Semi-Supervised Heterogeneous Domain Adaptation: Theory and Algorithms

Zhen Fang, *Member, IEEE*, Jie Lu, *Fellow, IEEE*, Feng Liu, *Member, IEEE*, and Guangquan Zhang

Abstract—Semi-supervised heterogeneous domain adaptation (SsHeDA) aims to train a classifier for the target domain, in which only unlabeled and a small number of labeled data are available. This is done by leveraging knowledge acquired from a heterogeneous source domain. From algorithmic perspectives, several methods have been proposed to solve the SsHeDA problem; yet there is still no theoretical foundation to explain the nature of the SsHeDA problem or to guide new and better solutions. Motivated by compatibility condition in semi-supervised probably approximately correct (PAC) theory, we explain the SsHeDA problem by proving its generalization error – that is, why labeled heterogeneous source data and unlabeled target data help to reduce the target risk. Guided by our theory, we devise two algorithms as proof of concept. One, *kernel heterogeneous domain alignment* (KHDA), is a kernel-based algorithm; the other, *joint mean embedding alignment* (JMEA), is a neural network-based algorithm. When a dataset is small, KHDA's training time is less than JMEA's. When a dataset is large, JMEA is more accurate in the target domain. Comprehensive experiments with image/text classification tasks show KHDA to be the most accurate among all non-neural network baselines, and JMEA to be the most accurate among all baselines.

Index Terms—Transfer learning, machine learning, classification

1 Introduction

Traditional supervised learning theories [1] are based on two assumptions: 1) that the training and test data are from the same distribution [2], [3]; and 2) that *sufficient labeled training data* are available [4], [5]. To ease the above assumptions, researchers have studied the *domain adaptation* (DA) problem [6], [7], [8], [9]. In DA, there are two different domains: source and target domains, where the source domain contains sufficient labeled data (training data) and the target domain only contains a few labeled data or unlabeled data (test data). Current learning theories of DA [10], [11], [12], [13], [14], [15] show that, when the source and target domains are from *the same feature space* (i.e., *homogeneous DA* (HoDA)), DA can be solved under proper assumptions [16].

In reality, however, it is not easy to find a source domain with the same feature space as the target domain of interest [17], [18], [19], [20], [21], [22]; specifically, the source and target domains might be from different feature spaces. To track this issue, researchers have proposed a challenging problem: *semi-supervised heterogeneous DA* (SsHeDA) [23], [24], where the source and target domains have different feature spaces, while unlabeled and just a few labeled target data are available in the target domain. Though many practical SsHeDA algorithms have been proposed [25], [26], [27], [28], very little theoretical groundwork has been undertaken to reveal the nature of the SsHeDA problem or why the current solutions work as they do [14].

One of our main purposes in this paper is to develop a SsHeDA theory to explain why the labeled source and unlabeled target data can help to reduce the need for labeled target data. We first discuss whether we can simply extend the semi-supervised HoDA (SsHoDA) theory to the heterogeneous situation by introducing *feature transformations* to adapt the heterogeneous source domain and target domain. Existing SsHoDA theory [13], [29] is based on the theoretical analysis of a *weighted sum of source and target risks* (weighted risk). Researchers have provided a uniform bound on the target risk of a classifier trained to minimize the weighted risk. This has shown that the need for labeled target data can be lowered by reducing the weight of the target risk. But, an obstacle appears in the heterogeneous situation: the combined risk [13], [29], as a constant term in the SsHoDA uniform bound, becomes a function related to feature transformations. This means that the target risk estimation might be impracticable without sufficient labeled target data.

Motivated by the compatibility condition introduced by semi-supervised *probably approximately correct* (PAC) theory [30], we devise a novel theory for SsHeDA from a quite new perspective compared with previous domain adaptation theories [10], [11], [12], [13], [14], [31]. 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 data, together with a suitable compatibility condition, can reduce the need for labeled target data.

• *The authors are with Australian Artificial Intelligence Institute, University of Technology Sydney, Ultimo, NSW 2007, Australia. E-mail: {Zhen.Fang, Jie.Lu, Feng.Liu, Guangquan.Zhang}@uts.edu.au.*

Manuscript received 11 Aug. 2021; revised 23 Dec. 2021; accepted 18 Jan. 2022. Date of publication 27 Jan. 2022; date of current version 5 Dec. 2022.
This work was supported in part by Australian Research Council (ARC) under Grant FL190100149.
(Corresponding authors: Jie Lu and Feng Liu.)
Recommended for acceptance by V. Lepetit.
Digital Object Identifier no. 10.1109/TPAMI.2022.3146234

Guided by our SsHeDA theory, we devise two SsHeDA algorithms to bring the proposed SsHeDA theory to reality. *Kernel heterogeneous domain alignment* (KHDA) is a kernel method designed for small datasets. *Joint mean embedding alignment* (JMEA) is a network method designed for large-scale data. Both algorithms maintain two main branches, where the first branch aims to transfer knowledge from source to target domain, and the second branch aims to transfer knowledge from the labeled target data to the unlabeled target data.

Both algorithms are proved to have good performance in a set of experiments comprising seven representative SsHeDA baselines, 30 text classification tasks, and 74 image classification tasks. Extensive experiments demonstrate that KHDA achieves competitive performance compared with non-neural network baselines, and that JMEA achieves better performance than all of the baselines. Our contributions are summarized as follows:

1) We introduce the concepts of compatibility, transfer error rates, and uniform sample complexity as new tools for estimating the need for labeled target data. Co-opting these concepts gives a thoroughly new perspective for theoretically analyzing domain adaptation problems.
2) We propose a generalization error estimation for target risk in SsHeDA. This is the first work on SsHeDA to explain why combining labeled source data with unlabeled target data can reduce the need for labeled target data.
3) We develop two SsHeDA algorithms based on our theoretical work: KHDA and JMEA. KHDA is a kernel-based algorithm that takes less time than JMEA, when the size of datasets is small. JMEA is a neural network algorithm that is more flexible and suitable for handling massive data.

This paper is organized as follows. Section 2 reviews the current literature on SsHeDA; Section 3 introduces the problem setting and important notations; Section 4 sets out our fundamental theory of SsHeDA; Section 5 gives the details of how to design algorithms based on our theory; Sections 6 and 7 describe the KHDA and JMEA algorithms, respectively; Section 8 details our experiments; and Section 9 concludes the paper and introduces our future works.

2 Related Work

Here we briefly discuss the domain adaptation theories and representative SsHeDA algorithms.

2.1 Domain Adaptation Theory

Pioneering theoretical work was proposed by Ben-David *et al.* [10], which shows that the target risk is upper bounded by three terms: source risk, marginal distribution discrepancy, and combined risk. This learning bound has been extended from many perspectives, such as considering different loss functions [32], different distribution distances [33], [34], [35] or the PAC-Bayes framework [36], [37]. According to the survey [14], most works focus on proving tighter bounds by constructing a new distribution distance. For example, Zhang *et al.* [15] recently developed a new distribution distance termed margin disparity discrepancy.

Almost all the aforementioned works focus on the homogeneous and unsupervised situation. Only Blitzer *et al.* [13], Ben-David *et al.* [29] and Zhou *et al.* [31] investigated the semi-supervised situation. Blitzer *et al.* [13] and Ben-David *et al.* [29] mainly focused on the homogeneous situation. These works are based on the weighted sum of the source and target risks and show that, a decrease in the target weight results in a reduced need for the labeled target data. Zhou *et al.*[31] discussed the heterogeneous situation, however, Zhou's theoretical work is designed specially for their algorithm SHFR and is difficult to extend to more general situations.

2.2 SsHeDA Algorithms

The mainstream strategy to address SsHeDA is to align the source and target domains by constructing heterogeneous feature transformations [38]. Representative SsHeDA algorithms can be roughly separated into four main types: geometric/statistical alignment, instance reweighting, pseudo label strategy and feature augmentation.

• *Geometric or Statistical Alignment. Domain adaptation with manifold alignment* (DAMA) [26] and *domain adaptation by covariance matching* (DACoM) [39] utilize the manifold alignment technique [40] and covariance alignment, respectively. DAMA learns the source and target linear feature transformations to ensure that the geometry structures of the transformed domains are consistent. DACoM learns kernel/linear transformations to ensure that the transformed domains are matched with higher order moments.

• *Instance Reweighting. Cross domain landmarks selection* (CDLS) [41] does not regard data 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 data at the same time. To estimate the discrepancy between the transformed domains, CDLS utilizes the *maximum mean discrepancy* (MMD) [42].

• *Pseudo Label Strategy. Generalized joint distribution adaptation* (G-JDA) [43] and *soft transfer network* (STN) [44] both learn feature transformations to project the source and target data into a latent space, where the marginal and class-conditional distributions are matched. Lastly, the pseudo label iteration technique is used to update the target labels.

• *Feature Augmentation. Sparse heterogeneous domain adaptation* (SHFA) [23] utilizes the augmented feature transformations, which are special linear projections mapping the source and target data to a higher dimensional space. By incorporating the original features into the augmented features, SHFA enhances the similarities between domains.

In addition to these four strategies, other strategies have also been explored. *Transfer neural trees* (TNT) [24] effectively adapts domains by using the decision forest technique. *Semi-supervised entropic Gromov-Wasserstein discrepancy* (SGW) is proposed in [45] and relies on optimal transport theory.

2.3 Comparison With Existing Study

• *Theoretical Perspective.* There is only one SsHeDA theoretical work [31]. This work limits the loss function to hinge loss and the feature transformation to linear mapping. However, our work eases the restriction of the loss functions and the feature

TABLE 1
Parameters for Different Tasks

KHDA	ρ	p	σ	T
CIFAR-8	1.0	10.0	0.01	10.0
CIFAR-59	1.0	10.0	0.01	10.0
Food-101	1.0	10.0	0.01	10.0
Wikipedia	1.0	10.0	0.01	10.0
MRC	10.0	10.0	0.1	10.0
JMEA	ρ	$\rho\lambda$	r	T
CIFAR-8	0.005	0.02	100	300.0
CIFAR-59	0.001	0.002	100	300.0
Food-101	0.001	0.001	100	1000.0
Wikipedia	0.001	0.0005	100	300.0
MRC	0.001	0.001	100	300.0

transformations. Hence, our theory can be used in more general situations. Besides, the proposed generalization error in [31] does not provide an explanation as to why the labeled source and unlabeled target data can help reduce the need for labeled target data. However, providing such an explanation is our main purpose.

• *Algorithmic and Experimental Perspectives.* Compared with the SsHeDA algorithms mentioned in Section 2.2, our algorithms KHDA and JMEA are theoretical-guided. This ensures that our algorithms have good generalization ability under proper conditions. Additionally, we construct two new datasets and introduce a new dataset for validating the effectiveness of SsHeDA algorithms. These new datasets are very challenging and practical, and increase the diversity of benchmark datasets in the field of SsHeDA.

3 Problem Setting and Concepts

This section shows the problem setting and related concepts. Main notations are summarized in Table 1 of Appendix I, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPAMI.2022.3146234.

3.1 Problem Setting

Let $\mathcal{X}_s \subset \mathbb{R}^{d_1}$, $\mathcal{X}_t \subset \mathbb{R}^{d_2}$ be feature spaces and $\mathcal{Y} = \{1, \ldots, K\}$ be a label space. *Source domain* and *target domain* are two different joint distributions $P_{X_sY_s}$ and $P_{X_tY_t}$, where $X_s \in \mathcal{X}_s$, $X_t \in \mathcal{X}_t$ and $Y_s, Y_t \in \mathcal{Y}$ are random variables. Then, the SsHeDA problem is defined as follows:

Problem 1 (SsHeDA). *Given sets of samples called the* labeled source, labeled target *and* unlabeled target *data*

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

where $n_l \ll n_u$, $n_l \ll n_s$, the aim of semi-supervised heterogeneous domain adaptation is to train a classifier $g : \mathcal{X}_t \to \mathcal{Y}$ such that g can classify the unlabeled target data by using $\mathcal{S}, \mathcal{T}_l, \mathcal{T}_u$.

3.2 Concepts

• *Data.* $\mathbf{x}_s^i$, $\mathbf{x}_l^i$ and $\mathbf{x}_u^i$ represent the ith labeled source data, the ith labeled target data, and the ith unlabeled target data, respectively. We use $\{\mathbf{x}_t^i\}_{i=1}^{n_l+n_u}$ to denote the union of labeled target data and unlabeled target data: $\mathbf{x}_t^i = \mathbf{x}_l^i$, if $i \le n_l$; and $\mathbf{x}_t^i = \mathbf{x}_u^{i-n_l}$, if $i > n_l$. We also use $\{\mathbf{x}_{st}^i\}_{i=1}^{n_s+n_l+n_u}$ to denote the union of source data and target data: $\mathbf{x}_{st}^i = \mathbf{x}_s^i$, if $i \le n_s$; and $\mathbf{x}_{st}^i = \mathbf{x}_t^{i-n_s}$, if $i > n_s$.

• *Data Matrices.* For simplicity, we set $\mathbf{X}_s = [\mathbf{x}_s^1, \ldots, \mathbf{x}_s^{n_s}]$, $\mathbf{X}_l = [\mathbf{x}_l^1, \ldots, \mathbf{x}_l^{n_l}]$, $\mathbf{X}_u = [\mathbf{x}_u^1, \ldots, \mathbf{x}_u^{n_u}]$. 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 data 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 data with label c, and $n_t = n_l + n_u$. Given any data matrix $\mathbf{X}$, $\mathbf{x} \in \mathbf{X}$ means that $\mathbf{x}$ is a column vector of $\mathbf{X}$.

• *Empirical Distributions and Feature Transformations.* We denote the notation $\widehat{P}_{\mathbf{X}}$ be the corresponding empirical distribution over any data matrix $\mathbf{X} = [\mathbf{x}^1, \ldots, \mathbf{x}^n]$

$$\widehat{P}_{\mathbf{X}} = \frac{1}{n}\sum_{i=1}^{n} \delta_{\mathbf{x}^i},$$

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 $\boldsymbol{T}(\mathbf{X}) = [\boldsymbol{T}(\mathbf{x}^1), \ldots, \boldsymbol{T}(\mathbf{x}^n)]$.

• *Hypothesis Space and Risks.* In this paper, we consider a multi-class classification task with a *hypothesis space* $\mathcal{H}$ consisting of *scoring functions* (hypothesis functions)

$$\boldsymbol{h} : \ \mathcal{X} \to \mathbb{R}^{1\times|\mathcal{Y}|} = \mathbb{R}^{1\times K}$$
$$\mathbf{x} \to [h_1(\mathbf{x}), \ldots, h_K(\mathbf{x})],$$

where $h_c(\mathbf{x})$ $(c = 1, \ldots, K)$ indicates the confidence in the prediction of label c. Given $\ell : \mathbb{R}^K \times \mathbb{R}^K \to \mathbb{R}_{\ge 0}$ 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

$$R_s(\boldsymbol{h} \circ \boldsymbol{T}_s) = \mathbb{E}\ \ell(\boldsymbol{h} \circ \boldsymbol{T}_s(X_s), \boldsymbol{\phi}(Y_s)),$$
$$R_t(\boldsymbol{h} \circ \boldsymbol{T}_t) = \mathbb{E}\ \ell(\boldsymbol{h} \circ \boldsymbol{T}_t(X_t), \boldsymbol{\phi}(Y_t)),$$

where $\boldsymbol{\phi}$ maps a label to the corresponding one-hot vector. It is convenient to use notations $\widehat{R}_s(\boldsymbol{h} \circ \boldsymbol{T}_s)$ and $\widehat{R}_t(\boldsymbol{h} \circ \boldsymbol{T}_t)$ 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.

• *Domain Distance.* To estimate the discrepancy between domains, we use the following well-known measurements.

Definition 1 (Disparity Distance[15]). *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 distance $d_{\boldsymbol{h},\mathcal{H}}^{\ell}(P_1, P_2)$ between the distributions P_1 and P_2 over $\mathcal{X}$ is*

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

Compared with classical discrepancy distance [32], [34]

$$d_{\mathcal{H}}^{\ell}(P_1, P_2) = \sup_{\boldsymbol{h} \in \mathcal{H}} d_{\boldsymbol{h},\mathcal{H}}^{\ell}(P_1, P_2), \tag{1}$$

the disparity distance $d_{\boldsymbol{h},\mathcal{H}}^{\ell}(P_1, P_2)$ is tighter.

