beta}_{S,T,u}(\boldsymbol{h})| \leq& \gamma'\beta cU(\{\mathbf{x}:\ 0<r(\mathbf{x})\leq 2\tau\}) \\
&+c\gamma'\big(1+\frac{\beta}{\tau}+\tau+\beta\big)O_p(\lambda_{n,m}^{\frac{1}{2}}).
\end{aligned}$$

□

Proof of Theorem 6. Assume that

$$\widetilde{\boldsymbol{h}} \in \arg\min_{\boldsymbol{h}\in\mathcal{H}} \widetilde{R}^{\tau,\beta}_{S,T}(\boldsymbol{h}), \quad \boldsymbol{h}_Q \in \arg\min_{\boldsymbol{h}\in\mathcal{H}} R^{\alpha}_{Q}(\boldsymbol{h}).$$

Step 1. It is easy to check that

$$\begin{aligned} R^{\alpha}_{Q}(\widetilde{\boldsymbol{h}}) - R^{\alpha}_{Q}(\boldsymbol{h}_Q) =& R^{\alpha}_{Q}(\widetilde{\boldsymbol{h}}) - \widetilde{R}^{\tau,\beta}_{S,T}(\widetilde{\boldsymbol{h}}) \\ &+ \widetilde{R}^{\tau,\beta}_{S,T}(\widetilde{\boldsymbol{h}}) - R^{\alpha}_{Q}(\boldsymbol{h}_Q) \\ \leq& R^{\alpha}_{Q}(\widetilde{\boldsymbol{h}}) - \widetilde{R}^{\tau,\beta}_{S,T}(\widetilde{\boldsymbol{h}}) \\ &+ \widetilde{R}^{\tau,\beta}_{S,T}(\boldsymbol{h}_Q) - R^{\alpha}_{Q}(\boldsymbol{h}_Q) \\ \leq& 2\sup_{\boldsymbol{h}\in\mathcal{H}} |\widetilde{R}^{\tau,\beta}_{S,T}(\boldsymbol{h}) - R^{\alpha}_{Q}(\boldsymbol{h})|, \end{aligned}$$

and

$$\begin{aligned} R^{\alpha}_{Q}(\widetilde{\boldsymbol{h}}) - R^{\alpha}_{Q}(\boldsymbol{h}_Q) =& R^{\alpha}_{Q}(\widetilde{\boldsymbol{h}}) - \widetilde{R}^{\tau,\beta}_{S,T}(\boldsymbol{h}_Q) \\ &+ \widetilde{R}^{\tau,\beta}_{S,T}(\boldsymbol{h}_Q) - R^{\alpha}_{Q}(\boldsymbol{h}_Q) \\ \geq& R^{\alpha}_{Q}(\widetilde{\boldsymbol{h}}) - \widetilde{R}^{\tau,\beta}_{S,T}(\widetilde{\boldsymbol{h}}) \\ &+ \widetilde{R}^{\tau,\beta}_{S,T}(\boldsymbol{h}_Q) - R^{\alpha}_{Q}(\boldsymbol{h}_Q) \\ \geq& -2\sup_{\boldsymbol{h}\in\mathcal{H}} |\widetilde{R}^{\tau,\beta}_{S,T}(\boldsymbol{h}) - R^{\alpha}_{Q}(\boldsymbol{h})|, \end{aligned}$$

which implies that

$$|R^{\alpha}_{Q}(\widetilde{\boldsymbol{h}}) - R^{\alpha}_{Q}(\boldsymbol{h}_Q)| \leq 2\sup_{\boldsymbol{h}\in\mathcal{H}} |\widetilde{R}^{\tau,\beta}_{S,T}(\boldsymbol{h}) - R^{\alpha}_{Q}(\boldsymbol{h})|.$$

Using the result of Lemma 29 and Lemma 27, we obtain that

$$\begin{aligned} |R^{\alpha}_{Q}(\widetilde{\boldsymbol{h}}) - R^{\alpha}_{Q}(\boldsymbol{h}_Q)| \leq& 2c\gamma'\big(1+\tau+\frac{\beta}{\tau}+\beta\big)O_p(\lambda^{\frac{1}{2}}_{n,m}) \\ &+ 2\gamma' c\alpha\beta U(0 < p/q \leq 2\tau). \end{aligned} \tag{C.4.5}$$

Then, using the result of Step 3 in the proof of Theorem 8, we obtain

$$|R^{\alpha}_{Q}(\widetilde{\boldsymbol{h}}) - R^{\alpha}_{P}(\boldsymbol{h}_Q)| \leq 2c\gamma'\big(1+\tau+\frac{\beta}{\tau}+\beta\big)O_p(\lambda^{\frac{1}{2}}_{n,m}) + 2\gamma' c\alpha\beta U(0 < p/q \leq 2\tau). \tag{C.4.6}$$

Step 2.

$$
\begin{aligned}
\widetilde{R}^{\tau,\beta}_{S,T}(\widetilde{\boldsymbol{h}}) &= (1-\alpha)\widehat{R}_S(\widetilde{\boldsymbol{h}}) + \alpha\gamma'\widehat{R}^{\tau,\beta}_{S,T,u}(\widetilde{\boldsymbol{h}}) \\
&\leq (1-\alpha)\widehat{R}_S(\boldsymbol{h}_Q) + \alpha\gamma'\widehat{R}^{\tau,\beta}_{S,T,u}(\boldsymbol{h}_Q) \\
&\leq R^{\alpha}_Q(\boldsymbol{h}_Q) \\
&+ c\gamma'\big(1+\tau+\frac{\beta}{\tau}+\beta\big)O_p(\lambda^{\frac{1}{2}}_{n,m}) + \gamma' c\alpha\beta U(0 < p/q \leq 2\tau) \\
&\text{Using Lemma 29 and Lemma 27} \\
&= (1-\alpha)\min_{\boldsymbol{h}\in\mathcal{H}} R_{Q,k}(\boldsymbol{h}) \\
&+ c\gamma'\big(1+\tau+\frac{\beta}{\tau}+\beta\big)O_p(\lambda^{\frac{1}{2}}_{n,m}) + \gamma' c\alpha\beta U(0 < p/q \leq 2\tau) \\
&\text{Using the result of Step 3 in proof of Theorem 8} \\
&\leq (1-\alpha)R_{Q,k}(\widetilde{\boldsymbol{h}}) + c\gamma'\big(1+\tau+\frac{\beta}{\tau}+\beta\big)O_p(\lambda^{\frac{1}{2}}_{n,m}) \\
&+ \gamma' c\alpha\beta U(0 < p/q \leq 2\tau).
\end{aligned}
$$

