Cross-Domain Structure Preserving Projection for Heterogeneous Domain Adaptation

Qian Wang, Toby P. Breckon

Department of Computer Science, Durham University, UK.
qian.wang173@hotmail.com, toby.breckon@durham.ac.uk

Abstract

Heterogeneous Domain Adaptation (HDA) addresses the transfer learning problems where data from the source and target domains are of different modalities (e.g., texts and images) or feature dimensions (e.g., features extracted with different methods). It is useful for multi-modal data analysis. Traditional domain adaptation algorithms assume that the representations of source and target samples reside in the same feature space, hence are likely to fail in solving the heterogeneous domain adaptation problem. Contemporary state-of-the-art HDA approaches are usually composed of complex optimization objectives for favourable performance and are therefore computationally expensive and less generalizable. To address these issues, we propose a novel Cross-Domain Structure Preserving Projection (CDSPP) algorithm for HDA. As an extension of the classic LPP to heterogeneous domains, CDSPP aims to learn domain-specific projections to map sample features from source and target domains into a common subspace such that the class consistency is preserved and data distributions are sufficiently aligned. CDSPP is simple and has deterministic solutions by solving a generalized eigenvalue problem. It is naturally suitable for supervised HDA but has also been extended for semi-supervised HDA where the unlabelled target domain samples are available. Extensive experiments have been conducted on commonly used benchmark datasets (i.e. Office-Caltech, Multilingual Reuters Collection, NUS-WIDE-ImageNet) for HDA as well as the Office-Home dataset firstly introduced for HDA by ourselves due to its significantly larger number of classes than the existing ones (65 vs 10, 6 and 8). The experimental results of both supervised and semi-supervised HDA demonstrate the superior performance of our proposed method against contemporary state-of-the-art methods.

Key words: heterogeneous domain adaptation, cross-domain projection, image classification, text classification

1. Introduction

Supervised learning can achieve good performance given considerable amounts of labelled data for training. One essential factor accounting for the recent successes in deep learning and image classification is the ImageNet database which contains more than 14 million hand-annotated images [8]. However, there exist many tasks in real-world applications where sufficient labelled data are not available, hence the performance of traditional supervised learning approaches can degrade significantly. One promising technique alleviating this problem is transfer learning which aims to transfer knowledge learned from the source domain to the target domain in which labelled data are sparse and expensive to collect [35]. In many scenarios, domain adaptation is required since the data distributions in the source and target domains can be different and the models trained with source domain data are not directly applicable to the target domain [25].

Since domain adaptation is a promising solution to the training data sparsity issue in many real-world applications, it has been studied in a variety of research tasks including image classification [33], semantic segmentation [41], depth estimation [2], speech emotion recognition [42], text classification [44] and many others.

Domain adaptation approaches aim to model the domain shift between source and target domains and reduce the discrepancy by aligning the data distributions [33, 32]. In the scope of classification problems, this is usually boiled down to aligning the marginal and class conditional distributions across domains [31, 3]. However, most existing works are based on the assumption of homogeneity, i.e., the source and target data are represented in the same feature space with unaligned distributions [41, 33, 40, 32]. These approaches may not be applicable in situations where the source and target domains are *heterogeneous* in the forms of data modalities (e.g., texts vs images) or representations (e.g., features extracted with different methods).

Attempts have been made to extend the success of domain adaptation approaches to the HDA problems, however, it is non-trivial for the common subspace learning methods due to the heterogeneous feature spaces across the source and target domains. One common solution to such extension is to learn two domain-specific projections instead of one unified projection for the source and target domains in HDA problems [30, 19]. Nevertheless, there are at least two limitations in these existing methods. One is most of them use Maximum Mean Discrepancy (MMD) as the objective to learn the projection matrices. MMD based objectives have been outperformed

by more recent ones based on locality preserving projection [32, 18] in homogeneous domain adaptation. In HDA problems, locality preserving objectives have not been well explored despite some attempts in [30, 19]. In this paper, we present a succinct yet effective algorithm by extending the locality preserving objectives for heterogeneous domain adaptation. The other limitation of existing HDA approaches is the way how they exploit the unlabelled target-domain data are sub-optimal. In our work, we propose a novel selective pseudo-labelling strategy to take advantage of the unlabelled target-domain data. The selection is based on the classification confidence and applies to a variety of classification models (e.g., Nearest Neighbour, SVM and Neural Networks).

Specifically, we address the heterogeneous domain adaptation problem where the source and target data are represented in heterogeneous feature spaces. Following the same spirits of previous domain adaptation approaches [31, 33, 32], we try to learn a common latent subspace where both source and target data can be projected and well aligned in the learnt subspace. Specifically, we learn domain-specific projections using a novel Cross-Domain Structure Preserving Projection (CDSPP) algorithm which is an extension of the classic Locality Preserving Projection (LPP) algorithm [13]. CDSPP can facilitate class consistency preserving to learn domain-specific projections which can be used to map heterogeneous data representations into a common subspace for recognition. CDSPP is simple yet effective in solving the heterogeneous domain adaptation problem as empirically validated by our experimental results on several benchmark datasets. To take advantage of the unlabelled target-domain data in the semi-supervised HDA setting, a selective pseudo-labelling strategy is employed to progressively optimise the projections and target data label predictions. The contributions of this work can be summarised as follows:

- A novel Cross-Domain Structure Preserving Projection algorithm is proposed for heterogeneous domain adaptation and the algorithm has a concise solution by solving a generalized eigenvalue problem;

- The proposed CDSPP algorithm is naturally for supervised HDA and we extend it to solve the semi-supervised HDA problems by employing an iterative pseudo-labelling approach;

- We validate the effectiveness of the proposed method on several benchmark datasets including the newly introduced Office-Home which contains much more classes than the previously used ones; the experimental results provide evidence our algorithm outperforms

prior art.