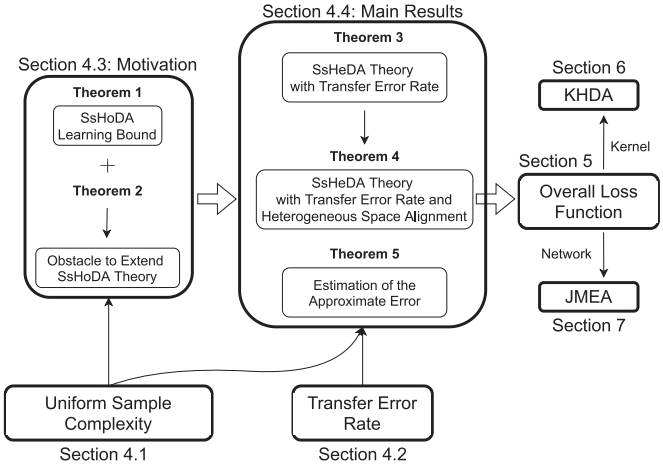


Fig. 1. The structure of our theorems and the relations between theorems and algorithms.

Definition 2 (Maximum Mean Discrepancy (MMD) [42]). *Given a feature space $\mathcal{X}$ and a class of function $\mathcal{F} \subset \{f : \mathcal{X} \to \mathbb{R}\}$, the MMD 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.* [42] propose the unit ball in a *reproducing kernel Hilbert space* (RKHS) $\mathcal{H}_k$ [34] (the subscript k represents the reproducing kernel) as the MMD function class $\mathcal{F}$.

Though the MMD distance is powerful with selected kernels [46], [47], it is not convenient to be optimized as a regularization term in shallow domain adaptation algorithms. The *projected MMD* [9], [34], [48] has been proposed to transform the MMD distance into a proper regularization term. Given a scoring function $\boldsymbol{h} = [h_1, \ldots, h_K] \in \mathcal{H}$, if $h_c \in \mathcal{H}_k, c = 1, \ldots, K$, the projected MMD is defined as follows: let $\|\cdot\|_2$ be the ℓ_2 norm, then

$$D_{\boldsymbol{h}}(P_1, P_2) = \left\| \mathop{\mathbb{E}}_{\mathbf{x} \sim P_1} \boldsymbol{h}(\mathbf{x}) - \mathop{\mathbb{E}}_{\mathbf{x} \sim P_2} \boldsymbol{h}(\mathbf{x}) \right\|_2. \tag{2}$$

4 Theoretical Foundation for SsHeDA

This section presents two novel 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. For better understanding the structure of our theorems and their connections with our algorithms, we draw Fig. 1 to illustrate the structure of the following contents of our paper.

4.1 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 data $\mathcal{S}$ 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 data (detailed definitions can be seen in Appendix III-A, available in the online supplemental material, (Definitions 7, 8)) 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 m_s^{ψ}, m_u^{ψ} and m_l^{ψ} are finite, then we say ψ has *uniform estimation*.

It is clear that if $\widetilde{\mathcal{H}} \subset \mathcal{H}$, $\widetilde{\mathcal{F}}_t \subset \mathcal{F}_t$, then

$$m_l^{\psi}(\epsilon, \delta, \widetilde{\mathcal{H}}, \mathcal{F}_s, \widetilde{\mathcal{F}}_t) \leq m_l^{\psi}(\epsilon, \delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t). \tag{3}$$

Above inequality shows that we can reduce the smallest sample number m_l^{ψ} for labeled target data by decreasing the size of the hypothesis space and target transformation space. Next, we provide an example to understand the uniform sample complexity.

Example 1. Let $\psi(\boldsymbol{h}, \boldsymbol{T}_s, \boldsymbol{T}_t, P_{X_sY_s}, P_{X_lY_l}, P_{X_t})$ be

$$R_s(\boldsymbol{h} \circ \boldsymbol{T}_s) + d^{\ell}_{\boldsymbol{h}, \mathcal{H}}(P_{\boldsymbol{T}_s(X_s)}, P_{\boldsymbol{T}_t(X_t)}). \tag{4}$$

If $\mathcal{H} \circ \mathcal{F}_s, \mathcal{H} \circ \mathcal{F}_t$ have finite Natarajan dimensions d_s and d_t [49], then $m_l^{\psi}(\epsilon, \delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t) = 0$ and

$$\begin{aligned} m_s^{\psi}(\epsilon, \delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t) &\leq C \frac{d_s \log(d_s/\epsilon) + \log(1/\delta)}{\epsilon^2}, \\ m_u^{\psi}(\epsilon, \delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t) &\leq C \frac{d_t \log(d_t/\epsilon) + \log(1/\delta)}{\epsilon^2}, \end{aligned}$$

where C is a uniform constant depending on K and loss ℓ.

Example 1 has shown that if the hypothesis space and transformation spaces satisfy appropriate conditions, Eq. (4) can be estimated by finite samples. To achieve a more accurate estimation for Eq. (4), more labeled source data and unlabeled target data are required.

4.2 Compatibility and Transfer Error Rate

Compatibility is proposed by [30] 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 data can help to reduce the need for labeled data.

To investigate how the source data and unlabeled target data reduce the need for labeled target data in the SsHeDA problem, we define the SsHeDA compatibility.

Definition 3. *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 4 (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 data $\mathcal{S} \sim P_{X_sY_s}$, $\mathcal{T}_u \sim P_{X_t}$, 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 $\mathrm{err}(\boldsymbol{h}, \boldsymbol{T}_s, \boldsymbol{T}_t)$.

Transfer error rate $\mathrm{err}(\boldsymbol{h}, \boldsymbol{T}_s, \boldsymbol{T}_t)$ aims to measure the degree of incompatibility, which is a kind of "error" that measures how unreasonable we believe some proposed hypothesis functions and feature transformations are. Then, we define the hypothesis function and target transformation whose incompatibility is at most a given value α:

Definition 5. *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 1. It is clear that $\mathcal{H}(\alpha) \subset \mathcal{H}, \mathcal{F}_t(\alpha) \subset \mathcal{F}_t$.

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 1. *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 $0 < \delta < 1$,*

$$m_s^{\mathrm{err}}(\epsilon, \delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t) < +\infty,$$
$$m_u^{\mathrm{err}}(\epsilon, \delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t) < +\infty,$$
$$m_l^{\mathrm{err}}(\epsilon, \delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t) = 0.$$

If Assumption 1 holds, the transfer error rate can be estimated by using finite source data and finite unlabeled target data. 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 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 1 (see Appendix III-B, available in the online supplemental material).

4.3 Obstacle to Extend SsHoDA Theory

Here we introduce the obstacle to extend SsHoDA theory in SsHeDA. All proofs are given in Appendix IV, available in the online supplemental material.

In SsHoDA theory [13], [29], 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 data can be reduced in homogeneous situation.

Theorem 1 (SsHoDA Learning Bound). *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 $\mathcal{X}$. Given labeled source data $\mathcal{S}$ with size n_s, labeled target data $\mathcal{T}_l$ with size n_l, and unlabeled target data $\mathcal{T}_u$ with size n_u, for any $0 < \delta < 1$, $0 < \gamma_1, \gamma_2 < 1$, and $\epsilon > 0$, if*

$$n_s > \max\Big\{ m_s^{d^{\ell}_{\mathcal{H}}}\Big(\frac{\gamma_1\epsilon}{2(1-\beta)}, \frac{\gamma_2\delta}{2}, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t\Big), m_s^{R_s}\Big(\frac{\gamma_1\epsilon}{2(1-\beta)}, \frac{\gamma_2\delta}{2}, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t\Big)\Big\},$$
$$n_u > m_u^{d^{\ell}_{\mathcal{H}}}\Big(\frac{\gamma_1\epsilon}{2(1-\beta)}, \frac{\gamma_2\delta}{2}, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t\Big),$$
$$n_l > m_l^{R_t}\Big(\frac{(1-\gamma_1)\epsilon}{2\beta}, (1-\gamma_2)\delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t\Big),$$

where $d^{\ell}_{\mathcal{H}}$ is $d^{\ell}_{\mathcal{H}}(P_{X_s}, P_{X_t})$ defined in Eq. (1) and R_s, R_t are $R_s(\boldsymbol{h}), R_t(\boldsymbol{h})$, then with a probability at least $1 - \delta > 0$

$$R_t(\widehat{\boldsymbol{h}}) \leq R_t(\boldsymbol{h}_t) + 2(1-\beta)\big(d_{\mathcal{H}}(\widehat{P}_{\mathbf{X}_s}, \widehat{P}_{\mathbf{X}_u}) + \Lambda\big) + \epsilon,$$

where $\widehat{\boldsymbol{h}} \in \arg\min_{\boldsymbol{h} \in \mathcal{H}} \widehat{R}_\beta(\boldsymbol{h})$, $\boldsymbol{h}_t \in \arg\min_{\boldsymbol{h} \in \mathcal{H}} R_t(\boldsymbol{h})$ and $\Lambda = \min_{\boldsymbol{h} \in \mathcal{H}}\big(R_s(\boldsymbol{h}) + R_t(\boldsymbol{h})\big)$ known as the combined risk.

The SsHoDA learning bound in Theorem 1 mainly contains three terms: the optimal target risk, the discrepancy distance and the combined risk. When $\beta \to 1$, the bound degenerates into the standard learning bound (that is, we use only labeled target data). Note that by choosing different values of β, the bound allows us to effectively trade off the number of labeled target data against the number of labeled source data and unlabeled target data.

Remark 2. The number of labeled target data $m_l^{R_t}(\frac{(1-\gamma_1)\epsilon}{2\beta}, (1-\gamma_2)\delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t)$ is reduced, with a decrease of β. Especially, if we set $\beta \to 0$, then $m_l^{R_t}(\frac{(1-\gamma_1)\epsilon}{2\beta}, (1-\gamma_2)\delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t) \to 0$.

It is natural 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

$$\min_{\boldsymbol{h} \in \mathcal{H}} \big(R_s(\boldsymbol{h} \circ \boldsymbol{T}_s) + R_t(\boldsymbol{h} \circ \boldsymbol{T}_t)\big). \tag{5}$$

Eq. (5) 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 data are indispensable. The following theorem provides a reason why Λ is the main obstacle.

Theorem 2. *There exist hypothesis space $\mathcal{H}$ and non-trivial transformation spaces $\mathcal{F}_s, \mathcal{F}_t$ such that for any $\epsilon > 0$ and $0 < \delta < 1$, we have $m_l^{\Lambda}(\epsilon, \delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t) \geq m_l^{R_t}(\epsilon, \delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t)$, if $|\mathcal{X}_s|, |\mathcal{X}_t| > 1$, where Λ is defined in (5) and R_t is $R_t(\boldsymbol{h} \circ \boldsymbol{T}_t)$.*

Note that there is a coefficient $2(1-\beta)$ of Λ in the bound of Theorem 1. Hence, to estimate $2(1-\beta)\Lambda$ given ϵ and δ, the number of labeled target data needed is at least

$$m_l^{\Lambda}(\frac{\epsilon}{2(1-\beta)}, \delta, \mathcal{H}, \mathcal{F}_s, \mathcal{F}_t). \tag{6}$$

Combining Theorems 1 and 2 with Eq. (6), we know that when $|\mathcal{X}_s|, |\mathcal{X}_t| > 1$, there exist hypothesis space $\mathcal{H}$ and non-trivial transformation spaces $\mathcal{F}_s, \mathcal{F}_t$ such that the number of labeled target data n_l is at least

$$\max\Big\{m_l^{R_t}\Big(\frac{\epsilon}{2\beta},\delta,\mathcal{H},\mathcal{F}_s,\mathcal{F}_t\Big),m_l^{R_t}\Big(\frac{\epsilon}{2(1-\beta)},\delta,\mathcal{H},\mathcal{F}_s,\mathcal{F}_t\Big)\Big\},$$

to obtain a similar bound in Theorem 1. Hence, the number n_l is greater than or equal to $m_l^{R_t}(\epsilon,\delta,\mathcal{H},\mathcal{F}_s,\mathcal{F}_t)$ (detailed discussion can be found in Appendix IV-D, available in the online supplemental material).

Remark 3. According to Theorem 1, in homogeneous situation, for any hypothesis $\mathcal{H}$, when $\beta\to 0$, the need for labeled target data is close to 0. However, in heterogeneous situation, there exist hypothesis $\mathcal{H}$ and non-trivial transformation spaces $\mathcal{F}_s,\mathcal{F}_t$ such that for any weight $\beta\in(0,1)$, the number of labeled target data may be larger than $m_l^{R_t}(\epsilon,\delta,\mathcal{H},\mathcal{F}_s,\mathcal{F}_t)$.

4.4 Theoretical Analysis

We start this section from a basic theorem, which contains our main idea about SsHeDA theory. More extensive discussions and all proofs are given in Appendix V, available in the online supplemental material.

Theorem 3. *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 1. Given labeled source data $\mathcal{S}$ with size n_s, labeled target data $\mathcal{T}_l$ with size n_l, and unlabeled target data $\mathcal{T}_u$ with size n_u, 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 $\min_{\boldsymbol{h}^*\in\mathcal{H},\boldsymbol{T}_s^*\in\mathcal{F}_s}\widehat{\mathrm{err}}(\boldsymbol{h}^*,\boldsymbol{T}_s^*,\boldsymbol{T}_t)<\alpha$

$$|R_t(\boldsymbol{h}\circ\boldsymbol{T}_t)-\widehat{R}_t(\boldsymbol{h}\circ\boldsymbol{T}_t)|<\epsilon.$$

Observing Theorem 3, when γ_1 is close to 0, then the number of labeled target data 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 data, we 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 data 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 3 provides an explanation of the uniform sample complexity for the labeled target data, we still cannot explain the representative algorithms [43], [44] by constructing different transfer error rates. This is because the transfer error rate is not related to labeled target data that can be used to help align the heterogeneous spaces and *control the approximate error*. Hence, we add an additional constraint called the heterogeneous space alignment

$$d(\boldsymbol{h},\boldsymbol{T}_s,\boldsymbol{T}_t),\tag{7}$$

which is able to be estimated by labeled source data and labeled target data. Motivated by previous work [11], [44], [50], we can set the heterogeneous space alignment as

- class-conditional distribution alignment

$$\sum_{c=1}^{K}d_{\boldsymbol{h},\mathcal{H}}^{\ell}(P_{\boldsymbol{T}_t(X_t)|Y_t=c},P_{\boldsymbol{T}_s(X_s)|Y_s=c}),\tag{8}$$

- projected MMD alignment for class

$$\sum_{c=1}^{K}D_{\boldsymbol{h}}^2(P_{\boldsymbol{T}_t(X_t)|Y_t=c},P_{\boldsymbol{T}_s(X_s)|Y_s=c}),\tag{9}$$

- MMD alignment for class

$$\sum_{c=1}^{K}D_{\mathcal{F}}^2(P_{\boldsymbol{T}_t(X_t)|Y_t=c},P_{\boldsymbol{T}_s(X_s)|Y_s=c}).\tag{10}$$

Theorem 4. *Given the same conditions and assumption in Theorem 3, 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 $\min_{\boldsymbol{h}^*\in\mathcal{H},\boldsymbol{T}_s^*\in\mathcal{F}_s}\big(\widehat{\mathrm{err}}(\boldsymbol{h}^*,\boldsymbol{T}_s^*,\boldsymbol{T}_t)+\widehat{d}(\boldsymbol{h}^*,\boldsymbol{T}_s^*,\boldsymbol{T}_t)\big)<\alpha$

$$|R_t(\boldsymbol{h}\circ\boldsymbol{T}_t)-\widehat{R}_t(\boldsymbol{h}\circ\boldsymbol{T}_t)|<\epsilon,$$

where $\widehat{d}(\boldsymbol{h},\boldsymbol{T}_s,\boldsymbol{T}_t)$ is the empirical form of heterogeneous space alignment defined in Eqs. (8), (9) or (10).

Theorem 4 is an extension of Theorem 3. By introducing the heterogeneous space alignment in Theorem 4, we can align the heterogeneous space between the source and target domains better. Using Theorem 4, we can provide an explanation for representative algorithms, such as STN [44]. Reducing the space size implies that the estimation error decreases and approximate error may increase. To estimate the approximate error, Theorem 5 provides an answer.