Hence,

$$
\begin{aligned}
&\alpha\gamma'\widehat{R}^{\tau,\beta}_{S,T,u}(\widetilde{\boldsymbol{h}}) \\
\leq&(1-\alpha)R_{Q,k}(\widetilde{\boldsymbol{h}}) - (1-\alpha)\widehat{R}_S(\widetilde{\boldsymbol{h}}) \\
+&c\gamma'\big(1+\tau+\frac{\beta}{\tau}+\beta\big)O_p(\lambda^{\frac{1}{2}}_{n,m}) + \gamma' c\alpha\beta U(0 < p/q \leq 2\tau) \\
=&(1-\alpha)R_{P,k}(\widetilde{\boldsymbol{h}}) - (1-\alpha)\widehat{R}_S(\widetilde{\boldsymbol{h}}) \\
+&c\gamma'\big(1+\tau+\frac{\beta}{\tau}+\beta\big)O_p(\lambda^{\frac{1}{2}}_{n,m}) + \gamma' c\alpha\beta U(0 < p/q \leq 2\tau) \\
\leq&(1-\alpha)cO_p(\lambda^{\frac{1}{2}}_{n,m}) + c\gamma'\big(1+\tau+\frac{\beta}{\tau}+\beta\big)O_p(\lambda^{\frac{1}{2}}_{n,m}) \\
+&\gamma' c\alpha\beta U(0 < p/q \leq 2\tau) \text{ Using Lemma 27} \\
\leq&2c\gamma'\big(1+\tau+\frac{\beta}{\tau}+\beta\big)O_p(\lambda^{\frac{1}{2}}_{n,m}) + \gamma' c\alpha\beta U(0 < p/q \leq 2\tau).
\end{aligned}
$$

Then, combining the above inequality with the result of Lemma 29, we obtain that

$$
\alpha R_{Q,u}(\widetilde{\boldsymbol{h}}) \leq 3c\gamma'\big(1+\tau+\frac{\beta}{\tau}+\beta\big)O_p(\lambda^{\frac{1}{2}}_{n,m}) + 2\gamma' c\alpha\beta U(0 < p/q \leq 2\tau).
$$

Step 3.

$$
\begin{aligned}
&|R_P^{\alpha}(\widetilde{\boldsymbol{h}}) - R_P^{\alpha}(\boldsymbol{h}_Q)| \\
\leq &|R_P^{\alpha}(\widetilde{\boldsymbol{h}}) - R_Q^{\alpha}(\widetilde{\boldsymbol{h}})| + |R_Q^{\alpha}(\widetilde{\boldsymbol{h}}) - R_P^{\alpha}(\boldsymbol{h}_Q)| \\
= &\alpha|R_{P,u}(\widetilde{\boldsymbol{h}}) - R_{Q,u}^{\alpha}(\widetilde{\boldsymbol{h}})| + |R_Q^{\alpha}(\widetilde{\boldsymbol{h}}) - R_P^{\alpha}(\boldsymbol{h}_Q)| \\
\leq &\alpha R_{Q,u}(\widetilde{\boldsymbol{h}}) + |R_Q^{\alpha}(\widetilde{\boldsymbol{h}}) - R_P^{\alpha}(\boldsymbol{h}_Q)| \\
\leq &5c\gamma'\big(1+\tau+\frac{\beta}{\tau}+\beta\big)O_p(\lambda_{n,m}^{\frac{1}{2}}) + 4\gamma' c\alpha\beta U(0 < p/q \leq 2\tau)
\end{aligned}
$$

Using the results of Step 1 and Step 2.

Briefly, we can write (absorbing coefficient 5 into O_p)

$$
\begin{aligned}
|R_P^{\alpha}(\widetilde{\boldsymbol{h}}) - R_P^{\alpha}(\boldsymbol{h}_Q)| \leq &c\gamma'\big(1+\tau+\frac{\beta}{\tau}+\beta\big)O_p(\lambda_{n,m}^{\frac{1}{2}}) \\
&+ 4\gamma' c\alpha\beta U(0 < p/q \leq 2\tau).
\end{aligned}
$$

Combining the above inequality with Theorem 8, we obtain that

$$
\begin{aligned}
|R_P^{\alpha}(\widetilde{\boldsymbol{h}}) - \min_{\boldsymbol{h}\in\mathcal{H}} R_P^{\alpha}(\boldsymbol{h})| \leq &c\gamma'\big(1+\tau+\frac{\beta}{\tau}+\beta\big)O_p(\lambda_{n,m}^{\frac{1}{2}}) \\
&+ 4\gamma' c\alpha\beta U(0 < p/q \leq 2\tau).
\end{aligned}
$$

□

Bibliography

[1] J Quiñonero Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D Lawrence. *Dataset Shift in Machine Learning*. MIT Press, 2009.

[2] Antonio Torralba and Alexei A. Efros. Unbiased look at dataset bias. In *Conference on Computer Vision and Pattern Recognition*, pages 1521–1528. IEEE Computer Society, 2011.

[3] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge 2007 (voc2007) results. 2007.

[4] Sinno Jialin Pan, James T Kwok, and Qiang Yang. Transfer learning via dimensionality reduction. In *Conference on Computer Association for the Advancement of Artificial Intelligence*, pages 677–682, 2008.

[5] Sinno Jialin Pan, Ivor W Tsang, James T Kwok, and Qiang Yang. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, 22(2):199–210, 2011.

[6] Mingsheng Long, Jianmin Wang, Guiguang Ding, Sinno Jialin Pan, and Philip S. Yu. Adaptation regularization: A general framework for transfer learning. *IEEE Trans. Knowl. Data Eng.*, 26(5):1076–1089, 2014.

[7] Raghuraman Gopalan, Ruonan Li, and Rama Chellappa. Domain adaptation for object recognition: An unsupervised approach. In Dimitris N. Metaxas, Long Quan, Alberto Sanfeliu, and Luc Van Gool, editors, *International Conference on Computer Vision*, pages 999–1006, 2011.

[8] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael I. Jordan. Learning transferable features with deep adaptation networks.

In *International Conference on Machine Learning*, pages 97–105, 2015.

[9] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *European Conference on Computer Vision*, Lecture Notes in Computer Science, pages 491–507. Springer, 2020.

[10] Han Guo, Ramakanth Pasunuru, and Mohit Bansal. Multi-source domain adaptation for text classification via distancenet-bandits. In *Conference on Computer Association for the Advancement of Artificial Intelligence*, pages 7830–7838. AAAI Press, 2020.