2. Related Work

Most exiting research in domain adaptation for classification is based on the assumption of homogeneity [32, 18, 17]. The approaches are dedicated to either learning a domain-invariant feature extraction model (e.g., deep CNN [4, 40]) or learning a unified feature projection matrix [31, 33, 32] for all domains whilst neither of them applies to HDA. In this section, we briefly review related works on heterogeneous domain adaptation. The existing approaches to HDA can be roughly categorized into *cross-domain mapping* and *common subspace learning*.

2.1. Cross-Domain Mapping

Cross-domain mapping approaches learn a projection from the source to the target domain. The projection can be learned for either *feature transformation* [15, 27] or *model parameter transformation* (e.g., SVM weights [44, 24]). Feature transformation approaches learn a projection to map the source data into the target data by aligning the data distribution [15] or the second-order moment [27]. As a result, the transformed source data can help to learn a classifier for the target domain. To avoid mapping a lower-dimensional feature to a higher-dimensional space, PCA is usually employed to learn subspaces for both domains respectively [15] as a pre-processing which can suffer from information loss.

Model parameter transformation approaches focus mainly on SVM classifier weights. For a multi-class classification problem, one-vs-all classifiers are learned for source and target domains using the respective labelled samples. Subsequently, the cross-domain mapping is learned from the paired class-level weight vectors [44, 24]. Since the number of classes is far less than the number of samples, these approaches are more computationally efficient but rely too much on the learned classifiers and overlooked abundant information underlying the data distribution.

2.2. Common Subspace Learning

Common subspace learning is a more popular strategy for HDA. It learns domain-specific projections which map source and target domain data into a common subspace. To this end, different approaches have been proposed with varying algorithms, e.g., Manifold Alignment [30, 20, 10, 36], Canonical Correlation Analysis [37], Coding Space Learning [21, 19, 9], Deep

Matrix Completion [16] and Deep Neural Networks [43, 38]. Despite the diversity of implementation, the main objective of common subspace learning based HDA is similar, i.e., the alignment of the source and target domains.

To align the distributions, [15, 21, 19, 20, 16] chose to minimize the Maximum Mean Discrepancy (MMD) in their objectives which, however, can only align the means of domains (for marginal distributions) and the means of classes (for conditional distributions). As a result, the subspace learned via minimizing the MMD is not sufficiently discriminative. One alternative to MMD is the manifold learning using graph Laplacian [30, 19, 20].

Li et al. [23] proposed a Heterogeneous Feature Augmentation (HFA) method and its semi-supervised version SHFA by learning domain-specific projections and a classifier (i.e. SVM) simultaneously. However, the computational complexity is $\mathcal{O}(n^3)$, where n is the number of labelled samples and makes it extremely slow when n is large. Li et al. [21] learned new feature representations for source and target data by encoding them with a shared codebook which requires the original features have the same dimensions for source and target domains. PCA was employed for this purpose as a pre-processing but can suffer from information loss. Lately, the authors incorporated the learning of two domain-specific projections (in place of PCA) into the coding framework [19]. This work is similar to ours in the sense of local consistency using the graph regularization, however, it fails to align cross-domain class consistency due to the use of k nearest neighbours to construct the similarity graph. In our work, the similarity graph is constructed based on class consistency, hence promoting the cross-domain conditional distribution alignment.

Transfer Independently Together (TIT) was proposed in [20]. It also learns domain-specific projections to align data distributions in the learned common subspace. The algorithm was based on a collection of tricks including kernel space, MMD, sample reweighting and landmark selection. In contrast, our solution is concise with one simple objective of cross-domain structure preserving. Recently, Huang et al. [14] proposed a novel algorithm, named heterogeneous discriminative features learning and label propagation (HDL). This algorithm is similar to ours in that both tend to preserve structure information in the learned common subspace. However, different objectives have been formulated. Our algorithm explicitly promotes the intra-class similarity for both within-domain and cross-domain samples, whilst HDL fails to consider the intra-class similarity for samples from the same domain in their formulation. In addition, different

strategies of unlabelled target sample exploration were employed in two algorithms.

In summary, although manifold learning has been well studied in HDA, the existing formulations for domain-specific projection learning are either inefficient or ineffective. Our approach solves this issue and addresses the HDA problem with a novel CDSPP algorithm.