Theorem 5. *Let ℓ be the loss satisfying the triangle inequality. If $\mathrm{err}(\boldsymbol{h},\boldsymbol{T}_s,\boldsymbol{T}_t)=\frac{1}{B}\big(R_s(\boldsymbol{h}\circ\boldsymbol{T}_s)+d_{\boldsymbol{h},\mathcal{H}}^{\ell}(P_{\boldsymbol{T}_s(X_s)},P_{\boldsymbol{T}_t(X_t)})\big)$, $d(\boldsymbol{h},\boldsymbol{T}_s,\boldsymbol{T}_t)=\frac{1}{B}\Lambda$, where $\Lambda=\min_{\boldsymbol{h}\in\mathcal{H}}\big(R_s(\boldsymbol{h}\circ\boldsymbol{T}_s)+R_t(\boldsymbol{h}\circ\boldsymbol{T}_t)\big)$, $B/2$ is the supremum of ℓ, then the approximate error*

$$\min_{\boldsymbol{h}\in\mathcal{H}(\alpha_1),\boldsymbol{T}_t\in\mathcal{F}_t(\alpha_2)}R_t(\boldsymbol{h}\circ\boldsymbol{T}_t)\le B\alpha_{\min},$$

for $\alpha_1,\alpha_2>\alpha_{\min}$, where

$$\alpha_{\min}=\min_{\boldsymbol{h}\in\mathcal{H},\boldsymbol{T}_s\in\mathcal{F}_s,\boldsymbol{T}_t\in\mathcal{F}_t}\big(\mathrm{err}(\boldsymbol{h},\boldsymbol{T}_s,\boldsymbol{T}_t)+d(\boldsymbol{h},\boldsymbol{T}_s,\boldsymbol{T}_t)\big).$$

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 (see Appendix VI-B, available in the online supplemental material), hence, it can be used to align heterogeneous spaces. In addition, if we require

$$R_t(\boldsymbol{h}\circ\boldsymbol{T}_t)\le C\big(\mathrm{err}(\boldsymbol{h},\boldsymbol{T}_s,\boldsymbol{T}_t)+d(\boldsymbol{h},\boldsymbol{T}_s,\boldsymbol{T}_t)\big),$$

we can also obtain a result similar to the above theorem. Furthermore, we provide an explanation for representative algorithm STN [44] using our theory (see Appendix VI-C, available in the online supplemental material).

5 Bringing SsHeDA Theory Into Reality

This section shows how to design a loss function according to Theorem 4, i.e., bringing Theorem 4 into the reality. As discussed in Theorem 4, we should consider the optimization problem as follows:

$$\min_{\boldsymbol{h}\in\mathcal{H},\boldsymbol{T}_t\in\mathcal{F}_t} \widehat{R}_t(\boldsymbol{h}\circ\boldsymbol{T}_t), \quad \textbf{subject to}$$
$$\min_{\boldsymbol{h}^*\in\mathcal{H},\boldsymbol{T}_s^*\in\mathcal{F}_s} \big(\widehat{\mathrm{err}}(\boldsymbol{h}^*,\boldsymbol{T}_s^*,\boldsymbol{T}_t)+\widehat{d}(\boldsymbol{h}^*,\boldsymbol{T}_s^*,\boldsymbol{T}_t)\big)\le\alpha. \tag{11}$$

However, such constraint optimization problem cannot be easily solved. Following [51], we replace the constraint in problem (11) as a penalty and have the revised problem

$$\min_{\boldsymbol{h},\boldsymbol{h}^*\in\mathcal{H},\boldsymbol{T}_s\in\mathcal{F}_s,\boldsymbol{T}_t\in\mathcal{F}_t} \widehat{R}_t(\boldsymbol{h}\circ\boldsymbol{T}_t)+\lambda\widehat{\mathrm{err}}(\boldsymbol{h}^*,\boldsymbol{T}_s,\boldsymbol{T}_t) +\lambda\widehat{d}(\boldsymbol{h}^*,\boldsymbol{T}_s,\boldsymbol{T}_t), \tag{12}$$

where λ ($\lambda>0$) is a free parameter. 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 [50], [52], in our kernel-based algorithm

$$\mathrm{err}(\boldsymbol{h},\boldsymbol{T}_s,\boldsymbol{T}_t)=R_s(\boldsymbol{h}\circ\boldsymbol{T}_s)+\rho D^2_{\boldsymbol{h}}(P_{\boldsymbol{T}_t(X_t)},P_{\boldsymbol{T}_s(X_s)}),$$
$$d(\boldsymbol{h},\boldsymbol{T}_s,\boldsymbol{T}_t)=\rho\sum_{c=1}^{K}D^2_{\boldsymbol{h}}(P_{\boldsymbol{T}_t(X_t)|Y_t=c},P_{\boldsymbol{T}_s(X_s)|Y_s=c}). \tag{13}$$

Motivated by [44], in our neural network-based algorithm

$$\mathrm{err}(\boldsymbol{h},\boldsymbol{T}_s,\boldsymbol{T}_t)=R_s(\boldsymbol{h}\circ\boldsymbol{T}_s)+\rho D^2_{\mathcal{F}}(P_{\boldsymbol{T}_t(X_t)},P_{\boldsymbol{T}_s(X_s)}),$$
$$d(\boldsymbol{h},\boldsymbol{T}_s,\boldsymbol{T}_t)=\rho\sum_{c=1}^{K}D^2_{\mathcal{F}}(P_{\boldsymbol{T}_t(X_t)|Y_t=c},P_{\boldsymbol{T}_s(X_s)|Y_s=c}), \tag{14}$$

where ρ ($\rho>0$) is free parameter, and $\mathcal{F}$ is the unit ball of linear kernel Hilbert space. Besides, we omit the coefficient that ensures the transfer error rate is not larger than 1.

• *Target Distribution Alignment*.Note that the selection bias may exist [53], [54], since the number of labeled target data might be small. Thus, to mitigate the selection bias, the target distribution alignment $\widehat{d}_t(\boldsymbol{h},\boldsymbol{T}_t,\boldsymbol{T}_t)$ is considered. An analysis for the target distribution alignment is in Appendix VI-D, available in the online supplemental material. In our kernel-based algorithm, we set $\widehat{d}_t(\boldsymbol{h},\boldsymbol{T}_t,\boldsymbol{T}_t)$ to

$$D^2_{\boldsymbol{h}}(\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_l)},\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_u)})+\sum_{c=1}^{K}D^2_{\boldsymbol{h}}(\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_l^c)},\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_u^{\bar{c}})}). \tag{15}$$

In our neural network-based algorithm, $\widehat{d}_t(\boldsymbol{h},\boldsymbol{T}_t,\boldsymbol{T}_t)$ is

$$D^2_{\mathcal{F}}(\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_l)},\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_u)})+\sum_{c=1}^{K}D^2_{\mathcal{F}}(\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_l^c)},\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_u^{\bar{c}})}), \tag{16}$$

where $\mathbf{X}_u^{\bar{c}}$ is the unlabeled data matrix with pseudo label c.

• *Overall Loss Function*. Inspired by the above discussions, to solve the SsHeDA problem well, we need to take care of the following optimization problem

$$\min_{\boldsymbol{h},\boldsymbol{h}^*\in\mathcal{H},\boldsymbol{T}_s\in\mathcal{F}_s,\boldsymbol{T}_t\in\mathcal{F}_t}\mathcal{L}(\boldsymbol{h},\boldsymbol{h}^*,\boldsymbol{T}_s,\boldsymbol{T}_t),\quad \textbf{where} \tag{17}$$

$$\mathcal{L}(\boldsymbol{h},\boldsymbol{h}^*,\boldsymbol{T}_s,\boldsymbol{T}_t)=\widehat{R}_t(\boldsymbol{h}\circ\boldsymbol{T}_t)+\lambda\widehat{R}_t(\boldsymbol{h}^*\circ\boldsymbol{T}_t)+\lambda\big(\widehat{d}(\boldsymbol{h}^*,\boldsymbol{T}_s,\boldsymbol{T}_t) +\widehat{\mathrm{err}}(\boldsymbol{h},\boldsymbol{T}_s,\boldsymbol{T}_t)\big)+\rho\widehat{d}_t(\boldsymbol{h},\boldsymbol{T}_t,\boldsymbol{T}_t), \tag{18}$$

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 Eqs. (13) and (14), and $\widehat{d}_t(\boldsymbol{h},\boldsymbol{T}_t,\boldsymbol{T}_t)$ is defined in Eqs. (15) or (16). Note that we have added $\widehat{R}_t(\boldsymbol{h}^*\circ\boldsymbol{T}_t)$ in Eq. (18), because we need to guarantee that the labeled target data can be classified accurately.

6 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 (17) are defined as follows:

$$\mathcal{F}_s=\{[T_1,\ldots,T_c,\ldots,T_d]\mid T_c\in\mathcal{H}_{k_s}\},$$
$$\mathcal{F}_t=\{[T_1,\ldots,T_c,\ldots,T_d]\mid T_c\in\mathcal{H}_{k_t}\},$$
$$\mathcal{H}=\{[h_1,\ldots,h_c,\ldots,h_K]\mid h_c(\mathbf{x})=\mathbf{x}\boldsymbol{a}^\top,\ \boldsymbol{a},\mathbf{x}\in\mathbb{R}^{1\times d}\},$$

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 $h_c\circ\boldsymbol{T}_t\in\mathcal{H}_{k_t}$.

6.1 Loss Function in KHDA

As introduced in Eq. (13), 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. (13), the heterogeneous space alignment $d(\boldsymbol{h},\boldsymbol{T}_s,\boldsymbol{T}_t)$ is set to 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 [50], [55], the pseudo labels are used to further improve the classification performance, hence, we replace above equation by

$$\sum_{c=1}^{K}D^2_{\boldsymbol{h}^*}(\widehat{P}_{\boldsymbol{T}_s(\mathbf{X}_s^c)},\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_t^{\bar{c}})}), \tag{19}$$

where $\mathbf{X}_t^{\bar{c}}=[\mathbf{X}_l^c,\mathbf{X}_u^{\bar{c}}]$, here $\mathbf{X}_u^{\bar{c}}$ is unlabeled data matrix with pseudo label c.

Then, to preserve domains' geometry structures, such as manifold structure and clustering structure, manifold regularization [40] is considered in KHDA. Many kernel-based DA algorithms [50], [52], [56] have studied the manifold regularization and shown that it can 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:

$$\sum_{\mathbf{x},\mathbf{x}'\in\mathbf{X}_s \text{ or } \mathbf{X}_t} \Big\|\boldsymbol{h}^*\circ\boldsymbol{T}(\mathbf{x})-\boldsymbol{h}^*\circ\boldsymbol{T}(\mathbf{x}')\Big\|_2^2\mathbf{W}^*(\mathbf{x},\mathbf{x}'),$$
$$\sum_{\mathbf{x},\mathbf{x}'\in\mathbf{X}_t} \Big\|\boldsymbol{h}\circ\boldsymbol{T}_t(\mathbf{x})-\boldsymbol{h}\circ\boldsymbol{T}_t(\mathbf{x}')\Big\|_2^2\mathbf{W}(\mathbf{x},\mathbf{x}'),$$

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 $\mathbf{x},\mathbf{x}'$. Additionally, when $\mathbf{x}$ and $\mathbf{x}'$ are from different domains, we set $\mathbf{W}^*(\mathbf{x},\mathbf{x}')=0$.

Summarizing the above discussion, the loss function (18) can be rewritten as follows:

$$\begin{aligned}&\widehat{R}_t(\boldsymbol{h}\circ\boldsymbol{T}_t)+\sigma\|\boldsymbol{h}\circ\boldsymbol{T}_s\|_s^2+\sigma\|\boldsymbol{h}\circ\boldsymbol{T}_t\|_t^2\\&+\rho\big(\widehat{d}_t(\boldsymbol{h},\boldsymbol{T}_t,\boldsymbol{T}_t)+\widehat{M}_2(\boldsymbol{h},\boldsymbol{T}_t,\boldsymbol{T}_t)\big)\\&+\lambda\widehat{R}_s(\boldsymbol{h}^*\circ\boldsymbol{T}_s)+\lambda\widehat{R}_t(\boldsymbol{h}^*\circ\boldsymbol{T}_t)+\lambda\sigma\|\boldsymbol{h}^*\circ\boldsymbol{T}_s\|_s^2\\&+\lambda\rho\big(\widehat{d}_s(\boldsymbol{h}^*,\boldsymbol{T}_s,\boldsymbol{T}_t)+\widehat{M}_1(\boldsymbol{h}^*,\boldsymbol{T}_s,\boldsymbol{T}_t)\big)+\lambda\sigma\|\boldsymbol{h}^*\circ\boldsymbol{T}_t\|_t^2,\end{aligned}\tag{20}$$

where $\widehat{d}_s(\boldsymbol{h}^*,\boldsymbol{T}_s,\boldsymbol{T}_t)$ and $\widehat{d}_t(\boldsymbol{h},\boldsymbol{T}_t,\boldsymbol{T}_t)$ are

$$\begin{aligned}&D^2_{\boldsymbol{h}^*}(\widehat{P}_{\boldsymbol{T}_s(\mathbf{X}_s)},\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_t)})+\sum_{c=1}^{K}D^2_{\boldsymbol{h}^*}(\widehat{P}_{\boldsymbol{T}_s(\mathbf{X}_s^c)},\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_t^{\bar{c}})}),\\&D^2_{\boldsymbol{h}}(\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_l)},\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_u)})+\sum_{c=1}^{K}D^2_{\boldsymbol{h}}(\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_l^c)},\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_u^{\bar{c}})}),\end{aligned}\tag{21}$$

respectively, $\|\cdot\|_s^2$, $\|\cdot\|_t^2$ 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.

6.2 Reformulation of the KHDA Loss Function

This section shows how to reformulate Eq. (20). Following the representer theorem [57], $\boldsymbol{T}_s$ and $\boldsymbol{T}_t$ can be written as

$$\boldsymbol{T}_s(\mathbf{x})=\sum_{i=1}^{n_s}\boldsymbol{\alpha}^i k_s(\mathbf{x},\mathbf{x}_s^i),\quad\forall\mathbf{x}\in\mathcal{X}_s,$$
$$\boldsymbol{T}_t(\mathbf{x})=\sum_{i=1}^{n_t}\boldsymbol{\beta}^i k_t(\mathbf{x},\mathbf{x}_t^i),\quad\forall\mathbf{x}\in\mathcal{X}_t,$$

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

$$\mathbf{K}=\begin{bmatrix}\mathbf{K}_{ss}&\mathbf{O}\\\mathbf{O}&\mathbf{K}_{tt}\end{bmatrix}\in\mathbb{R}^{(n_s+n_t)\times(n_s+n_t)},\tag{22}$$

where $\mathbf{K}_{ss}=[k_s(\mathbf{x}_s^i,\mathbf{x}_s^j)]\in\mathbb{R}^{n_s\times n_s}$, $\mathbf{K}_{tt}=[k_t(\mathbf{x}_t^i,\mathbf{x}_t^j)]\in\mathbb{R}^{n_t\times n_t}$ 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.

• *Empirical Risks*. Here we will use a matrix to rewrite the following equation:

$$\begin{aligned}&\lambda\widehat{R}_s(\boldsymbol{h}^*\circ\boldsymbol{T}_s)+\lambda\widehat{R}_t(\boldsymbol{h}^*\circ\boldsymbol{T}_t)+\widehat{R}_t(\boldsymbol{h}\circ\boldsymbol{T}_t)\\&+\sigma\|\boldsymbol{h}\circ\boldsymbol{T}_s\|_s^2+\sigma\|\boldsymbol{h}\circ\boldsymbol{T}_t\|_t^2+\sigma\lambda\|\boldsymbol{h}^*\circ\boldsymbol{T}_s\|_s^2+\sigma\lambda\|\boldsymbol{h}^*\circ\boldsymbol{T}_t\|_t^2.\end{aligned}\tag{23}$$

Let the label matrix be $\mathbf{Y}\in\mathbb{R}^{(n_s+n_t)\times K}$

$$\mathbf{Y}_{ic}=\begin{cases}1, & \mathbf{x}_{st}^i\in\mathbf{X}_s^c \text{ or } \mathbf{X}_l^c,\\0, & \text{otherwise.}\end{cases}\tag{24}$$

Then, Eq. (23) can be written as (The details about how to obtain Eq. (25) from Eq. (23) are shown in Appendix VII-D, available in the online supplemental material)

$$\begin{aligned}&\|\mathbf{A}(\mathbf{Y}-\mathbf{K}\boldsymbol{\Theta}\boldsymbol{\Gamma})\|_F^2+\lambda\|\mathbf{A}^*(\mathbf{Y}-\mathbf{K}\boldsymbol{\Theta}\boldsymbol{\Gamma}^*)\|_F^2\\&\quad+\ \sigma\mathrm{tr}((\boldsymbol{\Theta}\boldsymbol{\Gamma})^\top\mathbf{K}\boldsymbol{\Theta}\boldsymbol{\Gamma})+\sigma\lambda\mathrm{tr}((\boldsymbol{\Theta}\boldsymbol{\Gamma}^*)^\top\mathbf{K}\boldsymbol{\Theta}\boldsymbol{\Gamma}^*)\\&=\ \mathrm{tr}\big((\boldsymbol{\Theta}\boldsymbol{\Gamma})^\top(\mathbf{K}\mathbf{A}^2\mathbf{K}+\sigma\mathbf{K})\boldsymbol{\Theta}\boldsymbol{\Gamma}\big)-2\mathrm{tr}\big(\mathbf{Y}^\top\mathbf{A}^2\mathbf{K}\boldsymbol{\Theta}\boldsymbol{\Gamma}\big)\\&\quad+\ \lambda\mathrm{tr}\big((\boldsymbol{\Theta}\boldsymbol{\Gamma}^*)^\top(\mathbf{K}\mathbf{A}^{*2}\mathbf{K}+\sigma\mathbf{K})\boldsymbol{\Theta}\boldsymbol{\Gamma}^*\big)-2\lambda\mathrm{tr}\big(\mathbf{Y}^\top\mathbf{A}^{*2}\mathbf{K}\boldsymbol{\Theta}\boldsymbol{\Gamma}^*\big)\\&\quad+\ \mathrm{Constant},\end{aligned}\tag{25}$$

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 $\mathbf{x}_{st}^i\in\mathbf{X}_l$, 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 $\mathbf{x}_{st}^i\in\mathbf{X}_s$, $\mathbf{A}^*_{ii}=\sqrt{\frac{1}{n_l}}$, if $\mathbf{x}_{st}^i\in\mathbf{X}_l$, otherwise $\mathbf{A}^*_{ii}=0$; and $\|\cdot\|_F$ is the Frobenius norm.