[11] Lili Mou, Zhao Meng, Rui Yan, Ge Li, Yan Xu, Lu Zhang, and Zhi Jin. How transferable are neural networks in NLP applications? In Jian Su, Xavier Carreras, and Kevin Duh, editors, *Conference on Empirical Methods in Natural Language Processing*, pages 479–489. The Association for Computational Linguistics, 2016.

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

[13] Li Zhong, Zhen Fang, Feng Liu, Jie Lu, Bo Yuan, and Guangquan Zhang. How does the combined risk affect the performance of unsupervised domain adaptation approaches? *Conference on Computer Association for the Advancement of Artificial Intelligence*, 2021.

[14] Qian Wang and Toby P. Breckon. Unsupervised domain adaptation via structured prediction based selective pseudo-labeling. In *Conference on Computer Association for the Advancement of Artificial Intelligence*, pages 6243–6250. AAAI Press, 2020.

[15] Zhen Fang, Jie Lu, Feng Liu, and Guangquan Zhang. Unsupervised domain adaptation with sphere retracting transformation. In *Inter-*

national Joint Conference on Neural Networks, pages 1–8. IEEE, 2019.

[16] Remi Tachet des Combes, Han Zhao, Yu-Xiang Wang, and Geoffrey J. Gordon. Domain adaptation with conditional distribution matching and generalized label shift. *Conference on Neural Information Processing Systems*, 2020.

[17] Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. Analysis of representations for domain adaptation. In *Conference on Neural Information Processing Systems*, pages 137–144, 2006.

[18] Ievgen Redko, Emilie Morvant, Amaury Habrard, Marc Sebban, and Younès Bennani. A survey on domain adaptation theory. *arXiv preprint arXiv:2004.11829*, 2020.

[19] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I. Jordan. Conditional adversarial domain adaptation. In *Conference on Neural Information Processing Systems*, pages 1647–1657, 2018.

[20] Jian Shen, Yanru Qu, Weinan Zhang, and Yong Yu. Wasserstein distance guided representation learning for domain adaptation. In *Conference on Computer Association for the Advancement of Artificial Intelligence*, pages 4058–4065, 2018.

[21] Muhammad Ghifary, David Balduzzi, W Bastiaan Kleijn, and Mengjie Zhang. Scatter component analysis: A unified framework for domain adaptation and domain generalization. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 39(7):1414–1430, 2017.

[22] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor S. Lempitsky. Domain-adversarial training of neural networks. *J. Mach. Learn. Res.*, 2016.

[23] Yuan Yao, Yu Zhang, Xutao Li, and Yunming Ye. Heterogeneous domain adaptation via soft transfer network. In *Multimedia Conference on Multimedia Conference*, pages 1578–1586, 2019.

[24] Wen Li, Lixin Duan, Dong Xu, and Ivor W. Tsang. Learning with augmented features for supervised and semi-supervised heterogeneous domain adaptation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 36(6):1134–1148, 2014.

[25] Feng Liu, Guangquan Zhang, and Jie Lu. Heterogeneous domain adaptation: An unsupervised approach. *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[26] Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex C. Kot. Heterogeneous domain adaptation via nonlinear matrix factorization. *IEEE Trans. Neural Networks Learn. Syst.*, 31(3):984–996, 2020.

[27] Pau Panareda Busto and Juergen Gall. Open set domain adaptation. In *International Conference on Computer Vision*, pages 754–763, 2017.

[28] Kuniaki Saito, Shohei Yamamoto, Yoshitaka Ushiku, and Tatsuya Harada. Open set domain adaptation by backpropagation. In *European Conference on Computer Vision*, pages 156–171, 2018.

[29] Oscar Day and Taghi M. Khoshgoftaar. A survey on heterogeneous transfer learning. *J. Big Data*, 4:29, 2017.

[30] Yao-Hung Hubert Tsai, Yi-Ren Yeh, and Yu-Chiang Frank Wang. Learning cross-domain landmarks for heterogeneous domain adaptation. In *Conference on Computer Vision and Pattern Recognition*, pages 5081–5090, 2016.

[31] Zhen Fang, Jie Lu, Feng Liu, Junyu Xuan, and Guangquan Zhang. Open set domain adaptation: Theoretical bound and algorithm. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–14, 2020.

[32] Walter J. Scheirer, Anderson de Rezende Rocha, Archana Sapkota, and Terrance E. Boult. Toward open set recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(7):1757–1772, 2013.

[33] Yadan Luo, Zijian Wang, Zi Huang, and Mahsa Baktashmotlagh. Progressive graph learning for open-set domain adaptation. In *International Conference on Machine Learning*, volume 119, pages 6468–6478. PMLR, 2020.

[34] Li Zhong, Zhen Fang, Feng Liu, Bo Yuan, Guangquan Zhang, and Jie Lu. Bridging the theoretical bound and deep algorithms for open set domain adaptation. *CoRR*, abs/2006.13022, 2020.

[35] Jing Zhang, Wanqing Li, and Philip Ogunbona. Joint geometrical and statistical alignment for visual domain adaptation. In *Conference on Computer Vision and Pattern Recognition*, pages 5150–5158, 2017.

[36] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67, 2020.

[37] Zaid Alyafeai, Maged Saeed AlShaibani, and Irfan Ahmad. A survey on transfer learning in natural language processing. *CoRR*, abs/2007.04239, 2020.

[38] Qian Zhang, Dianshuang Wu, Jie Lu, Feng Liu, and Guangquan Zhang. A cross-domain recommender system with consistent information transfer. *Decis. Support Syst.*, 104:49–63, 2017.

[39] Qian Zhang, Jie Lu, Dianshuang Wu, and Guangquan Zhang. A cross-domain recommender system with kernel-induced knowledge transfer for overlapping entities. *IEEE Trans. Neural Networks Learn. Syst.*, 30(7):1998–2012, 2019.

[40] Matthew E. Taylor and Peter Stone. Transfer learning for reinforcement learning domains: A survey. *J. Mach. Learn. Res.*, 10:1633–1685, 2009.

[41] Jan Ramon, Kurt Driessens, and Tom Croonenborghs. Transfer learning in reinforcement learning problems through partial policy recycling. In *European Conference on Machine Learning*, volume 4701, pages 699–707. Springer, 2007.

[42] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010.

[43] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning. *Proc. IEEE*, 109(1):43–76, 2021.

[44] Qitao Gu and Qun Dai. A novel active multi-source transfer learning algorithm for time series forecasting. *Appl. Intell.*, 51(3):1326–1350, 2021.

[45] Qiang Yang, Yuqiang Chen, Gui-Rong Xue, Wenyuan Dai, and Yong Yu. Heterogeneous transfer learning for image clustering via the socialweb. In *International Joint Conference on Natural Language Processing*, pages 1–9, 2009.