3. Method

To facilitate our presentation, we firstly describe the heterogeneous domain adaptation problem and notations used throughout this paper. Given a labelled dataset $\mathcal{D}^s = \{(\boldsymbol{x}_i^s, y_i^s)\}, i = 1,2,...,n_s$ from the source domain $\mathcal{S}$, and a labelled dataset $\mathcal{D}^t = \{\boldsymbol{x}_i^t, y_i^t\}, i = 1,2,...,n_t$ from the target domain, $\boldsymbol{x}_i^s \in \mathbb{R}^{d_s}$ and $\boldsymbol{x}_i^t \in \mathbb{R}^{d_t}$ represent the feature vectors of i-th labelled samples in the source and target domains respectively; d_s and d_t are the dimensionalities of the source and target features; $y_i^s \in \mathcal{Y}$ and $y_i^t \in \mathcal{Y}$ denote the corresponding sample labels; n_s and n_t are the number of source and labelled target samples respectively. Let $\boldsymbol{X}^s \in \mathbb{R}^{d_s \times n_s}$ and $\boldsymbol{X}^t \in \mathbb{R}^{d_t \times n_t}$ be the feature matrices of labelled source and target data collectively, supervised HDA aims to learn a model from labelled source and target data, which can be used to classify samples from an unlabelled dataset $\mathcal{D}^u = \{\boldsymbol{x}_i^u\}, i = 1,2,...,n_u$ from the target domain, whose feature vectors can be collectively denoted as $\boldsymbol{X^u} \in \mathbb{R}^{d_t \times n_u}$.

The number of labelled target samples n_t is usually very small, hence it is difficult to capture the data distribution in the target domain. Semi-supervised HDA takes advantage of the unlabelled target samples $\boldsymbol{X^u}$ during model training and can usually achieve better performance.

In this section, we describe the CDSPP algorithm which is naturally for supervised heterogeneous domain adaptation but can be used to address the semi-supervised heterogeneous domain adaptation problem by incorporating it into an iterative learning framework [33, 32] as shown in Figure 1.

3.1. Locality Preserving Projection

To make the paper self-contained, we briefly describe the original LPP algorithm [13] before introducing our proposed CDSPP in the next subsection. Locality Preserving Projection (LPP) was proposed by He and Niyogi [13] to learn a favourable subspace where the local structures of data in the original feature space can be well preserved. Suppose $\boldsymbol{x}_i \in \mathbb{R}^{d_0}$ and $\boldsymbol{x}_j \in \mathbb{R}^{d_0}$ are two data points in the original feature space, LPP aims at learning a projection matrix $\boldsymbol{P} \in \mathbb{R}^{d \times d_0}$

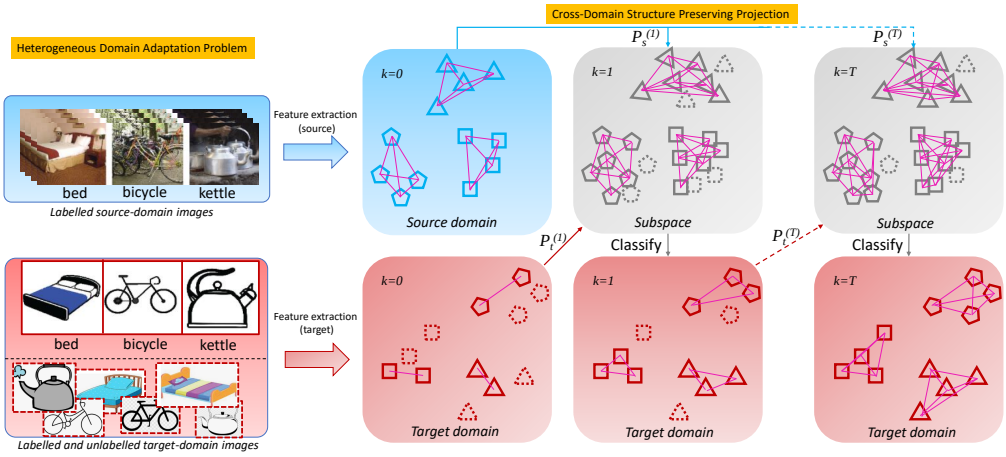

Figure 1: An illustration of the heterogeneous domain adaptation problem and our proposed approach using cross-domain structure preserving projection. Left: the HDA problem aims at recognizing unlabelled target-domain samples with the access of labelled source-domain samples and limited labelled target-domain samples. Right: The red and the blue colours are used to represent the feature vectors of samples in the target and source domains respectively; markers of different shapes represent samples from different classes; dashed markers represent unlabelled samples; our proposed CDSPP iteratively learn a common subspace in which the unlabelled target-domain samples are pseudo-labelled and selectively added to the training data set to promote the subspace learning in the next iteration.

($d << d_0$) so that data points close to each other in the original space will still be close in the projected subspace. The objective of LPP can be formulated as:

$$\min_{\boldsymbol{P}} \sum_{i,j} ||\boldsymbol{P}^T \boldsymbol{x}_i - \boldsymbol{P}^T \boldsymbol{x}_j||_2^2 \boldsymbol{W}_{ij}, \tag{1}$$

where $\boldsymbol{W}$ is the adjacency matrix of the graph constructed by all the data points. According to [13], the edges of the graph can be created by either $\varepsilon-$neighbourhoods or k-nearest neighbours. The edge weights can be determined by the heat kernel $W_{ij} = e^{-\frac{||\boldsymbol{x}_i - \boldsymbol{x}_j||^2}{t}}$ or the simple binary assignment (i.e. all edges have the weights of 1). Note that LPP is an unsupervised learning method without the need for labelling information. In the following subsection, we will describe how to extend the LPP algorithm to solve the HDA problems where there exist two heterogeneous domains and a mixture of labelled and unlabelled data.