• *Distribution Alignment*. Using the representer theorem [57] and kernel trick [52], we rewrite Eq. (21) as

$$\begin{aligned}\widehat{d}_s(\boldsymbol{h}^*,\boldsymbol{T}_s,\boldsymbol{T}_t)&=\mathrm{tr}\big((\boldsymbol{\Theta}\boldsymbol{\Gamma}^*)^\top\mathbf{K}\mathbf{M}^*\mathbf{K}\boldsymbol{\Theta}\boldsymbol{\Gamma}^*\big),\\\widehat{d}_t(\boldsymbol{h},\boldsymbol{T}_t,\boldsymbol{T}_t)&=\mathrm{tr}\big((\boldsymbol{\Theta}\boldsymbol{\Gamma})^\top\mathbf{K}\mathbf{M}\mathbf{K}\boldsymbol{\Theta}\boldsymbol{\Gamma}\big),\end{aligned}\tag{26}$$

where $\mathbf{M}$ and $\mathbf{M}^*$ are MMD matrices defined in Appendix I-G, available in the online supplemental material. The details about how to derive Eq. (26) from Eq. (21) are shown in Appendix VII-D, available in the online supplemental material.

• *Manifold Regularization*. The pair-wise source affinity matrix $\mathbf{W}_s$ is denoted as

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

where $sim(\mathbf{x}_s^i,\mathbf{x}_s^j)$ is the similarity function such as cosine similarity, $\mathcal{N}_p(\mathbf{x}_s^i)$ denotes the set of p-nearest neighbors to $\mathbf{x}_s^i$ and p is a free parameter.

The pair-wise target affinity matrix $\mathbf{W}_t$ is denoted as

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

Using $\mathbf{W}_s$ and $\mathbf{W}_t$, we have

$$\mathbf{W} = \begin{bmatrix} \mathbf{O} & \mathbf{O} \\ \mathbf{O} & \mathbf{W}_t \end{bmatrix} \in \mathbb{R}^{(n_s+n_t)\times(n_s+n_t)},$$
$$\mathbf{W}^* = \begin{bmatrix} \mathbf{W}_s & \mathbf{O} \\ \mathbf{O} & \mathbf{W}_t \end{bmatrix} \in \mathbb{R}^{(n_s+n_t)\times(n_s+n_t)}.$$

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

$$\mathrm{tr}\big((\boldsymbol{\Theta}\boldsymbol{\Gamma}^*)^\top \mathbf{K}\mathbf{L}^*\mathbf{K}\boldsymbol{\Theta}\boldsymbol{\Gamma}^*\big), \quad \mathrm{tr}\big((\boldsymbol{\Theta}\boldsymbol{\Gamma})^\top \mathbf{K}\mathbf{L}\mathbf{K}\boldsymbol{\Theta}\boldsymbol{\Gamma}\big), \tag{27}$$

where $\mathbf{L}^*$ and $\mathbf{L}$ are the Laplacian matrices, which can be written as $\mathbf{D}^* - \mathbf{W}^*$ and $\mathbf{D} - \mathbf{W}$. Here $\mathbf{D}^*$, $\mathbf{D}$ are diagonal matrices with $\mathbf{D}^*_{ii} = \sum_{j=1}^{n_s+n_t} \mathbf{W}^*_{ij}$, $\mathbf{D}_{ii} = \sum_{j=1}^{n_s+n_t} \mathbf{W}_{ij}$.

• *Overall Reformulation*.Combining Eqs. (25), (26), (27) with Eq. (20), the optimization problem is written as

$$\boldsymbol{\Gamma}, \boldsymbol{\Gamma}^*, \boldsymbol{\Theta} = \underset{\boldsymbol{\Gamma},\boldsymbol{\Gamma}^*\in\mathbb{R}^{d\times K}, \boldsymbol{\Theta}\in\mathbb{R}^{(n_s+n_t)\times d}}{\arg\min} \mathcal{L}(\boldsymbol{\Gamma}, \boldsymbol{\Gamma}^*, \boldsymbol{\Theta}), \tag{28}$$

where $\mathcal{L}(\boldsymbol{\Gamma}, \boldsymbol{\Gamma}^*, \boldsymbol{\Theta})$ is

$$\begin{aligned} &\mathrm{tr}\big((\boldsymbol{\Theta}\boldsymbol{\Gamma})^\top(\mathbf{K}(\mathbf{A}^2 + \rho\mathbf{M} + \rho\mathbf{L})\mathbf{K} + \sigma\mathbf{K})\boldsymbol{\Theta}\boldsymbol{\Gamma}\big) \\ &+ \lambda\mathrm{tr}\big((\boldsymbol{\Theta}\boldsymbol{\Gamma}^*)^\top(\mathbf{K}(\mathbf{A}^{*2} + \rho\mathbf{M}^* + \rho\mathbf{L}^*)\mathbf{K} + \sigma\mathbf{K})\boldsymbol{\Theta}\boldsymbol{\Gamma}^*\big) \\ &- 2\mathrm{tr}\big(\mathbf{Y}^\top\mathbf{A}^2\mathbf{K}\boldsymbol{\Theta}\boldsymbol{\Gamma}\big) - 2\lambda\mathrm{tr}\big(\mathbf{Y}^\top\mathbf{A}^{*2}\mathbf{K}\boldsymbol{\Theta}\boldsymbol{\Gamma}^*\big). \end{aligned} \tag{29}$$

6.3 Analytical Solution

We theoretically analyse the optimization problem (28) and the following theorem tells us that the optimization problem (28) has countless solutions. All proofs are in Appendix VI, available in the online supplemental material.

Theorem 6. *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. (29).

Although the optimization problem (28) has countless solutions, we are only interested in $\boldsymbol{\Theta}\boldsymbol{\Gamma}$ and $\boldsymbol{\Theta}\boldsymbol{\Gamma}^*$. Next, we investigate whether the optimization problem (28) can be transformed to an optimization problem with respective to $\boldsymbol{\Theta}\boldsymbol{\Gamma}$ and $\boldsymbol{\Theta}\boldsymbol{\Gamma}^*$ in Theorem 7.

Theorem 7. *Given any optimal solution $\underline{\boldsymbol{\Gamma}}, \underline{\boldsymbol{\Gamma}}^*, \underline{\boldsymbol{\Theta}}$ of the optimization problem (28), if the source kernel k_s and target kernel k_t are universal, and the latent subspace dimension $d \geq 2K$, then*

$$\underline{\boldsymbol{\Theta}\boldsymbol{\Gamma}}, \quad \underline{\boldsymbol{\Theta}\boldsymbol{\Gamma}}^*,$$

is the unique solution of the optimization problem

$$\min_{\mathbf{Z},\mathbf{Z}^*\in\mathbb{R}^{(n_s+n_t)\times K}} \mathcal{L}(\mathbf{Z}, \mathbf{Z}^*), \quad where \tag{30}$$

$$\begin{aligned} \mathcal{L}(\mathbf{Z}, \mathbf{Z}^*) = &\mathrm{tr}\big(\mathbf{Z}^\top(\mathbf{K}(\mathbf{A}^2 + \rho\mathbf{M} + \rho\mathbf{L})\mathbf{K} + \sigma\mathbf{K})\mathbf{Z}\big) \\ &+ \lambda\mathrm{tr}\big(\mathbf{Z}^{*\top}(\mathbf{K}(\mathbf{A}^{*2} + \rho\mathbf{M}^* + \rho\mathbf{L}^*)\mathbf{K} + \sigma\mathbf{K})\mathbf{Z}^*\big) \\ &- 2\mathrm{tr}\big(\mathbf{Y}^\top\mathbf{A}^2\mathbf{K}\mathbf{Z}\big) - 2\lambda\mathrm{tr}\big(\mathbf{Y}^\top\mathbf{A}^{*2}\mathbf{K}\mathbf{Z}^*\big). \end{aligned}$$

Theorem 7 implies an important result that though the solutions of problem (28) are not unique, $\underline{\boldsymbol{\Theta}\boldsymbol{\Gamma}}, \underline{\boldsymbol{\Theta}\boldsymbol{\Gamma}}^*$ are fixed and are the unique solution of the problem (30). Then, we present the solution to problem (30) in Theorem 8.

Theorem 8. *If the kernels k_s and k_t are universal, then the optimization problem (30) has a unique solution*

$$\underline{\mathbf{Z}} = \big((\mathbf{A}^2 + \rho\mathbf{M} + \rho\mathbf{L})\mathbf{K} + \sigma\mathbf{I}\big)^{-1}\mathbf{A}^2\mathbf{Y}, \tag{31}$$

$$\underline{\mathbf{Z}}^* = \big((\mathbf{A}^{*2} + \rho\mathbf{M}^* + \rho\mathbf{L}^*)\mathbf{K} + \sigma\mathbf{I}\big)^{-1}\mathbf{A}^{*2}\mathbf{Y}. \tag{32}$$

Algorithm 1. 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: $Y_u = SVM(\mathbf{X}_l, \mathbf{X}_u)$; // Classify $\mathcal{T}_u$ by SVM, $\mathcal{T}_l$;
3: Assign $Y_u^* = Y_u$;
for $i = 1, 2, \ldots, T$ **do**
4: Compute $\underline{\mathbf{Z}}, \underline{\mathbf{Z}}^*$ by Eqs. (31), (32) with pseudo labels Y_u and Y_u^*;
5: Obtain $\boldsymbol{h} \circ \boldsymbol{T}_t, \boldsymbol{h}^* \circ \boldsymbol{T}_t$ by $\underline{\mathbf{Z}}, \underline{\mathbf{Z}}^*$ (Eqs. (33), (34));
6: Update Y_u^* by $\boldsymbol{h}^* \circ \boldsymbol{T}_t(\mathbf{X}_u)$;
7: Compute $\underline{\mathbf{Z}}$ by Eq. (31) with pseudo labels Y_u^*;
8: Obtain $\widetilde{\boldsymbol{h}} \circ \boldsymbol{T}_t$ by $\underline{\mathbf{Z}}$ and Eq. (33);
9: Obtain the classifier $\boldsymbol{f}$ by Eq. (35);
10: Update pseudo labels Y_u by $\boldsymbol{f}(\mathbf{X}_u)$;
11: Output predicted target labels Y_u.

Based on Theorem 8, $\boldsymbol{h} \circ \boldsymbol{T}_t$ and $\boldsymbol{h}^* \circ \boldsymbol{T}_t$ can be written as

$$\boldsymbol{h} \circ \boldsymbol{T}_t(\mathbf{x}) = \sum_{i=1}^{n_t} \underline{\mathbf{Z}}_i k_t(\mathbf{x}, \mathbf{x}_t^i), \tag{33}$$

$$\boldsymbol{h}^* \circ \boldsymbol{T}_t(\mathbf{x}) = \sum_{i=1}^{n_t} \underline{\mathbf{Z}}_i^* k_t(\mathbf{x}, \mathbf{x}_t^i), \tag{34}$$

where $\mathbf{x} \in \mathcal{X}_t$, 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.

6.4 KHDA Algorithm

To compute Eqs. (31) and (32), the labels of the unlabeled target data are required. However, we have not any label information related to these data. A simple and effective method is to use the pseudo label iterative strategy [12], [55], [58], [59]. Motivated by pseudo-label iterative strategy, Algorithm 1 presents how to iteratively improve the quality of Eqs. (31) and (32) and give a final kernel-based solution to the SsHeDA problem, which is explained below.

Step 1 (Initialize pseudo labels, lines 2-3). We use SVM with the labeled target data $\mathcal{T}_l$ as the training data to predict the pseudo target labels Y_u for unlabeled target data $\mathcal{T}_u$. Then, we let $Y_u^* = Y_u$.

Step 2 (Construct classifiers, lines 4-5). Using Eqs. (31), (32) with the pseudo labels Y_u^* and Y_u, we obtain $\boldsymbol{h}^* \circ \boldsymbol{T}_t, \boldsymbol{h} \circ \boldsymbol{T}_t$.

Step 3 (Bridge classifiers, lines 6-9). 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. (31) 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

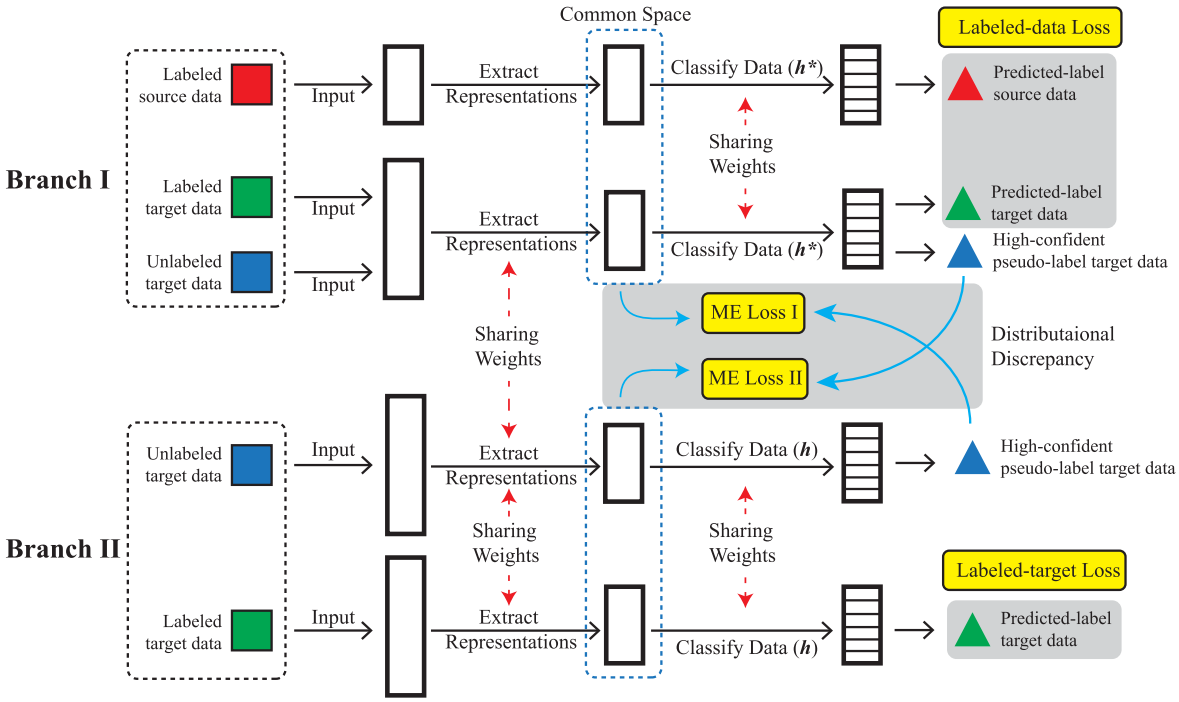


Fig. 2. Network structure of the joint mean embedding alignment (JMEA).

$$f_c(\mathbf{x}) = \max\{h_c \circ \boldsymbol{T}_t(\mathbf{x}), h_c^* \circ \boldsymbol{T}_t(\mathbf{x}), \widetilde{h}_c \circ \boldsymbol{T}_t(\mathbf{x})\}, \tag{35}$$

where h_c , h_c^* and $\widetilde{h}_c$ are the cth coordinate of $\boldsymbol{h}$, $\boldsymbol{h}^*$, $\widetilde{\boldsymbol{h}}$. As a result, the pseudo label of a given target data $\mathbf{x}$ can be predicted by $\arg\max_{c\in\mathcal{Y}}[f_1(\mathbf{x}), \ldots, f_c(\mathbf{x}), \ldots, f_K(\mathbf{x})]$. Using classifier $\boldsymbol{f}$, we obtain the pseudo labels Y_u, where $\boldsymbol{f} = [f_1, \ldots, f_c, .., f_K]$.

Step 4 (Update, line 10). We repeat Steps 2 and 3 until convergence, and choose $\boldsymbol{f}$ as the final classifier.