[46] Wenhao Jiang, Wei Liu, and Fulai Chung. Knowledge transfer for spectral clustering. *Pattern Recognit.*, 81:484–496, 2018.

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

[48] Yue Cao, Mingsheng Long, and Jianmin Wang. Unsupervised domain adaptation with distribution matching machines. In *Conference on Computer Association for the Advancement of Artificial Intelligence*, pages 2795–2802, 2018.

[49] Jin Gao, Haibin Ling, Weiming Hu, and Junliang Xing. Transfer learning based visual tracking with gaussian processes regression. In *European Conference on Computer Vision*, pages 188–203. Springer, 2014.

[50] Hua Zuo, Guangquan Zhang, Witold Pedrycz, Vahid Behbood, and Jie Lu. Fuzzy regression transfer learning in takagi–sugeno fuzzy models. *IEEE Transactions on Fuzzy Systems*, 25(6):1795–1807, 2016.

[51] Yabin Zhang, Bin Deng, Hui Tang, Lei Zhang, and Kui Jia. Unsupervised multi-class domain adaptation: Theory, algorithms, and practice. *CoRR*, abs/2002.08681, 2020.

[52] Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Domain adaptation: Learning bounds and algorithms. In *Conference on Learning Theory*, 2009.

[53] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine Learning*, 79(1-2):151–175, 2010.

[54] Tianyi Zhang, Ikko Yamane, Nan Lu, and Masashi Sugiyama. A one-step approach to covariate shift adaptation. In Sinno Jialin Pan and Masashi Sugiyama, editors, *Asian Conference on Machine Learning*, volume 129, pages 65–80. PMLR, 2020.

[55] Masashi Sugiyama, Matthias Krauledat, and Klaus-Robert Müller. Covariate shift adaptation by importance weighted cross validation. *J. Mach. Learn. Res.*, 8:985–1005, 2007.

[56] Jiayuan Huang, Alexander J. Smola, Arthur Gretton, Karsten M. Borgwardt, and Bernhard Schölkopf. Correcting sample selection bias by unlabeled data. In *Conference on Neural Information Processing Systems*, pages 601–608. MIT Press, 2006.

[57] Shai Shalev-Shwartz and Shai Ben-David. Understanding machine learning: From theory to algorithms. In *Cambridge university press*, 2014.

[58] Yaoliang Yu and Csaba Szepesvári. Analysis of kernel mean matching under covariate shift. In *International Conference on Machine Learning*. icml.cc / Omnipress, 2012.

[59] Masashi Sugiyama. Density ratio estimation: A new versatile tool for machine learning. In *Asian Conference on Machine Learning*, volume 5828, pages 6–9. Springer, 2009.

[60] Masashi Sugiyama, Taiji Suzuki, and Takafumi Kanamori. *Density ratio estimation in machine learning*. Cambridge University Press, 2012.

[61] Abdolnasser Sadeghkhani, Yingwei Peng, and Chunfang Devon Lin. A parametric bayesian approach in density ratio estimation. *Stats*, 2(2):189–201, 2019.

[62] Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander J. Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13:723–773, 2012.

[63] Takafumi Kanamori, Shohei Hido, and Masashi Sugiyama. A least-squares approach to direct importance estimation. *J. Mach. Learn. Res.*, 10:1391–1445, 2009.

[64] Masashi Sugiyama, Taiji Suzuki, Shinichi Nakajima, Hisashi Kashima, Paul von Bünau, and Motoaki Kawanabe. Direct importance estimation for covariate shift adaptation. *Annals of the Institute of Statistical Mathematics*, 60(4):699–746, 2008.

[65] Mingsheng Long, Jianmin Wang, Guiguang Ding, Jiaguang Sun, and Philip S Yu. Transfer feature learning with joint distribution adaptation. In *International Conference on Computer Vision*, pages 2200–2207, 2013.

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

[67] Basura Fernando, Amaury Habrard, Marc Sebban, and Tinne Tuytelaars. Unsupervised visual domain adaptation using subspace alignment. In *International Conference on Computer Vision*, pages 2960–2967, 2013.

[68] Baochen Sun, Jiashi Feng, and Kate Saenko. Return of frustratingly easy domain adaptation. In *Conference on Computer Association for the Advancement of Artificial Intelligence*, pages 2058–2065, 2016.

[69] Muhammad Ghifary, W Bastiaan Kleijn, and Mengjie Zhang. Domain adaptive neural networks for object recognition. In *Pacific Rim International Conference on Artificial Intelligence*, pages 898–904, 2014.

[70] Arthur Gretton, Dino Sejdinovic, Heiko Strathmann, Sivaraman Balakrishnan, Massimiliano Pontil, Kenji Fukumizu, and Bharath K Sriperumbudur. Optimal kernel choice for large-scale two-sample tests. In *Conference on Neural Information Processing Systems*, pages 1205–1213, 2012.

[71] Chuanqi Tan, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu. A survey on deep transfer learning. In *International Conference on Artificial Neural Networks*, volume 11141 of *Lecture Notes in Computer Science*, pages 270–279. Springer, 2018.

[72] Jiahua Dong, Yang Cong, Gan Sun, Bineng Zhong, and Xiaowei Xu. What can be transferred: Unsupervised domain adaptation for endoscopic lesions segmentation. In *Conference on Computer Vision and Pattern Recognition*, pages 4022–4031. Computer Vision Foundation / IEEE, 2020.

[73] Jiahua Dong, Yang Cong, Gan Sun, Yuyang Liu, and Xiaowei Xu. CSCL: critical semantic-consistent learning for unsupervised domain adaptation. In *European Conference on Computer Vision*, volume 12353 of *Lecture Notes in Computer Science*, pages 745–762. Springer, 2020.

[74] Jiahua Dong, Yang Cong, Gan Sun, and Dongdong Hou. Semantic-transferable weakly-supervised endoscopic lesions segmentation. In *International Conference on Computer Vision*, pages 10711–10720. IEEE, 2019.

[75] Jiahua Dong, Yang Cong, Gan Sun, Yunsheng Yang, Xiaowei Xu, and Zhengming Ding. Weakly-supervised cross-domain adaptation for endoscopic lesions segmentation. *IEEE Trans. Circuits Syst. Video Technol.*, 31(5):2020–2033, 2021.

[76] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Conference on Neural Information Processing Systems*, pages 3320–3328, 2014.

[77] Jui-Ting Huang, Jinyu Li, Dong Yu, Li Deng, and Yifan Gong. Cross-language knowledge transfer using multilingual deep neural network with shared hidden layers. In *ICASSP*, pages 7304–7308. IEEE, 2013.