3.2. Cross-Domain Structure Preserving Projection

The supervised version of LPP [34] was proved to be able to learn a subspace of better separability than other dimensionality reduction algorithms such as Linear Discriminant Analysis

(LDA) [33]. One limitation of LPP is that it can only learn the subspace from samples represented in a homogeneous feature space. To address this problem, we extend the traditional LPP so that its favourable characteristics can benefit cross-domain common subspace learning. Specifically, we aim to learn a projection matrix $\boldsymbol{P}_s \in \mathbb{R}^{d_s \times d}$ for the source domain and a projection matrix $\boldsymbol{P}_t \in \mathbb{R}^{d_t \times d}$ for the target domain to project the samples from source and target domains into a common subspace whose dimensionality is d. We expect the samples projections are close to one another if they are from the same class regardless of which domain they are from. To this end, we have the following objective:

$$\begin{aligned} \min_{\boldsymbol{P}_s, \boldsymbol{P}_t} \quad & (\textstyle\sum_{i,j}^{n_s} ||\boldsymbol{P}_s^T \boldsymbol{x}_i^s - \boldsymbol{P}_s^T \boldsymbol{x}_j^s||_2^2 \boldsymbol{W}_{ij}^s \\ & + \textstyle\sum_i^{n_s} \sum_j^{n_t} ||\boldsymbol{P}_s^T \boldsymbol{x}_i^s - \boldsymbol{P}_t^T \boldsymbol{x}_j^t||_2^2 \boldsymbol{W}_{ij}^c \\ & + \textstyle\sum_{i,j}^{n_t} ||\boldsymbol{P}_t^T \boldsymbol{x}_i^t - \boldsymbol{P}_t^T \boldsymbol{x}_j^t||_2^2 \boldsymbol{W}_{ij}^t) \end{aligned} \tag{2}$$

where $\boldsymbol{P}^T$ is the transpose of $\boldsymbol{P}$; $\boldsymbol{W}^s \in \mathbb{R}^{n_s \times n_s}$ is the similarity matrix of the source samples and $\boldsymbol{W}_{ij}^s = 1$ if $y_i^s = y_j^s$, 0 otherwise. Similary, $\boldsymbol{W}^t \in \mathbb{R}^{n_t \times n_t}$ is the similarity matrix of the *labelled* target samples and $\boldsymbol{W}_{ij}^t = 1$ if $y_i^t = y_j^t$, 0 otherwise. $\boldsymbol{W}^c \in \mathbb{R}^{n_s \times n_t}$ is the cross-domain similarity matrix and $\boldsymbol{W}_{ij}^c = 1$ if $y_i^s = y_j^t$, 0 otherwise. It is noteworthy that all the feature vectors are $l2$-normalised to get rid of the effect of different magnitudes across features. This pre-processing has been proved to be useful for common subspace learning in [34, 33, 32].

Proposition 3.1. *The objective in Eq.(2) can be reformulated as follows:*

$$\max_{\boldsymbol{P}_s, \boldsymbol{P}_t} \frac{tr(\boldsymbol{X}^{sT} \boldsymbol{P}_s \boldsymbol{P}_t^T \boldsymbol{X}^t \boldsymbol{W}^{cT})}{tr(\boldsymbol{X}^{sT} \boldsymbol{P}_s \boldsymbol{P}_s^T \boldsymbol{X}^s \boldsymbol{L}^s) + tr(\boldsymbol{X}^{tT} \boldsymbol{P}_t \boldsymbol{P}_t^T \boldsymbol{X}^t \boldsymbol{L}^t)} \tag{3}$$

where $\boldsymbol{L}^s = \boldsymbol{D}^s - \boldsymbol{W}^s + \frac{1}{2}\boldsymbol{D}^{cs}$ *and* $\boldsymbol{L}^t = \boldsymbol{D}^t - \boldsymbol{W}^t + \frac{1}{2}\boldsymbol{D}^{ct}$*;* $\boldsymbol{D}^s \in \mathbb{R}^{n_s \times n_s}$ *is a diagonal matrix with* $\boldsymbol{D}_{ii}^s = \sum_j^{n_s} \boldsymbol{W}_{ij}^s$ *and* $\boldsymbol{D}^t \in \mathbb{R}^{n_t \times n_t}$ *is a diagonal matrix with* $\boldsymbol{D}_{jj}^t = \sum_i^{n_t} \boldsymbol{W}_{ij}^t$*;* $\boldsymbol{D}^{cs} \in \mathbb{R}^{n_s \times n_s}$ *is a diagonal matrix with* $\boldsymbol{D}_{ii}^{cs} = \sum_j^{n_t} \boldsymbol{W}_{ij}^c$ *and* $\boldsymbol{D}^{ct} \in \mathbb{R}^{n_t \times n_t}$ *is a diagonal matrix with* $\boldsymbol{D}_{jj}^{ct} = \sum_i^{n_s} \boldsymbol{W}_{ij}^c$*.*