7 Network-Based Algorithm for SsHeDA

To address the SsHeDA problem in large-scale datasets, this section presents *joint mean embedding alignment* (JMEA) to train a network to classify target data. In JMEA, we use the fully-connected neural networks to construct the spaces $\mathcal{F}_s, \mathcal{F}_t$ and $\mathcal{H}$. Since JMEA does not need to compute the kernel matrix of the whole training set, the computational cost of JMEA is lower than that of the algorithm KHDA.

7.1 Network Structure in JMEA

According to Eqs. (14), (16) and (18), we have the following loss function:

$$\begin{aligned}
&\mathcal{L}(\boldsymbol{h}, \boldsymbol{h}^*, \boldsymbol{T}_s, \boldsymbol{T}_t) \\
&= \widehat{R}_t(\boldsymbol{h} \circ \boldsymbol{T}_t) + \lambda^*(\widehat{R}_s(\boldsymbol{h}^* \circ \boldsymbol{T}_s) + \widehat{R}_t(\boldsymbol{h}^* \circ \boldsymbol{T}_t)) \\
&\quad + \rho\lambda\Big(D^2_{\mathcal{F}}(\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_t)}, \widehat{P}_{\boldsymbol{T}_s(\mathbf{X}_s)}) + \sum_{c=1}^{K} D^2_{\mathcal{F}}(\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_l^c)}, \widehat{P}_{\boldsymbol{T}_s(\mathbf{X}_s^c)})\Big) \\
&\quad + \rho\Big(D^2_{\mathcal{F}}(\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_l)}, \widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_u)}) + \sum_{c=1}^{K} D^2_{\mathcal{F}}(\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_l^c)}, \widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_u^{\bar{c}})})\Big),
\end{aligned}$$

where λ^* is a parameter to re-weight the labeled-data loss $\widehat{R}_s(\boldsymbol{h}^* \circ \boldsymbol{T}_s) + \widehat{R}_t(\boldsymbol{h}^* \circ \boldsymbol{T}_t)$ and always set to 2 in this paper.

Since we need to minimize above loss function by using two different classifiers $\boldsymbol{h}^*$ and $\boldsymbol{h}$ over the representations of source and target data, the network used in JMEA contains two branches (see Fig. 2). The first branch takes the labeled source, labeled target and unlabeled target data as inputs and aims to train a classifier (i.e., $\boldsymbol{h}^*$) to classify source representations (i.e., $\boldsymbol{T}_s(\mathbf{X}_s)$) and target representations (i.e., $\boldsymbol{T}_t(\mathbf{X}_t)$) well. The second branch only takes labeled and unlabeled target data as inputs and aims to train a classifier (i.e., $\boldsymbol{h}$) for the target representation (i.e., $\boldsymbol{T}_t(\mathbf{X}_t)$). Note that, the second branch is used as the final target classifier.

7.2 Loss Function in JMEA

In the first branch, we need to train a classifier to classify source and target representations well. Thus, the following loss function is used to optimize the parameters of Branch I

$$\begin{aligned}
&\mathcal{L}_{\mathbf{I}}(\boldsymbol{h}^*, \boldsymbol{T}_s, \boldsymbol{T}_t) = 2\big(\widehat{R}_s(\boldsymbol{h}^* \circ \boldsymbol{T}_s) + \widehat{R}_t(\boldsymbol{h}^* \circ \boldsymbol{T}_t)\big) \\
&+ \rho\lambda\Big(D^2_{\mathcal{F}}(\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_t)}, \widehat{P}_{\boldsymbol{T}_s(\mathbf{X}_s)}) + \sum_{c=1}^{K} D^2_{\mathcal{F}}(\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_l^c)}, \widehat{P}_{\boldsymbol{T}_s(\mathbf{X}_s^c)})\Big),
\end{aligned}$$

where $D^2_{\mathcal{F}}(\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_t)}, \widehat{P}_{\boldsymbol{T}_s(\mathbf{X}_s)})$ is expressed by

$$\Big\| \frac{1}{n_s} \sum_{\mathbf{x}\in\mathbf{X}_s} \boldsymbol{T}_s(\mathbf{x}) - \frac{1}{n_t} \sum_{\mathbf{x}\in\mathbf{X}_t} \boldsymbol{T}_t(\mathbf{x}) \Big\|_2^2, \tag{36}$$

and $D^2_{\mathcal{F}}(\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_l^c)}, \widehat{P}_{\boldsymbol{T}_s(\mathbf{X}_s^c)})$ is expressed as follows:

$$\Big\| \frac{1}{n_s^c} \sum_{\mathbf{x}\in\mathbf{X}_s^c} \boldsymbol{T}_s(\mathbf{x}) - \frac{1}{n_l^c} \sum_{\mathbf{x}\in\mathbf{X}_l^c} \boldsymbol{T}_t(\mathbf{x}) \Big\|_2^2. \tag{37}$$

To compute target mean embedding in Eq. (37) accurately, pseudo labels are introduced to compute the second term in Eq. (37). Thus, Eq. (37) is revised to

$$\Big\| \frac{1}{n_s^c} \sum_{\mathbf{x}\in\mathbf{X}_s^c} \boldsymbol{T}_s(\mathbf{x}) - \frac{1}{n_l^c + n_u^{\bar{c}}} \sum_{\mathbf{x}\in\mathbf{X}_t^{\bar{c}}} \boldsymbol{T}_t(\mathbf{x}) \Big\|_2^2, \tag{38}$$

where $\mathbf{X}_t^{\bar{c}}$ is $[\mathbf{X}_l^c, \mathbf{X}_u^{\bar{c}}]$.

Algorithm 2. JMEA Algorithm for SsHeDA

1: Input Data $\mathcal{S}, \mathcal{T}_u, \mathcal{T}_l$; #Epochs T; Lowest selection rate r; Mini-batch size n_b; Parameters λ, ρ;
2: Initial $\boldsymbol{T}_s$, $\boldsymbol{T}_t$, $\boldsymbol{h}$, $\boldsymbol{h}^*$;
for $i = 1, 2, \ldots, T$ **do**
  3: Shuffle datasets $\mathcal{S}, \mathcal{T}_u, \mathcal{T}_l$;
  4: Update $k = \min\{\lfloor n_b \times i/T_{\max} + r \rfloor, n_b\}$; // Set the number of high-confident pseudo-label target data
  for $N = 1, \ldots, N_{max}$ **do**
    5: Fetch mini-batches from $\mathcal{S}$ and $\mathcal{T}_u$; // We use the full batch for $\mathcal{T}_l$
    6: Compute labeled data loss: $\mathcal{L}_l^* = 2\big(\widehat{R}_s(\boldsymbol{h}^* \circ \boldsymbol{T}_s) + \widehat{R}_t(\boldsymbol{h}^* \circ \boldsymbol{T}_t)\big)$;
    7: Compute labeled target loss: $\mathcal{L}_l = \widehat{R}_t(\boldsymbol{h} \circ \boldsymbol{T}_t)$;
    8: Obtain soft pseudo labels $\bar{Y}_u^* = \{\bar{y}_u^{*i}\}_{i=1}^{n_u}$ by $\boldsymbol{h}^* \circ \boldsymbol{T}_t$; // $\bar{y}_u^{*i} = \max_{c\in\mathcal{Y}} h_c^* \circ \boldsymbol{T}_t(\mathbf{x}_u^i)$
    9: Obtain soft pseudo labels $\bar{Y}_u = \{\bar{y}_u^{i}\}_{i=1}^{n_u}$ by $\boldsymbol{h} \circ \boldsymbol{T}_t$; // $\bar{y}_u^{i} = \max_{c\in\mathcal{Y}} h_c \circ \boldsymbol{T}_t(\mathbf{x}_u^i)$
    10: Select top k high-confident soft labels from $\bar{Y}_u$ and denote their index by $\mathcal{I}^k = \cup_{c\in\mathcal{Y}}\mathcal{I}_c^k$;
    11: Select top k high-confident soft labels from $\bar{Y}_u^*$ and denote their index by $\mathcal{I}^{*k} = \cup_{c\in\mathcal{Y}}\mathcal{I}_c^{*k}$;
    12: Compute ME loss I: $\mathcal{L}_M = \rho$ (Eq. (36)+Eq. (41)); // Discrepancy between source and target
    13: Compute ME loss II: $\mathcal{L}_M^* = \rho\lambda$ (Eq. (39)+Eq. (42)); // Discrepancy between pseudo-label and labeled target
    14: Compute the overall loss: $\mathcal{L}_l^* + \mathcal{L}_l + \mathcal{L}_M + \mathcal{L}_M^*$;
    15: Update $\boldsymbol{T}_s$, $\boldsymbol{T}_t$, $\boldsymbol{h}$, $\boldsymbol{h}^*$ by minimizing the overall loss;
16: Output Predicted target labels Y_u by $\boldsymbol{h} \circ \boldsymbol{T}_t(\mathbf{X}_u)$.

In the second branch, we need to train a classifier to classify target representations well. Thus, the following loss function is used to optimize the parameters of Branch II

$$\mathcal{L}_{\mathbf{II}}(\boldsymbol{h}, \boldsymbol{T}_s, \boldsymbol{T}_t) = \widehat{R}_t(\boldsymbol{h} \circ \boldsymbol{T}_t) + \rho D_{\mathcal{F}}^2(\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_l)}, \widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_u)}) + \rho \sum_{c=1}^{K} D_{\mathcal{F}}^2(\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_l^c)}, \widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_u^{\bar{c}})}),$$

where $D_{\mathcal{F}}^2(\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_l)}, \widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_u)})$ is expressed by

$$\Big\| \frac{1}{n_l} \sum_{\mathbf{x}\in\mathbf{X}_l} \boldsymbol{T}_t(\mathbf{x}) - \frac{1}{n_u} \sum_{\mathbf{x}\in\mathbf{X}_u} \boldsymbol{T}_t(\mathbf{x}) \Big\|_2^2, \tag{39}$$

and $D_{\mathcal{F}}^2(\widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_l^c)}, \widehat{P}_{\boldsymbol{T}_t(\mathbf{X}_u^{\bar{c}})})$ is expressed as follows:

$$\Big\| \frac{1}{n_l^c} \sum_{\mathbf{x}\in\mathbf{X}_l^c} \boldsymbol{T}_t(\mathbf{x}) - \frac{1}{n_u^{\bar{c}}} \sum_{\mathbf{x}\in\mathbf{X}_u^{\bar{c}}} \boldsymbol{T}_t(\mathbf{x}) \Big\|_2^2. \tag{40}$$

Since we need to use pseudo labels to compute Eqs. (38) and (40), we apply high-confident pseudo labels and the soft-label trick [44] to ensure the high quality of pseudo labels. Hence, we revise Eq. (38) to

$$\Big\| \frac{1}{n_s^c} \sum_{\mathbf{x}\in\mathbf{X}_s^c} \boldsymbol{T}_s(\mathbf{x}) - \frac{\sum_{\mathbf{x}\in\mathbf{X}_l^c} \boldsymbol{T}_t(\mathbf{x}) + \sum_{i\in\mathcal{I}_c^k} \boldsymbol{T}_t(\mathbf{x}_u^i)\bar{y}_u^i}{n_l^c + |\mathcal{I}_c^k|} \Big\|_2^2, \tag{41}$$

where $\mathcal{I}_c^k = \mathcal{I}^k \cap \mathcal{I}_c$, $\mathcal{I}^k$ is a set collecting the index of the top-k high-confident target data (annotated by $\boldsymbol{h} \circ \boldsymbol{T}_t$), $\mathcal{I}_c$ is a set collecting the index of target data with the pseudo label c, and $\bar{y}_u^i$ is the soft label of $\mathbf{x}_u^i$ (i.e., $\bar{y}_u^i = \max_{c\in\mathcal{Y}} h_c \circ \boldsymbol{T}_t(\mathbf{x}_u^i)$). Similarly, we revise Eq. (40) to

$$\Big\| \frac{1}{n_l^c} \sum_{\mathbf{x}\in\mathbf{X}_l^c} \boldsymbol{T}_t(\mathbf{x}) - \frac{\sum_{i\in\mathcal{I}_c^{*k}} \boldsymbol{T}_t(\mathbf{x}_u^i)\bar{y}_u^{*i}}{|\mathcal{I}_c^{*k}| + \eta} \Big\|_2^2, \tag{42}$$

where η is a small constant 10^{-6} to avoid numerical problems when $|\mathcal{I}_c^{*k}|$ gets close to 0, $\mathcal{I}_c^{*k} = \mathcal{I}^{*k} \cap \mathcal{I}_c^*$, $\mathcal{I}^{*k}$ is a set collecting the index of the top-k high-confident target data (annotated by $\boldsymbol{h}^* \circ \boldsymbol{T}_t$), $\mathcal{I}_c^*$ is a set collecting the index of target data with the pseudo label c, and $\bar{y}_u^{*i} = \max_{c\in\mathcal{Y}} h_c^* \circ \boldsymbol{T}_t(\mathbf{x}_u^i)$.

Note that we have applied the co-teaching manner to avoid accumulating errors of pseudo target labels [60], [61]. Namely, the pseudo-labeled target data used in Branch I are annotated by Branch II (i.e., $\bar{y}_u^i$), and the pseudo-labeled target data used in Branch II are annotated by Branch I (i.e., $\bar{y}_u^{*i}$). Since two branches have different views to annotate unlabeled target data, two branches can teach each other to avoid accumulating errors of pseudo target labels [61], [62].

Finally, JMEA has four parts of the overall loss function:

1. Labeled-data loss: $\mathcal{L}_l^* = 2\big(\widehat{R}_s(\boldsymbol{h}^* \circ \boldsymbol{T}_s) + \widehat{R}_t(\boldsymbol{h}^* \circ \boldsymbol{T}_t)\big)$;
2. Labeled-target loss: $\mathcal{L}_l = \widehat{R}_t(\boldsymbol{h} \circ \boldsymbol{T}_t)$;
3. *Mean embedding* (ME) loss I: $\mathcal{L}_M = \rho$(Eq. (36)+ Eq. (41));
4. ME loss II: $\mathcal{L}_M^* = \rho\lambda$ (Eq. (39) + Eq. (42)).

7.3 JMEA Algorithm

Algorithm 2 presents how JMEA trains a network to classify data from the target domain. First, we initialize parameters of $\boldsymbol{T}_s$, $\boldsymbol{T}_t$, $\boldsymbol{h}$, $\boldsymbol{h}^*$ (line 2). Then we shuffle data $\mathcal{S}, \mathcal{T}_u, \mathcal{T}_l$, and update the value of k to decide how many high-confident pseudo-label target data should be selected (lines 3 and 4). After a mini-batch is fetched, we obtain the pseudo labels of the unlabeled target data by $\boldsymbol{h}^* \circ \boldsymbol{T}_s$ and $\boldsymbol{h} \circ \boldsymbol{T}_t$ (lines 5 and 6), respectively. Based on the confidence of each pseudo label, we select top-k high-confident pseudo-label target data (lines 7 and 8). Then we can compute the overall loss (lines 9-11) and update the parameters of $\boldsymbol{T}_s$, $\boldsymbol{T}_t$, $\boldsymbol{h}$, $\boldsymbol{h}^*$ by minimizing the overall loss.