[78] Maxime Oquab, Léon Bottou, Ivan Laptev, and Josef Sivic. Learning and transferring mid-level image representations using convolutional neural networks. In *Conference on Computer Vision and Pattern Recognition*, pages 1717–1724. IEEE Computer Society, 2014.

[79] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *Conference on Computer Vision, Graphics & Image Processing*, pages 722–729. IEEE Computer Society, 2008.

[80] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. Imagenet: A large-scale hierarchical image database. In *Conference on Computer Vision and Pattern Recognition*, pages 248–255. IEEE Computer Society, 2009.

[81] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Fei-Fei Li. Imagenet large scale visual recognition challenge. *Int. J. Comput. Vis.*, 115(3):211–252, 2015.

[82] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[83] Omkar M. Parkhi, Andrea Vedaldi, Andrew Zisserman, and C. V. Jawahar. Cats and dogs. In *Conference on Computer Vision and Pattern Recognition*, pages 3498–3505. IEEE Computer Society, 2012.

[84] Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. Deep domain confusion: Maximizing for domain invariance. *CoRR*, abs/1412.3474, 2014.

[85] Eric Tzeng, Judy Hoffman, Trevor Darrell, and Kate Saenko. Simultaneous deep transfer across domains and tasks. In *International Conference on Computer Vision*, pages 4068–4076. IEEE Computer Society, 2015.

[86] Zelun Luo, Yuliang Zou, Judy Hoffman, and Fei-Fei Li. Label efficient learning of transferable representations acrosss domains and tasks. In *Conference on Neural Information Processing Systems*, pages 165–177, 2017.

[87] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In *Conference on Neural Information Processing Systems*, pages 2672–2680, 2014.

[88] Garrett Wilson and Diane J. Cook. A survey of unsupervised deep domain adaptation. *ACM Trans. Intell. Syst. Technol.*, 11(5):51:1–51:46, 2020.

[89] Saeid Motiian, Marco Piccirilli, Donald A. Adjeroh, and Gianfranco Doretto. Unified deep supervised domain adaptation and generalization. In *International Conference on Computer Vision*, pages 5716–5726. IEEE Computer Society, 2017.

[90] Kuniaki Saito, Donghyun Kim, Stan Sclaroff, Trevor Darrell, and Kate Saenko. Semi-supervised domain adaptation via minimax entropy. In *International Conference on Computer Vision*, pages 8049–8057. IEEE, 2019.

[91] Jing Zhang, Zewei Ding, Wanqing Li, and Philip Ogunbona. Importance weighted adversarial nets for partial domain adaptation. In *Conference on Computer Vision and Pattern Recognition*, pages 8156–8164. IEEE Computer Society, 2018.

[92] Kaichao You, Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I. Jordan. Universal domain adaptation. In *Conference on Computer Vision and Pattern Recognition*, pages 2720–2729. Computer Vision Foundation / IEEE, 2019.

[93] Shai Ben-David, Tyler Lu, Teresa Luu, and Dávid Pál. Impossibility theorems for domain adaptation. In *International Conference on Artificial Intelligence and Statistics*, volume 9 of *JMLR Proceedings*, pages 129–136. JMLR.org, 2010.

[94] Mingsheng Long, Jianmin Wang, Guiguang Ding, Jiaguang Sun, and Philip S Yu. Transfer joint matching for unsupervised domain adaptation. In *Conference on Computer Vision and Pattern Recognition*, pages 1410–1417, 2014.

[95] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei A. Efros, and Trevor Darrell. Cycada:

Cycle-consistent adversarial domain adaptation. In *International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1994–2003. PMLR, 2018.

[96] Jindong Wang, Wenjie Feng, Yiqiang Chen, Han Yu, Meiyu Huang, and Philip S. Yu. Visual domain adaptation with manifold embedded distribution alignment. In *Multimedia Conference on Multimedia Conference*, pages 402–410, 2018.

[97] Kuniaki Saito, Yoshitaka Ushiku, and Tatsuya Harada. Asymmetric tri-training for unsupervised domain adaptation. In *International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 2988–2997. PMLR, 2017.

[98] Xiang Gu, Jian Sun, and Zongben Xu. Spherical space domain adaptation with robust pseudo-label loss. In *Conference on Computer Vision and Pattern Recognition*, pages 9098–9107. IEEE, 2020.

[99] Yuchen Zhang, Tianle Liu, Mingsheng Long, and Michael I. Jordan. Bridging theory and algorithm for domain adaptation. In *International Conference on Machine Learning*, pages 7404–7413, 2019.

[100] Brian Quanz and Jun Huan. Large margin transductive transfer learning. In *Conference on Information and Knowledge Management*, pages 1327–1336, 2009.

[101] Chen-Yu Lee, Tanmay Batra, Mohammad Haris Baig, and Daniel Ulbricht. Sliced wasserstein discrepancy for unsupervised domain adaptation. In *Conference on Computer Vision and Pattern Recognition*, pages 10285–10295. Computer Vision Foundation / IEEE, 2019.

[102] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Conference on*

Computer Vision and Pattern Recognition, pages 2962–2971. IEEE Computer Society, 2017.

[103] Hong Liu, Zhangjie Cao, Mingsheng Long, Jianmin Wang, and Qiang Yang. Separate to adapt: Open set domain adaptation via progressive separation. In *Conference on Computer Vision and Pattern Recognition*, pages 2927–2936. Computer Vision Foundation / IEEE, 2019.

[104] Mahsa Baktashmotlagh, Masoud Faraki, Tom Drummond, and Mathieu Salzmann. Learning factorized representations for open-set domain adaptation. In *The International Conference on Learning Representations*. OpenReview.net, 2019.

[105] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In Yoshua Bengio and Yann LeCun, editors, *The International Conference on Learning Representations*, 2015.

[106] Alex Graves and Navdeep Jaitly. Towards end-to-end speech recognition with recurrent neural networks. In *International Conference on Machine Learning*, pages 1764–1772. JMLR.org, 2014.

[107] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: deep neural networks with multitask learning. In *International Conference on Machine Learning*, volume 307, pages 160–167, 2008.

[108] Chuanxing Geng, Sheng-Jun Huang, and Songcan Chen. Recent advances in open set recognition: A survey. *CoRR*, abs/1811.08581, 2018.

[109] Haipeng Xiong, Hao Lu, Chengxin Liu, Liang Liu, Zhiguo Cao, and Chunhua Shen. From open set to closed set: Counting objects by spatial divide-and-conquer. In *International Conference on Computer Vision*, pages 8361–8370. IEEE, 2019.