Proof. By firstly doing the binomial expansion then transforming it to the form of matrix multiplication and trace of matrices, the first term in Eq.(2) can be reformulated as follows:

$$\begin{aligned} & \textstyle\sum_{i,j}^{n_s} ||\boldsymbol{P}_s^T \boldsymbol{x}_i^s - \boldsymbol{P}_s^T \boldsymbol{x}_j^s||_2^2 \boldsymbol{W}_{ij}^s \\ & = \textstyle\sum_{i,j}^{n_s} (\boldsymbol{x}_i^{sT} \boldsymbol{P}_s \boldsymbol{P}_s^T \boldsymbol{x}_i^s - 2\boldsymbol{x}_i^{sT} \boldsymbol{P}_s \boldsymbol{P}_s^T \boldsymbol{x}_j^s + \boldsymbol{x}_j^{sT} \boldsymbol{P}_s \boldsymbol{P}_s^T \boldsymbol{x}_j^s) \boldsymbol{W}_{ij}^s \\ & = 2\textstyle\sum_i^{n_s} \boldsymbol{x}_i^{sT} \boldsymbol{P}_s \boldsymbol{P}_s^T \boldsymbol{x}_i^s \boldsymbol{D}_{ii}^s - 2\sum_{i,j}^{n_s} \boldsymbol{x}_i^{sT} \boldsymbol{P}_s \boldsymbol{P}_s^T \boldsymbol{x}_j^s \boldsymbol{W}_{ij}^s \\ & = 2tr(\boldsymbol{X}^{sT} \boldsymbol{P}_s \boldsymbol{P}_s^T \boldsymbol{X}^s \boldsymbol{D}^s) - 2tr(\boldsymbol{X}^{sT} \boldsymbol{P}_s \boldsymbol{P}_s^T \boldsymbol{X}^s \boldsymbol{W}^s) \end{aligned} \tag{4}$$

In the similar way, the third term in Eq.(2) can be rewritten as:

$$\begin{aligned}&\textstyle\sum_{i,j}^{n_t} ||\boldsymbol{P}_t^T \boldsymbol{x}_i^t - \boldsymbol{P}_t^T \boldsymbol{x}_j^t||_2^2 \boldsymbol{W}_{ij}^t \\ &= 2tr(\boldsymbol{X}^{tT} \boldsymbol{P}_t \boldsymbol{P}_t^T \boldsymbol{X}^t \boldsymbol{D}^t) - 2tr(\boldsymbol{X}^{tT} \boldsymbol{P}_t \boldsymbol{P}_t^T \boldsymbol{X}^t \boldsymbol{W}^t)\end{aligned} \tag{5}$$

The second term in Eq.(2) can be rewritten as:

$$\begin{aligned}&\textstyle\sum_i^{n_s} \sum_j^{n_t} ||\boldsymbol{P}_s^T \boldsymbol{x}_i^s - \boldsymbol{P}_t^T \boldsymbol{x}_j^t||_2^2 \boldsymbol{W}_{ij}^c \\ &= \textstyle\sum_i^{n_s} \sum_j^{n_t} (\boldsymbol{x}_i^{sT} \boldsymbol{P}_s \boldsymbol{P}_s^T \boldsymbol{x}_i^s - 2\boldsymbol{x}_i^{sT} \boldsymbol{P}_s \boldsymbol{P}_t^T \boldsymbol{x}_j^t \\ &\qquad + \boldsymbol{x}_j^{tT} \boldsymbol{P}_t \boldsymbol{P}_t^T \boldsymbol{x}_j^t) \boldsymbol{W}_{ij}^c \\ &= \textstyle\sum_i^{n_s} \boldsymbol{x}_i^{sT} \boldsymbol{P}_s \boldsymbol{P}_s^T \boldsymbol{x}_i^s \boldsymbol{D}_{ii}^{cs} - 2\sum_i^{n_s} \sum_j^{n_t} \boldsymbol{x}_i^{sT} \boldsymbol{P}_s \boldsymbol{P}_t^T \boldsymbol{x}_j^t \boldsymbol{W}_{ij}^c \\ &\qquad + \textstyle\sum_j^{n_t} \boldsymbol{x}_j^{tT} \boldsymbol{P}_t \boldsymbol{P}_t^T \boldsymbol{x}_j^t \boldsymbol{D}_{jj}^{ct} \\ &= tr(\boldsymbol{X}^{sT} \boldsymbol{P}_s \boldsymbol{P}_s^T \boldsymbol{X}^s \boldsymbol{D}^{cs}) - 2tr(\boldsymbol{X}^{sT} \boldsymbol{P}_s \boldsymbol{P}_t^T \boldsymbol{X}^t \boldsymbol{W}^{cT}) \\ &\qquad + tr(\boldsymbol{X}^{tT} \boldsymbol{P}_t \boldsymbol{P}_t^T \boldsymbol{X}^t \boldsymbol{D}^{ct})\end{aligned} \tag{6}$$