8 Experiments and Evaluations

This section empirically evaluates the proposed algorithms KHDA and JMEA on different SsHeDA tasks. We then conducted experiments to analyze the sensitivity of

TABLE 2
Accuracy (%) With Standard Error on *Image↔Image* Tasks

Tasks	Non-neural Network Algorithm								Neural Network Algorithm		
	1NN	SVMt	DAMA	SHFA	G-JDA	CDLS	DACoM	KHDA	TNT	STN	JMEA
#labeled target data/class = 1											
I8-50→C8-101	49.6±2.2	49.6±2.4	43.5±1.3	59.8±1.6	73.1±2.4	70.3±2.0	64.1±1.1	75.7±2.5	76.8±2.8	76.9±3.7	**83.6**±3.6
I8-101→C8-50	49.1±1.3	49.1±1.4	44.0±1.7	64.2±0.7	71.0±2.2	71.2±1.6	61.2±1.7	78.5±2.3	75.3±2.0	72.9±2.9	**77.0**±2.0
I59-50→C59-101	39.0±0.8	39.0±0.8	38.9±0.6	46.2±0.8	48.9±0.9	51.0±1.0	33.8±0.5	57.0±0.9	38.9±1.1	30.9±0.5	**51.0**±0.8
I59-101→C59-50	37.6±0.9	38.0±0.9	38.1±0.9	41.5±1.2	45.1±1.0	47.3±1.0	34.8±0.9	52.8±1.0	40.1±0.7	34.7±1.1	**48.1**±1.3
Avg	43.8	43.9	41.1	52.9	59.5	61.5	48.5	66.0	57.8	53.9	**64.9**
#labeled target data/class = 3											
I8-50→C8-101	65.9±1.1	73.5±1.5	73.4±1.1	80.4±1.9	86.4±1.9	86.5±1.4	78.2±1.5	89.8±1.2	87.4±1.4	88.9±0.7	**91.1**±0.2
I8-101→C8-50	64.6±1.9	73.5±1.5	68.5±1.6	80.2±1.0	86.1±0.4	85.7±0.7	78.3±1.1	87.8±1.0	85.8±0.6	86.9±0.8	**89.3**±0.3
I59-50→C59-101	52.7±0.4	59.1±0.3	52.4±0.4	62.4±0.3	63.4±0.6	64.1±0.7	56.1±0.6	69.8±0.5	55.3±0.6	63.9±0.8	**70.6**±0.6
I59-101→C59-50	48.6±0.5	55.4±0.4	51.5±0.4	59.6±0.6	59.9±0.5	60.3±0.4	52.9±0.6	65.4±0.4	50.6±0.5	60.6±0.8	**66.8**±0.5
Avg	58.0	65.4	61.5	70.7	74.0	74.2	66.4	78.2	69.8	75.1	**79.5**
#labeled target data/class = 5											
I8-50→C8-101	76.4±0.8	82.8±0.8	81.9±0.7	87.4±0.3	90.2±0.2	89.5±0.2	84.6±0.5	90.9±0.3	90.1±0.2	89.6±0.8	**91.5**±0.2
I8-101→C8-50	69.8±0.9	77.7±0.6	76.7±0.6	82.3±0.3	88.2±0.2	86.7±0.3	80.4±0.3	88.8±0.1	86.6±0.3	88.1±0.2	**89.3**±0.4
I59-50→C59-101	60.5±0.5	66.0±0.3	61.2±0.4	69.1±0.4	68.1±0.3	69.2±0.3	63.6±0.4	72.9±0.2	62.8±0.4	70.7±0.3	**74.8**±0.1
I59-101→C59-50	48.6±0.5	62.2±0.4	61.6±0.3	64.4±0.4	63.7±0.4	64.8±0.6	60.7±0.4	69.3±0.4	57.4±0.4	67.3±0.4	**71.3**±0.4
Avg	63.8	72.2	70.4	75.8	77.6	77.6	72.3	80.5	74.2	78.9	**81.8**

The underline indicates the best accuracy among all non-neural network algorithms, and the bold color indicates the best accuracy among all neural network algorithms.

hyperparameters. More experiments are given in Appendix VIII, available in the online supplemental material.

8.1 Datasets and Baseline Algorithms

• *Image↔Image. CIFAR-10/100* [63] is an image dataset with $10/100$ classes. *The ILSVRC2012 of ImageNet* is an image dataset, which has 1,000 classes [64]. CIFAR-10/100 and ILSVRC2012 have different resolutions. We construct ILSVRC2012-8/59 and CIFAR-8/59 by selecting 8/59 shared classes from CIFAR-10/100 and ILSVRC2012. Then, we use Big Transfer-M (BiTM) with ResNet-50 and ResNet-101 [65] to extract the features. By considering each as a domain, we build 8 datasets: I8-50/101 (ILSVRC2012-8 with BiTM ResNet-50/101), I59-50/101 (ILSVRC2012-59 with BiTM ResNet-50/101), C8-50/101 (CIFAR-8 with BiTM ResNet-50/101) and C59-50/101 (CIFAR-59 with BiTM ResNet-50/101). With these datasets, we construct several SsHeDA tasks: I8-50 → C8-101, I8-101 → C8-50, I59-50 → C59-101, and I59-101 → C59-50 (i.e., to transfer knowledge from ImageNet to CIFAR10/100). For I8/I59 and C8/C59, we randomly select $500/50$ labeled source data per class and $500/50$ unlabeled target data per class for training. We randomly choose 1,3,5 data per class as labeled target data. The average accuracy and standard error of 10 random trials are shown in Table 2.

• *Text↔Image. UPMC Food-101* dataset [66] contains text and image datasets and consist of about 100,000 recipes with 101 food categories. For images (I), we use the Big Transfer-M (BiTM) with ResNet-50 [65] to extract the features. For text features (T), we adopt NLP model BERT [67] to extract the features [68]. Then, we randomly select 30 data per class as the source data, 30 data per class as the unlabeled target data, and 1,3,5 data per class as labeled target data. There are 6 SsHeDA tasks. The average accuracy and standard error of 10 random trials are shown in Table 3.

• *Text↔Image. Wikipedia* dataset [69], [70] is extracted from Wikipedia feature articles and consists of 2,866 image-text pairs with 10 semantic classes. For images (I), we use the Big Transfer-M (BiTM) with ResNet-101 [65] to extract the features. For text features (T), since most of Wikipedia's texts are long-sequence, we adopt the NLP model Big Bird [71] to extract the features. All data in the source domain are selected randomly as the labeled source data. For the target domain, we randomly choose 3,5,7 data per class as labeled target data, and randomly choose 50 data per class in the remaining data as the unlabeled target data. There are 6 SsHeDA tasks. The average accuracy and standard error of 10 random trials are shown in Table 3.

• *Text↔Text. Multilingual Reuters Collection (MRC)* [72], [73] 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 [43], we use BOW with TF-ITF to describe each article. Then we use PCA in the BoW features to preserve 60% energy [72], [73]. We set English, French, Italian and German as the source domains and Spanish as the target domain. 100 data per class in the source domain are selected randomly as the labeled source data. For the target domain, we randomly choose 10,15,20 data per class as the labeled target data and randomly choose 500 data per class as the unlabeled target data. There are 12 SsHeDA tasks. The average accuracy and standard error of 20 random trials are shown in Table 4.

• *Image→Text (end-to-end). Road-View* dataset is constructed through the *natural language-based vehicle retrieval (NLVR)* dataset [74] for end-to-end learning tasks. The road-

TABLE 3
Accuracy (%) With Standard Error on *Text*↔*Image* Tasks

Tasks	Non-neural Network Algorithm								Neural Network Algorithm		
	1NN	SVMt	DAMA	SHFA	G-JDA	CDLS	DACoM	KHDA	TNT	STN	JMEA
#labeled target data/class = 1											
I→T Food	50.3±0.6	50.3±0.7	51.8±0.6	53.9±0.6	54.3±0.7	52.5±0.8	50.8±0.6	55.4±0.7	49.7±0.7	54.2±0.6	**56.3**±0.7
T→I Food	21.4±0.2	21.4±0.4	21.8±0.2	25.2±0.3	20.8±0.3	21.9±0.3	15.4±0.8	28.1±0.3	19.7±0.5	27.2±0.4	**30.8**±0.4
Avg	35.9	35.9	36.7	39.6	37.6	37.2	33.1	41.8	34.7	40.7	**43.6**
#labeled target data/class = 3											
I→T Food	61.3±0.3	61.0±0.3	57.3±0.3	64.9±0.4	65.3±0.3	62.4±0.5	62.1±0.3	66.7±0.4	59.4±0.4	67.0±0.3	**68.2**±0.4
T→I Food	29.6±0.5	37.2±0.4	34.5±0.3	39.8±0.4	35.3±0.4	38.3±0.3	27.4±0.4	43.8±0.4	22.7±0.6	44.4±0.3	**45.9**±0.4
Avg	45.5	49.1	45.9	52.4	50.3	50.4	44.8	55.3	41.1	55.7	**57.1**
#labeled target data/class = 5											
I→T Food	64.4±0.3	63.8±0.3	66.4±0.3	68.6±0.3	67.9±0.2	65.8±0.4	64.8±0.3	70.1±0.3	65.3±0.4	70.4±0.3	**71.1**±0.3
T→I Food	33.4±0.3	42.7±0.3	41.3±0.3	47.6±0.3	42.4±0.3	43.3±0.4	33.4±0.2	50.0±0.3	31.5±0.4	50.6±0.4	**51.3**±0.4
Avg	48.9	53.3	53.9	58.1	55.2	54.6	49.1	60.1	48.4	60.5	**61.2**
#labeled target data/class = 3											
I→T Wiki	80.0±1.1	78.7±0.9	80.5±1.0	86.6±0.5	84.0±0.9	86.8±0.6	80.4±0.6	87.3±0.6	87.5±0.8	87.4±0.7	**88.0**±0.7
T→I Wiki	23.7±0.9	25.9±0.8	23.1±0.9	29.2±1.1	28.4±1.4	26.9±1.9	27.0±0.7	34.3±1.1	31.8±1.5	32.7±1.3	**33.2**±1.1
Avg	51.9	52.3	51.8	57.9	56.2	56.9	53.7	60.8	59.7	60.1	**60.6**
#labeled target data/class = 5											
I→T Wiki	80.8±0.9	79.3±0.5	80.8±0.9	86.9±0.5	85.3±0.9	87.2±0.6	81.4±0.7	87.6±0.4	87.3±0.6	87.7±0.7	**88.1**±0.6
T→I Wiki	26.8±0.8	30.7±0.9	24.4±0.5	31.8±0.9	34.9±0.7	25.8±1.0	32.1±0.7	38.7±1.0	36.5±1.0	36.1±0.8	**36.7**±0.8
Avg	53.8	55.0	52.6	59.4	60.1	56.5	56.8	63.2	61.9	61.9	**62.4**
#labeled target data/class = 7											
I→T Wiki	83.0±0.7	81.1±0.6	81.8±0.9	88.0±0.5	87.7±0.9	87.3±0.6	85.3±0.3	88.2±0.4	88.1±0.3	88.4±0.4	**88.9**±0.4
T→I Wiki	29.7±0.9	34.3±0.6	27.1±0.9	36.7±1.0	38.1±0.6	31.2±0.9	36.9±0.5	42.1±0.6	39.5±0.9	38.8±0.9	**40.0**±1.0
Avg	56.4	57.7	54.5	62.4	62.9	59.3	61.1	65.2	63.8	63.6	**64.5**

The underline indicates the best accuracy among all non-neural network algorithms, and the bold color indicates the best accuracy among all neural network algorithms.

TABLE 4
Accuracy (%) With Standard Error on *Text*↔*Text* Tasks

Tasks	Non-neural Network Algorithm								Neural Network Algorithm		
	1NN	SVMt	DAMA	SHFA	G-JDA	CDLS	DACoM	KHDA	TNT	STN	JMEA
#labeled target data/class = 10											
English	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	73.0±0.5	69.7±0.7	73.3±0.6	**74.4**±0.5
France	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	73.1±0.5	69.0±0.8	73.4±0.6	**74.4**±0.5
German	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	73.2±0.5	69.2±0.8	73.4±0.6	**74.0**±0.6
Italian	63.0±0.5	61.6±0.9	63.2±1.0	71.3±0.5	70.1±1.0	69.8±0.9	72.8±0.5	73.0±0.5	69.2±0.9	73.4±0.6	**74.3**±0.5
Avg	63.0	61.6	63.1	70.9	69.9	69.7	72.6	73.1	69.2	73.4	**74.3**
#labeled target data/class = 15											
English	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	75.9±0.4	70.0±0.5	76.3±0.5	**76.5**±0.4
France	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	75.9±0.4	70.5±0.6	76.3±0.4	**76.8**±0.4
German	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	75.9±0.4	71.4±0.6	**76.5**±0.5	**76.5**±0.4
Italian	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	75.8±0.4	70.5±0.6	76.3±0.5	**76.6**±0.4
Avg	66.9	68.0	65.4	74.1	73.7	73.2	75.4	75.9	70.6	76.3	**76.6**
#labeled target data/class = 20											
English	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	78.1±0.5	71.1±0.5	78.3±0.4	**78.6**±0.5
France	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	77.8±0.5	71.7±0.5	78.1±0.5	**78.4**±0.5
German	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	77.9±0.5	71.6±0.7	78.1±0.5	**78.4**±0.5
Italian	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	78.0±0.5	71.9±0.6	78.2±0.5	**78.6**±0.5
Avg	69.3	71.2	71.7	76.8	76.6	75.0	77.3	77.9	71.6	78.2	**78.5**

The underline indicates the best accuracy among all non-neural network algorithms, and the bold color indicates the best accuracy among all neural network algorithms.

TABLE 5
Accuracy (%) of JMEA and Baselines on the Road-View Task (End-to-End Learning Task)

n_l^{p}	Target-ERM	ST-ERM	STN	JMEA-BII	JMEA
1	29.3±1.0	32.0±0.7	30.6±2.3	28.4±2.5	**34.9**±1.7
3	62.3±2.0	58.9±0.7	66.7±1.3	61.2±3.6	**68.2**±2.0
5	75.1±1.2	79.9±1.3	88.5±1.5	89.4±3.5	**91.0**±1.4
Avg	55.6	56.9	62.0	59.7	**64.7**

n_l^{p} *is the number of labeled target data per class.*

view images (i.e., the source domain) are from 20 different cameras in the NLVR dataset and are reconstructed into 20 different classes, containing 68,254 images. Meanwhile, we construct a text dataset (i.e., the target domain) that describes such road-view images, containing 7,700 sequences. In this road-view task, we aim to train a classifier to identify 20 different roads that text sequences describe using few-labeled text sequences (1,3,5 per class), unlabeled text sequences and road-view images. We use the ResNet-50 [75] and the NLP model RoBERTa-large [76] as the backbones for images and texts, respectively. The average accuracy and standard error of 5 random trials are shown in Table 5.

• *Other Datasets.* Due to space limitations, details about *Office+Caltech256* [77] (24 tasks), *ImageCLEF-DA* [78] (32 tasks), *Reuters-21578* [18] (6 tasks) and *Cross-lingual Sentiment* [31] (3 tasks) are given in Appendix VIII, available in the online supplemental material.

• *Baseline Algorithms.* 1NN, SVMt, DAMA [26], SHFA [23], G-JDA [43], CDLS [41], DACoM [39], TNT [24] and STN [44] are used as the baseline algorithms for non-end-to-end tasks. Except for 1NN and SVMt, the details on other baselines are given in Section 2. In the end-to-end task, we consider the following baselines: 1) Target-ERM where we only use labeled target data to fine-tune the RoBERTa-large model [76], and 2) ST-ERM where we train JMEA only using labeled information in both domains, and 3) JMEA-BII where we use labeled and unlabeled target data to fine-tune the RoBERTa-large model [76] (i.e., only training Branch II of JMEA), and 4) the end-to-end version of STN [44].

8.2 Experimental Setup

Before detailing the evaluation results, we explain how the parameters of KHDA and JMEA are set.

• *Parameters for KHDA.* There are several parameters: 1) the kernel k_s and k_t; 2) #iterations T; 3) σ, ρ, #neighbor p.

As suggested in [42], [50], we choose the Gaussian kernel

$$k_\nu(\mathbf{x}_\nu, \mathbf{x}'_\nu) = \exp\Big(-\frac{\|\mathbf{x}_\nu - \mathbf{x}'_\nu\|_2^2}{2r_\nu^2}\Big),$$

where $\nu \in \{s, t\}$, $\mathbf{x}_\nu, \mathbf{x}'_\nu \in \mathbf{X}_\nu$ and the kernel bandwidth r_ν is median$(\|\mathbf{x}_\nu - \mathbf{x}'_\nu\|_2)$, $\forall\, \mathbf{x}_\nu, \mathbf{x}'_\nu \in \mathbf{X}_\nu$. When $\nu = s$, $\mathbf{X}_\nu = \mathbf{X}_s$. When $\nu = t$, $\mathbf{X}_\nu = \mathbf{X}_t$. The details of the parameters are shown in Table 1.

• *Parameters for JMEA.* There are several parameters in JMEA: 1) #epochs T; 2) ρ, λ and r. Except for the end-to-end task, the details of these parameters are shown in Table 1. $\boldsymbol{T}_s$ and $\boldsymbol{T}_t$ are three-layer fully-connected neural networks. The $\boldsymbol{h}$ and $\boldsymbol{h}^*$ are two-layer fully-connected neural networks. In the end-to-end task, the backbone of the Branch I of JMEA is the ResNet-50 model [75], and the backbone of Branch II of JMEA is the RoBERTa-large model [76], and we use the end-to-end manner to implement JMEA. Due to the complexity of Road-View task, we detail JMEA's parameter setting regarding the end-to-end task in Appendix VIII, available in the online supplemental material.

• *Metric.* The classification accuracy [55] on the test data is

$$Accuracy = \frac{\sum_{c=1}^{K} |\mathbf{x} \in \mathbf{X}_u^c : \boldsymbol{g}(\mathbf{x}) = c|}{|\mathbf{x} : \mathbf{x} \in \mathbf{X}_u|},$$

where $\boldsymbol{g}$ is the predicted classifier and $\mathbf{X}_u^c$ is the unlabeled target data matrix with true label c.

8.3 Experimental Results

The classification accuracy and standard error on different tasks are shown in Tables 2, 3 and 4.