[110] Hong-Ming Yang, Xu-Yao Zhang, Fei Yin, Qing Yang, and Cheng-Lin Liu. Convolutional prototype network for open set recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[111] Douglas de O. Cardoso, João Gama, and Felipe M. G. França. Weightless neural networks for open set recognition. *Mach. Learn.*, pages 1547–1567, 2017.

[112] Akshay Raj Dhamija, Manuel Günther, and Terrance E. Boult. Reducing network agnostophobia. In *Conference on Neural Information Processing Systems*, pages 9175–9186, 2018.

[113] Pramuditha Perera, Vlad I. Morariu, Rajiv Jain, Varun Manjunatha, Curtis Wigington, Vicente Ordonez, and Vishal M. Patel. Generative-discriminative feature representations for open-set recognition. In *Conference on Computer Vision and Pattern Recognition*, pages 11811–11820. IEEE, 2020.

[114] Abhijit Bendale and Terrance E. Boult. Towards open set deep networks. In *Conference on Computer Vision and Pattern Recognition*, pages 1563–1572. IEEE Computer Society, 2016.

[115] Zongyuan Ge, Sergey Demyanov, and Rahil Garnavi. Generative openmax for multi-class open set classification. In *British Machine Vision Conference*, 2017.

[116] Lalit P. Jain, Walter J. Scheirer, and Terrance E. Boult. Multi-class open set recognition using probability of inclusion. In *European Conference on Computer Vision*, pages 393–409, 2014.

[117] Ethan M. Rudd, Lalit P. Jain, Walter J. Scheirer, and Terrance E. Boult. The extreme value machine. *IEEE Trans. Pattern Anal. Mach. Intell.*, pages 762–768, 2018.

[118] Lawrence Neal, Matthew L. Olson, Xiaoli Z. Fern, Weng-Keen Wong, and Fuxin Li. Open set learning with counterfactual im-

ages. In *European Conference on Computer Vision*, pages 620–635. Springer, 2018.

[119] Poojan Oza and Vishal M. Patel. C2AE: class conditioned auto-encoder for open-set recognition. In *Conference on Computer Vision and Pattern Recognition*, pages 2307–2316. Computer Vision Foundation / IEEE, 2019.

[120] Chang Wang and Sridhar Mahadevan. Heterogeneous domain adaptation using manifold alignment. In Toby Walsh, editor, *International Joint Conferences on Artificial Intelligence* , pages 1541–1546. IJCAI, 2011.

[121] Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, 7:2399–2434, 2006.

[122] Yuan-Ting Hsieh, Shi-Yen Tao, Yao-Hung Hubert Tsai, Yi-Ren Yeh, and Yu-Chiang Frank Wang. Recognizing heterogeneous cross-domain data via generalized joint distribution adaptation. In *International Conference on Multimedia and Expo (ICME)*, pages 1–6, 2016.

[123] Limin Li and Zhenyue Zhang. Semi-supervised domain adaptation by covariance matching. *IEEE Trans. Pattern Anal. Mach. Intell.*, 41(11):2724–2739, 2019.

[124] Wei-Yu Chen, Tzu-Ming Harry Hsu, Yao-Hung Hubert Tsai, Yu-Chiang Frank Wang, and Ming-Syan Chen. Transfer neural trees for heterogeneous domain adaptation. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *European Conference on Computer Vision*, pages 399–414. Springer, 2016.

[125] Han Zhao, Remi Tachet des Combes, Kun Zhang, and Geoffrey J. Gordon. On learning invariant representations for domain adaptation. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors,

International Conference on Machine Learning, pages 7523–7532, 2019.

[126] James C Bezdek, Robert Ehrlich, and William Full. Fcm: The fuzzy c-means clustering algorithm. *Computers & Geosciences*, 10(2-3):191–203, 1984.

[127] Mingsheng Long, Jianmin Wang, Guiguang Ding, Dou Shen, and Qiang Yang. Transfer learning with graph co-regularization. *IEEE Trans. Knowl. Data Eng.*, 26(7):1805–1818, 2014.

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

[129] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Conference on Computer Vision and Pattern Recognition*, pages 5385–5394, 2017.

[130] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. DeCAF: A deep convolutional activation feature for generic visual recognition. In *International Conference on Machine Learning*, pages 647–655, 2014.

[131] Minmin Chen, Zhixiang Eddie Xu, Kilian Q. Weinberger, and Fei Sha. Marginalized denoising autoencoders for domain adaptation. In *International Conference on Machine Learning*, 2012.

[132] Akash Sheoran, Diptesh Kanojia, Aditya Joshi, and Pushpak Bhattacharyya. Recommendation chart of domains for cross-domain sentiment analysis: Findings of A 20 domain study. In *LREC*, pages 4982–4990. European Language Resources Association, 2020.

[133] Kevin Bascol, Rémi Emonet, and Élisa Fromont. Improving domain adaptation by source selection. In *ICIP*, pages 3043–3047. IEEE, 2019.

[134] Luyu Yang, Yogesh Balaji, Ser-Nam Lim, and Abhinav Shrivastava. Curriculum manager for source selection in multi-source domain adaptation. *CoRR*, abs/2007.01261, 2020.

[135] Dongrui Wu, Vernon J Lawhern, and Brent J Lance. Reducing offline bci calibration effort using weighted adaptation regularization with source domain selection. In *SMC*, pages 3209–3216. IEEE, 2015.

[136] Dongrui Wu, Vernon J Lawhern, and Brent J Lance. Reducing bci calibration effort in rsvp tasks using online weighted adaptation regularization with source domain selection. In *ACII*, pages 567–573. IEEE, 2015.

[137] Yitong Li, Michael Murias, Geraldine Dawson, and David E. Carlson. Extracting relationships by multi-domain matching. In *Conference on Neural Information Processing Systems*, pages 6799–6810, 2018.

[138] Han Zhao, Shanghang Zhang, Guanhang Wu, José M. F. Moura, João Paulo Costeira, and Geoffrey J. Gordon. Adversarial multiple source domain adaptation. In *Conference on Neural Information Processing Systems*, pages 8568–8579, 2018.

[139] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *International Conference on Computer Vision*, pages 1406–1415. IEEE, 2019.

[140] Sicheng Zhao, Bo Li, Xiangyu Yue, Pengfei Xu, and Kurt Keutzer. MADAN: multi-source adversarial domain aggregation network for domain adaptation. *CoRR*, abs/2003.00820, 2020.

[141] Fuzhen Zhuang, Ping Luo, Zhiyong Shen, Qing He, Yuhong Xiong, Zhongzhi Shi, and Hui Xiong. Mining distinction and commonality across multiple domains using generative model for text classification. *IEEE Trans. Knowl. Data Eng.*, 24(11):2025–2039, 2012.