Substitute Eqs.(4-6) into the objective Eq.(2), we have the following form of objective:

$$\begin{aligned}&\min_{\boldsymbol{P}_s, \boldsymbol{P}_t} \big(tr(\boldsymbol{X}^{sT} \boldsymbol{P}_s \boldsymbol{P}_s^T \boldsymbol{X}^s \boldsymbol{L}^s) + tr(\boldsymbol{X}^{tT} \boldsymbol{P}_t \boldsymbol{P}_t^T \boldsymbol{X}^t \boldsymbol{L}^t) \\ &- tr(\boldsymbol{X}^{sT} \boldsymbol{P}_s \boldsymbol{P}_t^T \boldsymbol{X}^t \boldsymbol{W}^{cT})\big)\end{aligned} \tag{7}$$

where $\boldsymbol{L}^s = \boldsymbol{D}^s - \boldsymbol{W}^s + \frac{1}{2}\boldsymbol{D}^{cs}$ and $\boldsymbol{L}^t = \boldsymbol{D}^t - \boldsymbol{W}^t + \frac{1}{2}\boldsymbol{D}^{ct}$.

Minimizing the objective in Eq.(7) is equivalent to maximizing the objective in Eq.(3).

□

Proposition 3.2. *The objective in Eq.(3) is equivalent to the following generalized eigenvalue problem and the optimal projection matrix* $\boldsymbol{P} = \begin{bmatrix} \boldsymbol{P}_s \\ \boldsymbol{P}_t \end{bmatrix}$ *can be formed by d eigenvectors corresponding to the largest d eigenvalues:*

$$\boldsymbol{A}\boldsymbol{P} = (\boldsymbol{B} + \alpha \boldsymbol{I})\boldsymbol{P}\Lambda \tag{8}$$

where $\boldsymbol{I} \in \mathbb{R}^{(n_s+n_t)\times(n_s+n_t)}$ *is an identity matrix,* α *is a hyper-parameter for regularization [34],* Λ *is a diagonal eigenvalue matrix and*

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{X}^s \boldsymbol{W}^c \boldsymbol{X}^{tT} \\ \boldsymbol{X}^t \boldsymbol{W}^{cT} \boldsymbol{X}^{sT} & \boldsymbol{0} \end{bmatrix}, \tag{9}$$

$$\boldsymbol{B} = \begin{bmatrix} \boldsymbol{X}^s \boldsymbol{L}^s \boldsymbol{X}^{sT} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{X}^t \boldsymbol{L}^t \boldsymbol{X}^{tT} \end{bmatrix}. \tag{10}$$

Proof. To make the proof process concise, we introduce notations $\boldsymbol{S}_s = \boldsymbol{X}^s \boldsymbol{L}^s \boldsymbol{X}^{sT}$, $\boldsymbol{S}_t = \boldsymbol{X}^t \boldsymbol{L}^t \boldsymbol{X}^{tT}$ and $\boldsymbol{S}_c = \boldsymbol{X}^s \boldsymbol{W}^c \boldsymbol{X}^{tT}$.

Let

$$\mathcal{J}(\boldsymbol{P}_s, \boldsymbol{P}_t) = \frac{tr(\boldsymbol{P}_t^T \boldsymbol{S}_c^T \boldsymbol{P}_s)}{tr(\boldsymbol{P}_s^T \boldsymbol{S}_s \boldsymbol{P}_s) + tr(\boldsymbol{P}_t^T \boldsymbol{S}_t \boldsymbol{P}_t)} \tag{11}$$

be the objective function in Eq.(3), we calculate the partial derivatives [26] of $\mathcal{J}$ w.r.t. $\boldsymbol{P}_s$ and $\boldsymbol{P}_t$ respectively, set them to 0 and get the following equations:

$$\boldsymbol{S}_c \boldsymbol{P}_t = \frac{2tr(\boldsymbol{P}_t^T \boldsymbol{S}_c \boldsymbol{P}_s)}{tr(\boldsymbol{P}_s^T \boldsymbol{S}_s \boldsymbol{P}_s) + tr(\boldsymbol{P}_t^T \boldsymbol{S}_t \boldsymbol{P}_t)} \boldsymbol{S}_s \boldsymbol{P}_s \tag{12}$$

$$\boldsymbol{S}_c^T \boldsymbol{P}_s = \frac{2tr(\boldsymbol{P}_t^T \boldsymbol{S}_c \boldsymbol{P}_s)}{tr(\boldsymbol{P}_s^T \boldsymbol{S}_s \boldsymbol{P}_s) + tr(\boldsymbol{P}_t^T \boldsymbol{S}_t \boldsymbol{P}_t)} \boldsymbol{S}_t \boldsymbol{P}_t \tag{13}$$

Note that the coefficients on the right side of Eqs(12-13) are exactly the objective in Eq.(11). It is easy to construct the following generalized eigenvalue problem by combining Eqs.(12-13):

$$\begin{bmatrix} \boldsymbol{0} & \boldsymbol{S}_c \\ \boldsymbol{S}_c^T & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{P}_s \\ \boldsymbol{P}_t \end{bmatrix} = \begin{bmatrix} \boldsymbol{S}_s & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{S}_t \end{bmatrix} \begin{bmatrix} \boldsymbol{P}_s \\ \boldsymbol{P}_t \end{bmatrix} \Lambda. \tag{14}$$

The maximum objective is given by the largest eigenvalue solution to the generalized eigenvalue problem [13] and the eigenvectors corresponding to the largest d eigenvalues will form the projection matrix $\boldsymbol{P}_s$ and $\boldsymbol{P}_t$.