• *Image↔Image.* 1) In Table 2, compared to all non-neural network baselines, KHDA works the best for almost all tasks $(12/12)$ and the mean accuracy achieves an improvement at least 2.5%. Compared to all neural network baselines, JMEA works the best for all tasks $(11/12)$ and the average accuracy of JMEA achieves an improvement at least 2.5%. KHDA and JMEA both achieve better performance than all baseline algorithms. 2) It is notable that the accuracy of all algorithms increases when using more labeled target data per class. In addition, KHDA is better than JMEA when the number of labeled target data per class is 1. KHDA becomes worse than JMEA if the number of labeled target data per class is 3 or 5. 3) Except for DAMA, baselines SHFA, G-JDA, CDLS, DACoM, TNT and STN achieve better mean performance than 1NN and SVMt. This indicates that the baselines SHFA, G-JDA, CDLS, DACoM, TNT and STN can transfer knowledge from the source data to the target data.

• *Text↔Image.* The results for the Wikipedia and Food-101 datasets are reported in Table 3. 1) JMEA works the best for all tasks $(12/12)$ and has achieved a mean improvement at least 0.5%, compared to all baselines. 2) Among all non-neural network algorithms, KHDA works the best for all tasks $(12/12)$ and has achieved a mean improvement at least 2.0%. 3) In some tasks, STN is slightly better than KHDA $(0.1\% \sim 0.6\%)$. However, in tasks T→I Wiki, KHDA is better than STN $(0.9\% \sim 3.3\%)$. 4) DACoM and TNT are worse than 1NN and SVMt in tasks I→T and T→I Food. The reason is that the number (101) of classes for Food-101 may be beyond the capacity of DACoM and TNT.

• *Text↔Text.* Table 4 shows the means and standard errors of classification accuracy for all algorithms on the MRC. 1) Of all the non-neural network-based algorithms, KHDA performs the best on 11 tasks $(11/12)$. 2) JMEA has achieved the best performance compared to all algorithms and generally outperforms all other baseline algorithms by at least $0.9\%, 0.3\%$ and 0.3% (average accuracy), respectively, for different labeled target data per class. 3) According to the results from Table 4, the accuracy of all algorithms increases when using more labeled target data per class.

• *Image→Text (end-to-end).* Table 5 shows the means and standard errors of classification accuracy of JMEA and

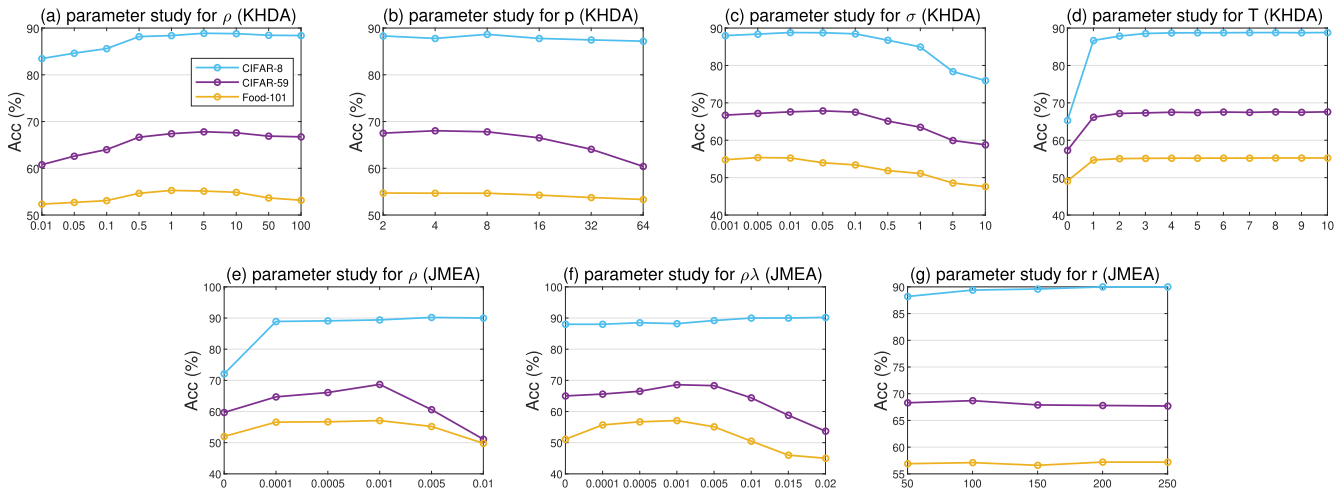


Fig. 3. Parameter study of KHDA and JMEA.

baselines on the Road-View task. In Table 5, JMEA outperforms all baselines. In particular, JMEA has higher accuracy than the state-of-the-art network-based SsHeDA algorithm STN.

8.4 Parameter Sensitivity

We conduct experiments on three different tasks: CIFAR-8, CIFAR-59 and Food-101 (3 labeled target samples per class) to evaluate the mean-accuracy variations of KHDA and JMEA using different parameters.

• *Parameter ρ in KHDA.* We run KHDA with varying values of ρ. Fig. 3a plots the classification accuracy w.r.t. different values of ρ. From this figure, we observe that 1) when ρ is from [1.0,5.0], the performance may be the best and when $\rho = 0.01$, the performance is the worst; 2) as increasing ρ from 0.01 to 1.0, the accuracy increases; 3) as increasing ρ from 1.0 to 100, the accuracy decreases slowly. KHDA can achieve satisfactory performance, if $\rho \in [1.0, 10.0]$.

• *Parameter p in KHDA.* We run KHDA with varying values of p. Fig. 3b plots the classification accuracy w.r.t. different values of p. From this figure, we observe that as increasing p from 2.0 to 64.0, the accuracy on CIFAR-8 is quite stable, and the accuracy on CIFAR-59 and Food-101 decrease slowly. In particular, by changing p in the range of [2.0,10.0], KHDA achieves satisfactory performance.

• *Parameter σ in KHDA.* We run KHDA with varying values of σ. Fig. 3c plots the classification accuracy w.r.t. different values of σ. From this figure, we observe that 1) the performance is the best when $\sigma = 0.001$, and the performance is the worst when $\sigma = 10.0$; 2) as increasing σ from 0.001 to 10.0, the accuracy decreases gradually. Specifically, by changing σ in the range of $[0.001, 0.05]$, the mean accuracy of KHDA is still higher than that of the non-neural network baselines.

• *Parameter T in KHDA.* The results of the convergence analysis are provided in Fig. 3d, which shows that KHDA achieves steady performance in a few iterations ($T < 5$).

• *Parameter ρ in JMEA.* We run JMEA with varying values of ρ. Fig. 3e plots the classification accuracy w.r.t. different values of ρ. From this figure, we observe that 1) the performance of JMEA is very steady, when ρ is in the range [0.0001,0.001] on CIFAR-8 and Food-101 datasets; 2) as increasing ρ from 0.0001 to 0.001, the accuracy on CIFAR-59 dataset increases and achieves the highest value when $\rho = 0.001$. Thus, we recommend selecting ρ in $[0.005, 0.001]$.

• *Parameter $\rho\lambda$ in JMEA.* We run JMEA with varying values of $\rho\lambda$. Fig. 3f plots the classification accuracy w.r.t. different values of $\rho\lambda$. From this figure, we observe that 1) the mean accuracy of JMEA will drop significantly when $\rho\lambda$ is greater than 0.001, meaning that a small value of $\rho\lambda$ will be better if the number of classes is large; 2) when increasing $\rho\lambda$ from 0.0001 to 0.02, the accuracy on CIFAR-8 dataset is quite stable. Overall, if we select $\rho\lambda$ in the range of $[0.0005, 0.001]$, JMEA can achieve satisfactory performance.

• *Parameter r in JMEA.* We run JMEA with varying values of r. Fig. 3g plots the classification accuracy w.r.t. different values of r. From this figure, we observe that the accuracy on all datasets is quite stable, when increasing r from 50 to 250. We recommend selecting r in the range of [100,200].

8.5 Ablation Study

Here we present the ablation study for KHDA and JMEA. This study is conducted on different tasks: CIFAR-8, CIFAR-59 and Food-101 (3 labeled target samples per class). We report average accuracy for different dataset.

• *Ablation Study for KHDA.* We conduct comprehensive experiments to show the contribution of the individual components in KHDA in Table 6. We consider the following baselines: 1) w/o D: In KHDA, train classifiers without distribution alignment Eq. (21). 2) w/o M: In KHDA, train classifiers without manifold regularization Eq. (27). 3) $\rho = 0$: In KHDA, train classifiers without the combination of manifold regularization and distribution alignment. 4) $\sigma = 0$: In KHDA, train classifiers without kernel regularization. 5) $\boldsymbol{h}$, $\boldsymbol{h}^*$, $\widetilde{\boldsymbol{h}}$: In KHDA, the performance of $\boldsymbol{h}$, $\boldsymbol{h}^*$ and $\widetilde{\boldsymbol{h}}$. 6) w/o $\boldsymbol{h}$: In KHDA, train classifiers without $\boldsymbol{h}$. 7) w/o $\boldsymbol{h}^*$: In KHDA, train classifiers without $\boldsymbol{h}^*$. 8) w/o $\widetilde{\boldsymbol{h}}$: In KHDA, train classifiers without $\widetilde{\boldsymbol{h}}$. From these results, we find that:

1) When we omit the distribution alignment (i.e., w/o D), the mean accuracy drops from 70.6% to 66.4%. This indicates that the distribution alignment is important for KHDA. If we omit the manifold regularization (i.e., w/o M), the mean accuracy 70.1% is

TABLE 6
Accuracy (%) of Ablation Study of KHDA on CIFAR-8/59, Food-101 Datasets With 3 Labeled Target Data per Class

Dataset	w/o D	w/o M	$\rho = 0$	$\sigma = 0$	$\boldsymbol{h}^*$	$\widetilde{\boldsymbol{h}}$	$\boldsymbol{h}$	w/o $\boldsymbol{h}^*$	w/o $\widetilde{\boldsymbol{h}}$	w/o $\boldsymbol{h}$	KHDA ($\boldsymbol{f}$)
CIFAR-8	83.7	88.3	81.4	87.0	88.5	88.7	**88.8**	88.3	88.3	87.4	**88.8**
CIFAR-59	62.9	67.0	60.2	50.9	66.1	67.2	67.5	65.0	65.0	67.2	**67.6**
Food-101	52.7	55.1	52.2	1.3	51.4	54.6	55.2	54.0	54.4	49.1	**55.3**
Avg	66.4	70.1	64.5	46.4	68.7	70.2	70.5	69.1	69.2	67.9	**70.6**

TABLE 7
Accuracy (%) of Ablation Study of JMEA on CIFAR-8/59, Food-101 Datasets With 3 Labeled Target Data per Class

Dataset	w/o $\mathcal{L}_l^*$	w/o $\mathcal{L}_l$	w/o $\mathcal{L}_M$	w/o $\mathcal{L}_M^*$	w/o C&S	w/o S	w/o C	JMEA
CIFAR-8	89.4	43.7	88.9	88.0	89.9	89.9	89.6	**90.2**
CIFAR-59	57.8	28.4	64.4	65.0	67.8	67.9	68.2	**68.7**
Food-101	50.1	31.1	53.0	51.1	52.3	52.3	52.4	**57.1**
Avg	65.8	34.4	68.8	68.0	70.0	70.0	70.1	**72.0**

worse than the mean accuracy 70.6% of KHDA. This indicates that the manifold regularization helps KHDA to achieve better performance.

2) If we set $\rho = 0$, then the mean accuracy 64.5% is much lower than the mean accuracy 70.6% of KHDA. This indicates that the combination of manifold regularization and distribution alignment can greatly improve the performance of KHDA. If $\sigma = 0$, the mean accuracy 46.4% is much lower than the mean accuracy 70.6% of KHDA with $\sigma = 0.01$. The reason may be that the overfitting occurs when $\sigma = 0$.
3) We observe that the performance of $\boldsymbol{h}^*$ (68.7%) and $\widetilde{\boldsymbol{h}}$ (70.2%) is worse than the performance of $\boldsymbol{h}$ (70.5%) and $\boldsymbol{f}$ (70.6%). The performance of $\boldsymbol{h}$ and $\boldsymbol{f}$ is similar, which means 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 (70.6%) decreases to 67.9%, 69.1% and 69.2%, respectively. This implies that every classifier is important for KHDA.

• *Ablation Study for JMEA.* We conduct comprehensive experiments and show the contribution of the individual components in JMEA. We consider the following baselines: 1) w/o $\mathcal{L}_l^*$: In JMEA, train two branches without minimizing loss function $\mathcal{L}_l^*$. 2) w/o $\mathcal{L}_l$: In JMEA, train two branches without minimizing loss function $\mathcal{L}_l$. 3) w/o $\mathcal{L}_M$: In JMEA, train two branches without minimizing loss function $\mathcal{L}_M$. 4) w/o $\mathcal{L}_M^*$: In JMEA, train two branches without minimizing loss function $\mathcal{L}_M^*$. 5) w/o C : In JMEA, turn off the checking process. Namely, we replace $\mathcal{I}_c^k$ in Eq. (41) with $\mathcal{I}_c^{*k}$ and to replace $\mathcal{I}_c^{*k}$ in Eq. (42) with $\mathcal{I}_c^k$. 6) w/o S : In JMEA, turn off the selection process. Namely, we set r in Algorithm 2 to n_b. 7) w/o C&S : In JMEA, turning off the checking process and the selection process.

Note that in the full JMEA, we minimize all loss functions and turn on the checking process (C) and the selection process (S) simultaneously. Table 7 reports the average performance of above baselines and JMEA. It is clear that all components in JMEA can help improve the performance. From these results, we can find the following insights:

1) When we remove $\mathcal{L}_M^*$ from the overall loss function used by JMEA, the performance will drop more than the accuracy when we remove $\mathcal{L}_M$. Hence, minimizing $\mathcal{L}_M^*$ is important to improve the performance.
2) It is important to turn on the checking process and the selection process simultaneously since the performance of w/o C&S (70.0%) is much lower than that of JMEA (72.0%). If we only turn on checking (i.e., w/o S) or selection (i.e., w/o C), the accuracy will drop significantly on the Food-101 dataset (from 57.1% to 52.3% or to 52.4%).

9 Conclusion and Future Works

This paper 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 data combined with unlabeled target data help reduce the need for labeled data in the target domain. Then we propose two novel SsHeDA algorithms to bring the proposed theory into reality: KHDA and JMEA. KHDA is kernel-based and is well-suited to situations with small amounts of data. JMEA can deal with massive datasets. It is neural network-based and highly flexible. In sweeping experiments with representative baselines and 104 SsHeDA tasks, the accuracy of JMEA outstrips that of all baselines, and KHDA outperforms all non-neural network baselines. In the future, we will consider HeDA in semantic segmentation tasks [79], [80]. Inspired by Dong *et al.* [81], [82] that develop a novel perspective to distinguish transferable or untransferable representations across domains, we will develop a novel learning theory in semantic segmentation tasks to quantify transferability across heterogeneous domains.

REFERENCES

[1] S. Shalev-Shwartz and S. Ben-David, *Understanding Machine Learning: From Theory to Algorithms*. Cambridge, U.K.: Cambridge Univ. Press, 2014.

[2] J. Q. Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence, *Dataset Shift in Machine Learning*. Cambridge, MA, USA: MIT Press, 2009.

[3] A. Torralba and A. A. Efros, "Unbiased look at dataset bias," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2011, pp. 1521–1528.

[4] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes challenge 2007 results," 2007. [Online]. Available: http://host.robots.ox.ac.uk/pascal/VOC/voc2007/

[5] J. Li, Y. Liu, R. Yin, H. Zhang, L. Ding, and W. Wang, "Multi-class learning: From theory to algorithm," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 1593–1602.

[6] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.

[7] H. Chi *et al.*, "TOHAN: A one-step approach towards few-shot hypothesis adaptation," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2021.

[8] Z. Fang, J. Lu, F. Liu, and G. Zhang, "Unsupervised domain adaptation with sphere retracting transformation," in *Proc. Int. Joint Conf. Neural Netw.*, 2019, pp. 1–8.

[9] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," *IEEE Trans. Neural Netw.*, vol. 22, no. 2, pp. 199–210, Feb. 2011.

[10] S. Ben-David , J. Blitzer, K. Crammer, and F. Pereira, "Analysis of representations for domain adaptation," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2006, pp. 137–144.

[11] C. Cortes, Y. Mansour, and M. Mohri, "Learning bounds for importance weighting," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2010, pp. 442–450.

[12] L. Zhong, Z. Fang, F. Liu, J. Lu, B. Yuan, and G. Zhang, "How does the combined risk affect the performance of unsupervised domain adaptation approaches?," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 11 079–11 087.

[13] J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. Wortman, "Learning bounds for domain adaptation," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2007, pp. 129–136.

[14] I. Redko, E. Morvant, A. Habrard, M. Sebban, and Y. Bennani, *Adv. Domain Adaptation Theory*," Elsevier, 2019.