[142] Fuzhen Zhuang, Ping Luo, Hui Xiong, Yuhong Xiong, Qing He, and Zhongzhi Shi. Cross-domain learning from multiple sources: A consensus regularization perspective. *IEEE Trans. Knowl. Data Eng.*, 22(12):1664–1678, 2010.

[143] Feng Liu, Jie Lu, Bo Han, Gang Niu, Guangquan Zhang, and Masashi Sugiyama. Butterfly: A panacea for all difficulties in wildly unsupervised domain adaptation. *Conference on Neural Information Processing Systems LTS Workshop*, abs/1905.07720, 2019.

[144] Yiyang Zhang, Feng Liu, Zhen Fang, Bo Yuan, Guangquan Zhang, and Jie Lu. Clarinet: A one-step approach towards budget-friendly unsupervised domain adaptation. In Christian Bessiere, editor, *International Joint Conferences on Artificial Intelligence*, pages 2526–2532, 2020.

[145] Hongyi Zhang, Moustapha Cissé, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *The International Conference on Learning Representations*, 2018.

[146] Zirui Wang, Zihang Dai, Barnabás Póczos, and Jaime G. Carbonell. Characterizing and avoiding negative transfer. In *Conference on Computer Vision and Pattern Recognition*, pages 11293–11302. Computer Vision Foundation / IEEE, 2019.

[147] Jindong Wang, Yiqiang Chen, Han Yu, Meiyu Huang, and Qiang Yang. Easy transfer learning by exploiting intra-domain structures. In *International Conference on Multimedia and Expo (ICME)*, pages 1–6, 2016.

[148] Hao Lu, Chunhua Shen, Zhiguo Cao, Yang Xiao, and Anton van den Hengel. An embarrassingly simple approach to visual domain adaptation. *IEEE Trans. Image Process.*, 27(7):3403–3417, 2018.

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

[150] Nicola Ueffing, Michel Simard, Samuel Larkin, and Howard Johnson. Nrc's PORTAGE system for WMT 2007. In *ACL workshop*, pages 185–188, 2007.

[151] Shuang Li, Shiji Song, and Gao Huang. Prediction reweighting for domain adaptation. *IEEE Trans. Neural Netw. Learning Syst.*, 28(7):1682–1695, 2017.

[152] Nikolaos Passalis and Anastasios Tefas. Unsupervised knowledge transfer using similarity embeddings. *IEEE Trans. Neural Netw. Learning Syst.*, 30(3):946–950, 2019.

[153] Feng Liu, Jie Lu, and Guangquan Zhang. Unsupervised heterogeneous domain adaptation via shared fuzzy equivalence relations. *IEEE Trans. Fuzzy Systems*, 26(6):3555–3568, 2018.

[154] Feng Liu, Guangquan Zhang, and Jie Lu. Heterogeneous domain adaptation: An unsupervised approach. *IEEE Transactions on Neural Networks and Learning Systems*, Early Access:1–15, 2020.

[155] Xingchao Peng, Ben Usman, Kuniaki Saito, Neela Kaushik, Judy Hoffman, and Kate Saenko. Syn2real: A new benchmark for synthetic-to-real visual domain adaptation. *CoRR*, abs/1806.09755, 2018.

[156] Michael T Rosenstein, Zvika Marx, Leslie Pack Kaelbling, and Thomas G Dietterich. To transfer or not to transfer. In *Conference on Neural Information Processing Systems*, pages 1–4, 2005.

[157] Mahsa Baktashmotlagh, Masoud Faraki, Tom Drummond, and Mathieu Salzmann. Learning Factorized Representations for open-set domain adaptation. In *The International Conference on Learning Representations*, 2019.

[158] Hong Liu, Zhangjie Cao, Mingsheng Long, Jianmin Wang, and Qiang Yang. Separate to adapt: Open set domain adaptation via progressive separation. In *Conference on Computer Vision and Pattern Recognition*, 2019.

[159] Hongjie Zhang, Ang Li, Xu Han, Zhaoming Chen, Yang Zhang, and Yanwen Guo. Improving open set domain adaptation using image-to-image translation. In *IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6, 2016.

[160] Vladimir Vapnik. *Statistical Learning Theory*. Wiley, 1998.

[161] Balas K. Natarajan. *Machine Learning: A Theoretical Approach*. Morgan Kaufmann, 1991.

[162] Pedro Ribeiro Mendes-Junior, Roberto Medeiros de Souza, Rafael de Oliveira Werneck, Bernardo V. Stein, Daniel V. Pazinato, Waldir R. de Almeida, Otávio A. B. Penatti, Ricardo da Silva Torres, and Anderson Rocha. Nearest neighbors distance ratio open-set classifier. *Machine Learning*, 106(3):359–386, 2017.

[163] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Conference on Neural Information Processing Systems*, pages 1106–1114, 2012.

[164] Terence Sim, Simon Baker, and Maan Bsat. The CMU pose, illumination, and expression database. *IEEE Trans. Pattern Anal. Mach. Intell.*, 25(12):1615–1618, 2003.

[165] Deng Cai, Xiaofei He, Jiawei Han, and Thomas S. Huang. Graph regularized nonnegative matrix factorization for data representation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(8):1548–1560, 2011.

[166] Terrance E. Boult, Steve Cruz, Akshay Raj Dhamija, Manuel Günther, James Henrydoss, and Walter J. Scheirer. Learning and

the unknown: Surveying steps toward open world recognition. In *Conference on Computer Association for the Advancement of Artificial Intelligence*, pages 9801–9807, 2019.

[167] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.*, 22(10):1345–1359, 2010.

[168] Jie Lu, Vahid Behbood, Peng Hao, Hua Zuo, Shan Xue, and Guangquan Zhang. Transfer learning using computational intelligence: A survey. *Knowl.-Based Syst.*, 80:14–23, 2015.

[169] Shibani Santurkar, Ludwig Schmidt, and Aleksander Madry. A classification-based study of covariate shift in GAN distributions. In *International Conference on Machine Learning*, volume 80, pages 4487–4496. PMLR, 2018.

[170] Z. Fang, Jie Lu, F. Liu, Junyu Xuan, and G. Zhang. Open set domain adaptation: Theoretical bound and algorithm. *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[171] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of Machine Learning*. 2012.

[172] Takafumi Kanamori, Taiji Suzuki, and Masashi Sugiyama. Statistical analysis of kernel-based least-squares density-ratio estimation. *Mach. Learn.*, pages 335–367, 2012.