□

3.3. Recognition in the Subspace

Once the projection matrices $\boldsymbol{P}_s$ and $\boldsymbol{P}_t$ are learned, we are able to project all the labelled samples into the learned common subspace by $\boldsymbol{z}_i^s = \boldsymbol{P}_s^T \boldsymbol{x}_i^s$ and $\boldsymbol{z}_i^t = \boldsymbol{P}_t^T \boldsymbol{x}_i^t$. Similar to the pre-processing for the training data, the feature vectors $\boldsymbol{x}$ need to be $l2$-normalised before being projected to the subspace. For the same reason, we also apply $l2$-normalisation to the projected vectors $\boldsymbol{z}$. The $l2$-normalisation re-allocates the projected vectors in the subspace to the surface of a hyper-sphere which will benefit the measurement of distances when do the recognition using the nearest neighbour method. More importantly, the $l2$-normalisation adds non-linearity to the process so that our proposed CDSPP method can handle practical problems when linear projection assumptions do not hold.

For each class, we calculate the class mean $\bar{\boldsymbol{z}}_c$ for $c = 1, 2, ..., C$ using all the labelled sample from both source and target domains. Given an unlabelled target sample $\boldsymbol{x}^u$, we classify it to the closest class in terms of its Euclidean distances to the class means:

$$y^* = \arg\min_c d(\bar{\boldsymbol{z}}_c, \boldsymbol{P}_t^T \boldsymbol{x}^u) \tag{15}$$

The proposed CDSPP for supervised HDA is summarized in Algorithm 1.

Relation to DAMA The CDSPP algorithm is quite similar to DAMA proposed in [30] at the first glance, however, they are essentially different from each other in that CDSPP does not seek to push the sample projections belonging to different classes apart, since the penalty imposed for this purpose (e.g., maximizing the term B in [30]) might misguide the solution to focus too much on the separation of classes which are originally close to each other and hurt the overall separability of the learned subspace. In contrast, our objective in Eq.(2) can guarantee the separability of the learned subspace by promoting the preserving of cluster structures underlying the original data distributions, which is simpler but more effective as validated by experiments.

Algorithm 1 Supervised HDA using CDSPP

Input: labelled source data set $\mathcal{D}^s = \{(\boldsymbol{x}_i^s, y_i^s)\}, i = 1, 2, ..., n_s$ and labelled target data set $\mathcal{D}^t = \{\boldsymbol{x}_i^t, y_i^t\}, i = 1, 2, ..., n_t$, the dimensionality of subspace d.

Output: The projection matrix $\boldsymbol{P}_s$ and $\boldsymbol{P}_t$ for source and target domains, the labels predicted for unlabelled target data $\boldsymbol{X}^u$.

Training:

1: Learn the projection $\boldsymbol{P}_s$ and $\boldsymbol{P}_t$ using labelled data $\mathcal{D}^s \cup \mathcal{D}^t$ by solving the generalized eigenvalue problem in Eq.(8);

Testing:

2: Classify unlabelled target samples $\boldsymbol{X}^u$ using Eq.(15).

3.4. Extending to Semi-Supervised HDA

The CDSPP algorithm is naturally suitable for supervised HDA but can be extended to semi-supervised HDA by incorporating it into an iterative pseudo-labelling framework [33]. Given a set of unlabelled target samples $\boldsymbol{X}^u$, they can be labelled by Eq.(15). The pseudo-labelled target samples can be used to update the projection matrices $\boldsymbol{P}_s$ and $\boldsymbol{P}_t$. However, when the domain

Algorithm 2 Semi-supervised HDA using CDSPP

Input: labelled source data set $\mathcal{D}^s = \{(\boldsymbol{x}_i^s, y_i^s)\}, i = 1,2,...,n_s$, labelled target data set $\mathcal{D}^t = \{\boldsymbol{x}_i^t, y_i^t\}, i = 1,2,...,n_t$, unlabelled target data set $\mathcal{D}^u = \{\boldsymbol{x}_i^u\}, i = 1,2,...,n_u$ the dimensionality of subspace d, number of iteration T.

Output: The projection matrix $\boldsymbol{P}_s$ and $\boldsymbol{P}_t$ for source and target domains, the labels predicted for unlabelled target data $\boldsymbol{X}^u$.