[15] Y. Zhang, T. Liu, M. Long, and M. I. Jordan, "Bridging theory and algorithm for domain adaptation," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 7404–7413.

[16] Y. Zhang, F. Liu, Z. Fang, B. Yuan, G. Zhang, and J. Lu, "Clarinet: A one-step approach towards budget-friendly unsupervised domain adaptation," in *Proc. Int. Joint Conf. Artif. Intell.*, 2020, pp. 2526–2532.

[17] C. Shen and Y. Guo, "Unsupervised heterogeneous domain adaptation with sparse feature transformation," in *Proc. Asian Conf. Mach. Learn.*, 2018, pp. 375–390.

[18] F. Liu, G. Zhang, and J. Lu, "Heterogeneous domain adaptation: An unsupervised approach," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 12, pp. 5588–5602, Dec. 2020.

[19] H. Li, S. J. Pan, S. Wang, and A. C. Kot, "Heterogeneous domain adaptation via nonlinear matrix factorization," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 3, pp. 984–996, Mar. 2020.

[20] Y. Yan *et al.*, "Learning discriminative correlation subspace for heterogeneous domain adaptation," in *Proc. Int. Joint Conf. Artif. Intell.*, 2017, pp. 3252–3258.

[21] Y. Yan, Q. Wu, M. Tan, M. K. Ng, H. Min, and I. W. Tsang, "Online heterogeneous transfer by hedge ensemble of offline and online decisions," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 7, pp. 3252–3263, Jul. 2018.

[22] J. T. Zhou, S. J. Pan, I. W. Tsang, and Y. Yan, "Hybrid heterogeneous transfer learning through deep learning," in *Proc. AAAI Conf. Artif. Intell.*, 2014, pp. 2213–2220.

[23] W. Li, L. Duan, D. Xu, and I. W. Tsang, "Learning with augmented features for supervised and semi-supervised heterogeneous domain adaptation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 6, pp. 1134–1148, Jun. 2014.

[24] W. Chen, T. H. Hsu, Y. H. Tsai, Y. F. Wang, and M. Chen, "Transfer neural trees for heterogeneous domain adaptation," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 399–414.

[25] S. Sukhija, N. C. Krishnan, and G. Singh, "Supervised heterogeneous domain adaptation via random forests," in *Proc. Int. Joint Conf. Artif. Intell.*, 2016, pp. 2039–2045.

[26] C. Wang and S. Mahadevan, "Heterogeneous domain adaptation using manifold alignment," in *Proc. Int. Joint Conf. Artif. Intell.*, 2011, pp. 1541–1546.

[27] H. Li, S. J. Pan, R. Wan, and A. C. Kot, "Heterogeneous transfer learning via deep matrix completion with adversarial kernel embedding," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 8602–8609.

[28] X. Shi, Q. Liu, W. Fan, P. S. Yu, and R. Zhu, "Transfer learning on heterogenous feature spaces via spectral transformation," in *Proc. IEEE Int. Conf. Data Mining*, 2010, pp. 1049–1054.

[29] S. Ben-David , J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan, "A theory of learning from different domains," *Mach. Learn.*, vol. 79, no. 1/2, pp. 151–175, 2010.

[30] M. Balcan and A. Blum, "A PAC-style model for learning from labeled and unlabeled data," in *Proc. Annu. Conf. Learn. Theory*, 2005, pp. 111–126.

[31] J. T. Zhou, I. W. Tsang, S. J. Pan, and M. Tan, "Multi-class heterogeneous domain adaptation," *J. Mach. Learn. Res.*, vol. 20, pp. 57:1–57:31, 2019.

[32] Y. Mansour, M. Mohri, and A. Rostamizadeh, "Domain adaptation: Learning bounds and algorithms," in *Proc. Conf. Learn. Theory*, 2009.

[33] J. Shen, Y. Qu, W. Zhang, and Y. Yu, "Wasserstein distance guided representation learning for domain adaptation," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 4058–4065.

[34] M. Ghifary, D. Balduzzi, W. B. Kleijn, and M. Zhang, "Scatter component analysis: A unified framework for domain adaptation and domain generalization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 7, pp. 1414–1430, Jul. 2017.

[35] Y. Zhang, B. Deng, H. Tang, L. Zhang, and K. Jia, "Unsupervised multi-class domain adaptation: Theory, algorithms, and practice," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, Nov. 10, 2020, doi: 10.1109/TPAMI.2020.3036956.

[36] P. Germain, A. Habrard, F. Laviolette, and E. Morvant, "A new PAC-Bayesian perspective on domain adaptation," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 859–868.

[37] P. Germain, A. Habrard, F. Laviolette, and E. Morvant, "A PAC-Bayesian approach for domain adaptation with specialization to linear classifiers," in *Proc. Int. Conf. Mach. Learn.*, 2013, pp. 738–746.

[38] O. Day and T. M. Khoshgoftaar, "A survey on heterogeneous transfer learning," *J. Big Data*, vol. 4, 2017, Art. no. 29.

[39] L. Li and Z. Zhang, "Semi-supervised domain adaptation by covariance matching," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 11, pp. 2724–2739, Nov. 2019.

[40] M. Belkin, P. Niyogi, and V. Sindhwani, "Manifold regularization: A geometric framework for learning from labeled and unlabeled examples," *J. Mach. Learn. Res.*, vol. 7, pp. 2399–2434, 2006.

[41] Y.-H. H. Tsai, Y.-R. Yeh, and Y.-C. F. Wang, "Learning cross-domain landmarks for heterogeneous domain adaptation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 5081–5090.

[42] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. J. Smola, "A kernel two-sample test," *J. Mach. Learn. Res.*, vol. 13, pp. 723–773, 2012.

[43] Y.-T. Hsieh, S.-Y. Tao, Y.-H. H. Tsai, Y.-R. Yeh, and Y.-C. F. Wang, "Recognizing heterogeneous cross-domain data via generalized joint distribution adaptation," in *Proc. IEEE Int. Conf. Multimedia Expo*, 2016, pp. 1–6.

[44] Y. Yao, Y. Zhang, X. Li, and Y. Ye, "Heterogeneous domain adaptation via soft transfer network," in *Proc. 27th ACM Int. Conf. Multimedia*, 2019, pp. 1578–1586.

[45] Y. Yan, W. Li, H. Wu, H. Min, M. Tan, and Q. Wu, "Semi-supervised optimal transport for heterogeneous domain adaptation," in *Proc. Int. Joint Conf. Artif. Intell.*, 2018, pp. 2969–2975.

[46] F. Liu, W. Xu, J. Lu, G. Zhang, A. Gretton, and D. J. Sutherland, "Learning deep kernels for non-parametric two-sample tests," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 6316–6326.

[47] F. Liu, W. Xu, J. Lu, and D. J. Sutherland, "Meta two-sample testing: Learning kernels for testing with limited data," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2021.

[48] B. Quanz and J. Huan, "Large margin transductive transfer learning," in *Proc. ACM Int. Conf. Inf. Knowl. Manage.*, 2009, pp. 1327–1336.

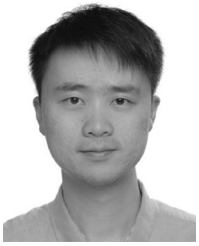
[49] B. K. Natarajan, *Machine Learning: A Theoretical Approach*. San Mateo, CA, USA: Morgan Kaufmann, 1991.

[50] J. Wang, W. Feng, Y. Chen, H. Yu, M. Huang, and P. S. Yu, "Visual domain adaptation with manifold embedded distribution alignment," in *Proc. 26th ACM Int. Conf. Multimedia*, 2018, pp. 402–410.

[51] C. Chuang, A. Torralba, and S. Jegelka, "Estimating generalization under distribution shifts via domain-invariant representations," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 1984–1994.
[52] Z. Fang, J. Lu, F. Liu, J. Xuan, and G. Zhang, "Open set domain adaptation: Theoretical bound and algorithm," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 10, pp. 4309–4322, Oct. 2021.
[53] A. Gretton, A. Smola, J. Huang, M. Schmittfull, K. Borgwardt, and B. Schölkopf, "Covariate shift by kernel mean matching," *Dataset Shift Mach. Learn.*, vol. 3, no. 4, 2009, Art. no. 5.
[54] Y. Yu and C. Szepesvári, "Analysis of kernel mean matching under covariate shift," in *Proc. Int. Conf. Mach. Learn.*, 2012, pp. 607–614.
[55] M. Long, J. Wang, G. Ding, J. Sun, and P. S. Yu, "Transfer feature learning with joint distribution adaptation," in *Proc. Int. Conf. Comput. Vis.*, 2013, pp. 2200–2207.
[56] M. Long, J. Wang, G. Ding, S. J. Pan, and P. S. Yu, "Adaptation regularization: A general framework for transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 5, pp. 1076–1089, May 2014.
[57] B. Schölkopf, R. Herbrich, and A. Smola, "A generalized representer theorem," in *Proc. Int. Conf. Comput. Learn. Theory*, 2001, pp. 416–426.
[58] X. Xu, W. Li, D. Xu, and I. W. Tsang, "Co-labeling for multi-view weakly labeled learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 6, pp. 1113–1125, Jun. 2016.
[59] A. Kumar, T. Ma, and P. Liang, "Understanding self-training for gradual domain adaptation," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 5468–5479.
[60] B. Han *et al.*, "Co-teaching: Robust training of deep neural networks with extremely noisy labels," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 8536–8546.
[61] F. Liu, J. Lu, B. Han, G. Niu, G. Zhang, and M. Sugiyama, "Butterfly: A panacea for all difficulties in wildly unsupervised domain adaptation," in *Proc. Int. Conf. Neural Inf. Process. Syst. Learn. Transferable Skills Workshop*, 2019.
[62] X. Yu, B. Han, J. Yao, G. Niu, I. W. Tsang, and M. Sugiyama, "How does disagreement help generalization against label corruption?," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 7164–7173.
[63] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Master's thesis, Dept. Comput. Sci., Univ. Toronto, Toronto, Canada, 2009.
[64] O. Russakovsky *et al.*, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, 2015.
[65] A. Kolesnikov *et al.*, "Big transfer (BiT): General visual representation learning," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 491–507.
[66] X. Wang, D. Kumar, N. Thome, M. Cord, and F. Precioso, "Recipe recognition with large multimodal food dataset," in *Proc. IEEE Int. Conf. Multimedia Expo Workshops*, 2015, pp. 1–6.
[67] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, 2019, pp. 4171–4186.
[68] I. Gallo, G. Ria, N. Landro, and R. L. Grassa, "Image and text fusion for UPMC food-101 using BERT and CNNs," in *Proc. 35th Int. Conf. Image Vis. Comput. New Zealand*, 2020, pp. 1–6.
[69] N. Rasiwasia *et al.*, "A new approach to cross-modal multimedia retrieval," in *Proc. 18th ACM Int. Conf. Multimedia*, 2010, pp. 251–260.
[70] Y.-R. Yeh, C.-H. Huang, and Y.-C. F. Wang, "Heterogeneous domain adaptation and classification by exploiting the correlation subspace," *IEEE Trans. Image Process.*, vol. 23, no. 5, pp. 2009–2018, May 2014.
[71] M. Zaheer *et al.*, "Big bird: Transformers for longer sequences," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, vol. 33, 2020.
[72] M. Amini, N. Usunier, and C. Goutte, "Learning from multiple partially observed views - an application to multilingual text categorization," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2009, pp. 28–36.
[73] N. Ueffing, M. Simard, S. Larkin, and H. Johnson, "NRC's PORTAGE system for WMT 2007, " in *Proc. 2nd Workshop Statist. Mach. Transl.*, 2007, pp. 185–188.
[74] S. Bai *et al.*, "Connecting language and vision for natural language-based vehicle retrieval," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops*, 2021, pp. 4029–4038.
[75] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.
[76] Y. Liu *et al.*, "RoBERTa: A robustly optimized bert pretraining approach," 2019, *arXiv:1907.11692*.
[77] K. Saenko, B. Kulis, M. Fritz, and T. Darrell, "Adapting visual category models to new domains," in *Proc. Eur. Conf. Comput. Vis.*, 2010, pp. 213–226.
[78] M. Long, H. Zhu, J. Wang, and M. I. Jordan, "Deep transfer learning with joint adaptation networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 2208–2217.
[79] J. Dong, Y. Cong, G. Sun, Y. Liu, and X. Xu, "CSCL: Critical semantic-consistent learning for unsupervised domain adaptation," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 745–762.
[80] J. Dong, Y. Cong, G. Sun, Y. Yang, X. Xu, and Z. Ding, "Weakly-supervised cross-domain adaptation for endoscopic lesions segmentation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 5, pp. 2020–2033, May 2021.
[81] J. Dong, Y. Cong, G. Sun, B. Zhong, and X. Xu, "What can be transferred: Unsupervised domain adaptation for endoscopic lesions segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 4022–4031.
[82] J. Dong, Y. Cong, G. Sun, Z. Fang, and Z. Ding, "Where and how to transfer: Knowledge aggregation-induced transferability perception for unsupervised domain adaptation," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, Nov. 16, 2021, doi: 10.1109/TPAMI.2021.3128560.

Zhen Fang (Member, IEEE) received the MSc degree in pure mathematics from the School of Mathematical Sciences, Xiamen University, Xiamen, China, in 2017. He is currently working toward the PhD degree in the Faculty of Engineering and Information Technology, University of Technology Sydney, Ultimo, Australia. He is a member of the Decision Systems and e-Service Intelligence (DeSI) Research Laboratory, Australian Artificial Intelligence Institute, University of Technology Sydney. His research interests include transfer learning and domain adaptation. He has published several paper related to domain adaptation and transfer learning in IJCNN, AAAI, IJCAI, ICML, NeurIPS, *IEEE Transactions on Neural Networks and Learning Systems*.

Jie Lu (Fellow, IEEE) received the PhD degree from the Curtin University of Technology, Perth, Australia, in 2000. She is currently a distinguished professor and the director of Australian Artificial Intelligence Institute, University of Technology Sydney, Australia. Her research interests include the areas of fuzzy transfer learning, concept drift, decision support systems, and recommender systems. She is an IFSA fellow and Australian Laureate fellow. She has published six research books and more than 450 papers in referred journals and conference proceedings; has won more than 20 ARC Laureate, ARC Discovery Projects, government and industry projects. She serves as editor-in-chief of *Knowledge-Based Systems* (Elsevier) and editor-in-chief of *International Journal of Computational Intelligence Systems*. She has delivered more than 25 keynote speeches at international conferences and chaired 15 international conferences. She has received various awards such as the UTS Medal for Research and Teaching Integration (2010), the UTS Medal for Research Excellence (2019), the Computer Journal Wilkes Award (2018), the IEEE Transactions on Fuzzy Systems Outstanding Paper Award (2019), and the Australian Most Innovative Engineer Award (2019).



Feng Liu (Member, IEEE) received the BSc degree in pure mathematics and the MSc degree in probability and statistics from the School of Mathematics and Statistics, Lanzhou University, Lanzhou, China, in 2013 and 2015, respectively, and the PhD degree in computer science from Artificial Intelligence Institute (AAII), University of Technology Sydney (UTS), Ultimo, Australia, in 2020. He is currently a lecturer with the Australian Artificial Intelligence Institute, University of Technology Sydney, Australia. His research interests include transfer learning, hypothesis testing, and trustworthy machine learning. He has served as program committee members for NeurIPS, ICML, AISTATS, ICLR, AAAI, IJCAI, IJCNN, and FUZZ-IEEE. He has received the Outstanding Reviewer Award of NeurIPS (2021), the Outstanding Reviewer Award of ICLR (2021), and the Best Student Paper Award of FUZZ-IEEE (2019). He has published more than 40 papers in high-quality journals and conferences, including NeurIPS, ICML, AAAI, IJCAI, IJCNN, FUZZ-IEEE, *IEEE Transactions on Neural Networks and Learning Systems*, *IEEE Transactions on Fuzzy Systems*, etc.

Guangquan Zhang received the PhD degree in applied mathematics from the Curtin University of Technology, Perth, Australia, in 2001. He is currently a professor and director of the Decision Systems, Australian Artificial Intelligence Institute, and e-Service Intelligent (DeSI) Research Laboratory, Faculty of Engineering and Information Technology, University of Technology Sydney, Australia. His research interests include fuzzy machine learning, fuzzy optimization, and machine learning and data analytics. He has authored four monographs, five textbooks, and 350 papers including 160 referred international journal papers. He has won seven Australian Research Council (ARC) Discovery Project grants and many other research grants. He was awarded an ARC QEII Fellowship in 2005. He has served as a member of the editorial boards of several international journals, as a guest editor of eight special issues for IEEE Transactions and other international journals, and has co-chaired several international conferences and work-shops in the area of fuzzy decision-making and knowledge engineering.