[173] Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander J. Smola. A kernel two-sample test. *J. Mach. Learn. Res.*, pages 723–773, 2012.

[174] Corinna Cortes, Mehryar Mohri, Michael Riley, and Afshin Rostamizadeh. Sample selection bias correction theory. In *ALT*, volume 5254, pages 38–53. Springer, 2008.

[175] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *2008 eighth ieee international conference on data mining*, pages 413–422. IEEE, 2008.

[176] Ryota Yoshihashi, Wen Shao, Rei Kawakami, Shaodi You, Makoto Iida, and Takeshi Naemura. Classification-reconstruction learning for open-set recognition. In *Conference on Computer Vision and Pattern Recognition*, pages 4016–4025, 2019.

[177] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010.

[178] Simon Ager. Omniglot-writing systems and languages of the world. *Retrieved January*, 27:2008, 2008.

[179] Alex Krizhevsky and Geoff Hinton. Convolutional deep belief networks on cifar-10. *Technical report, Citeseer*, 2009.

[180] Xin Sun, Zhenning Yang, Chi Zhang, Guohao Peng, and Keck Voon Ling. Conditional gaussian distribution learning for open set recognition. *Conference on Computer Vision and Pattern Recognition*, abs/2003.08823, 2020.

[181] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.

[182] David MW Powers. Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. *arXiv preprint arXiv:2010.16061*, 2020.

[183] Ryuichi Kiryo, Gang Niu, Marthinus Christoffel du Plessis, and Masashi Sugiyama. Positive-unlabeled learning with non-negative risk estimator. In *Advances in Neural Information Processing Systems*, pages 1675–1685, 2017.

[184] Olivier Bousquet and André Elisseeff. Stability and generalization. *J. Mach. Learn. Res.*, 2:499–526, 2002.

[185] Chen Shen and Yuhong Guo. Unsupervised heterogeneous domain adaptation with sparse feature transformation. In Jun Zhu and

Ichiro Takeuchi, editors, *Asian Conference on Machine Learning*, volume 95, pages 375–390. PMLR, 2018.

[186] Sanatan Sukhija, Narayanan Chatapuram Krishnan, and Gurkanwal Singh. Supervised heterogeneous domain adaptation via random forests. In Subbarao Kambhampati, editor, *International Joint Conferences on Artificial Intelligence*, pages 2039–2045, 2016.

[187] Haoliang Li, Sinno Jialin Pan, Renjie Wan, and Alex C. Kot. Heterogeneous transfer learning via deep matrix completion with adversarial kernel embedding. In *Conference on Computer Association for the Advancement of Artificial Intelligence*, pages 8602–8609. AAAI Press, 2019.

[188] Peter Prettenhofer and Benno Stein. Cross-language text classification using structural correspondence learning. In *ACL*, pages 1118–1127. The Association for Computer Linguistics, 2010.

[189] Jingjing Li, Ke Lu, Zi Huang, Lei Zhu, and Heng Tao Shen. Heterogeneous domain adaptation through progressive alignment. *IEEE Trans. Neural Netw. Learning Syst.*, 30(5):1381–1391, 2019.

[190] Joey Tianyi Zhou, Ivor W. Tsang, Sinno Jialin Pan, and Mingkui Tan. Multi-class heterogeneous domain adaptation. *J. Mach. Learn. Res.*, 20:57:1–57:31, 2019.

[191] Brian Kulis, Kate Saenko, and Trevor Darrell. What you saw is not what you get: Domain adaptation using asymmetric kernel transforms. In *Conference on Computer Vision and Pattern Recognition*, pages 1785–1792. IEEE Computer Society, 2011.

[192] Maayan Harel and Shie Mannor. Learning from multiple outlooks. In *International Conference on Machine Learning*, pages 401–408. Omnipress, 2011.

[193] Xiaoxiao Shi, Qi Liu, Wei Fan, S Yu Philip, and Ruixin Zhu. Transfer learning on heterogeneous feature spaces via spectral transfor-

mation. In *International Conference on Data Mining*, pages 1049–1054, 2010.

[194] Min Xiao and Yuhong Guo. Semi-supervised kernel matching for domain adaptation. In *Conference on Computer Association for the Advancement of Artificial Intelligence*, 2012.

[195] John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman. Learning bounds for domain adaptation. In *Conference on Neural Information Processing Systems,2007*, pages 129–136.

[196] Maria-Florina Balcan and Avrim Blum. A pac-style model for learning from labeled and unlabeled data. In Peter Auer and Ron Meir, editors, *Conference on Learning Theory*, volume 3559, pages 111–126. Springer, 2005.

[197] Corinna Cortes, Yishay Mansour, and Mehryar Mohri. Learning bounds for importance weighting. In *Conference on Neural Information Processing Systems*, pages 442–450, 2010.

[198] Ching-Yao Chuang, Antonio Torralba, and Stefanie Jegelka. Estimating generalization under distribution shifts via domain-invariant representations. *International Conference on Machine Learning*, 2020.

[199] Zhen Fang, Jie Lu, Feng Liu, Junyu Xuan, and Guangquan Zhang. Open set domain adaptation: Theoretical bound and algorithm. *CoRR*, abs/1907.08375, 2019.

[200] B Schölkopf, R Herbrich, and AJ Smola. A generalized representer theorem computational learning theory ed d helmbold and b williamson, 2001.

[201] Nikhil Rasiwasia, Jose Costa Pereira, Emanuele Coviello, Gabriel Doyle, Gert R. G. Lanckriet, Roger Levy, and Nuno Vasconcelos. A

new approach to cross-modal multimedia retrieval. In *Multimedia Conference on Multimedia Conference*, pages 251–260. ACM, 2010.

[202] Yi-Ren Yeh, Chun-Hao Huang, and Yu-Chiang Frank Wang. Heterogeneous domain adaptation and classification by exploiting the correlation subspace. *IEEE Trans. Image Process.*, 23(5):2009–2018, 2014.

[203] Liang Zhang, Bingpeng Ma, Guorong Li, Qingming Huang, and Qi Tian. Generalized semi-supervised and structured subspace learning for cross-modal retrieval. *IEEE Trans. Multimedia*, 20(1):128–141, 2018.

[204] Massih-Reza Amini, Nicolas Usunier, and Cyril Goutte. Learning from multiple partially observed views - an application to multilingual text categorization. In *Conference on Neural Information Processing Systems*, pages 28–36, 2009.

[205] T. Kanamori, T. Suzuki, and Masashi Sugiyama. Condition number analysis of kernel-based density ratio estimation. *Technical Report TR09-0006, Department of Computer Science, Tokyo Institute of Technology*, 2009.