Training:

1: Initialize k=1;

2: Learn the projection $\boldsymbol{P}_s^{(0)}$ and $\boldsymbol{P}_t^{(0)}$ using labelled data $\mathcal{D}^s \cup \mathcal{D}^t$ by solving the generalized eigenvalue problem in Eq.(8);

3: Get the unlabelled target data set $\mathcal{D}^u$;

4: **while** $k \leq T$ **do**

5: Label all the samples from $\mathcal{D}^u$ by Eq.(15);

6: Select a subset of (top kn_u/T most confident) pseudo-labelled target samples $\mathcal{S}^{(k)} \subseteq \mathcal{D}^u$;

7: Learn $\boldsymbol{P}_s^{(k)}$ and $\boldsymbol{P}_t^{(k)}$ using a combination of labelled and pseudo-labelled data sets $\mathcal{D}^s \cup \mathcal{D}^t \cup \mathcal{S}^{(k)}$;

8: $k \leftarrow k+1$;

9: **end while**

Testing:

10: Classify unlabelled target samples $\boldsymbol{X}^u$ using Eq.(15).

shift is large and the number of labelled target samples is limited, the pseudo-labels can be wrong for a considerable number of target samples. In this case, the mistakenly pseudo-labelled target samples might hurt projection learning. To reduce this risk, confidence aware pseudo-labelling is proposed in [33]. We employ the same idea and progressively select the most confidently pseudo-labelled target samples for the next iteration of CDSPP learning. The proposed CDSPP for semi-supervised HDA is summarized in Algorithm 2.

3.5. Complexity Analysis

The time complexity of CDSPP is mainly contributed by two parts: the matrix multiplications in Eqs.(9-10) and the eigen decomposition problem. The complexity of matrix multiplications is $\mathcal{O}((n_s+n_t)d_s d_t)$. The complexity of eigen decomposition is generally $\mathcal{O}((d_s+d_t)^3)$. As a result, the CDSPP algorithm has a complexity of $\mathcal{O}((n_s+n_t)d_s d_t+(d_s+d_t)^3)$. In the case of semi-supervised HDA, the time complexity will be increased by T times and the value of n_t increases by the number of selected pseudo-labelled target samples in each iteration.

4. Experiments

To evaluate the effectiveness of the proposed method in heterogeneous domain adaptation, we conduct thorough experiments on commonly used benchmark datasets. We compare the proposed approach with existing HDA methods and analyze its sensitivity to hyper-parameters.

4.1. Datasets and Experimental Settings

Office-Caltech [11] is an image dataset containing four domains: Amazon (A), Webcam (W), DSLR (D) and Caltech (C) from 10 common classes. Two image features (i.e. 4096-dim Decaf_6 and 800-dim SURF) are used for cross-domain adaptation. **Multilingual Reuters Collection (MRC)** [1] is a cross-lingual text classification dataset containing 6 classes in 5 languages (i.e. EN, FR, GE, IT, SP). We follow the settings in [15] extracting BoW features and applying PCA to get heterogeneous feature dimensions (i.e. 1131, 1230, 1417, 1041, 807 respectively) for five domains. In our experiments, SP serves as the target domain and the other four languages as the source domains respectively. As a result, we have four HDA tasks. **NUS-WIDE** [6] and **ImageNet** [8] datasets are employed for text to image domain adaptation. Following [5] we consider 8 overlapping classes using tag information represented by 64-dim features from

Table 1: The statistics of datasets (notations: LSS/c – labelled Source Samples per class; LTS/c – labelled Target Sample per class; UTS/c – Unlabelled Target Samples per class; all – all samples except the ones chosen as labelled target samples).

Dataset	# Domain	# Task	# Class	# LSS/c	# LTS/c	# UTS/c
Office-Caltech	4	16	10	20	3	all
MRC	5	4	6	100	10	500
NUS-ImageNet	2	1	8	100	3	100
Office-Home	4	16	65	20	3	all

NUS-WIDE as the source domain and 4096-dim Decaf_6 features of images from ImageNet as the target domain. However, the above datasets contain very limited numbers of classes and may not discriminate capabilities of different methods. We introduce **Office-Home** [29] containing four domains (i.e. Art, Clipart, Product and Real-world) as a new testbed for HDA. We use VGG16 [28] and ResNet50 [12] models pre-trained on ImageNet to extract 4096-dim and 2048-dim features. More details of the datasets and protocols used in our experiments are summarized in Table 1.

4.2. Comparative Methods

To evaluate the effectiveness of the proposed CDSPP in different HDA problems, we conduct a comparative study and compare the performance of CDSPP with state-of-the-art methods in both supervised and semi-supervised settings. Specifically, we compare with SVM_t, HFA [23], CDLS_sup [15] and a variant of DAMA [30] under the supervised HDA setting (i.e. the unlabelled target samples are not available during training).

- SVM_t is a baseline method that trains an SVM model on the target dataset $\mathcal{D}^t$ in a conventional supervised learning manner without using the source domain data.

- HFA (Heterogeneous Feature Augmentation [23]) is designed to solve the supervised HDA problem by augmenting the original features $\boldsymbol{x}^s, \boldsymbol{x}^t$ with transformed features $\boldsymbol{P}\boldsymbol{x}^s$, $\boldsymbol{Q}\boldsymbol{x}^t$ and zero vectors. The projection matrices $\boldsymbol{P}$ and $\boldsymbol{Q}$ for the source and target domains map the original features into a common subspace so that the similarity of features across two domains can be directly compared. The objective of learning $\boldsymbol{P}$ and $\boldsymbol{Q}$ is incorporated into the framework of classifier (i.e. SVM) training.

- CDLS_*sup* (Cross-Domain Landmark Selection [15]) is the supervised version of CDLS which aims to learn a projection matrix $\boldsymbol{A}$ to map source-domain data into the target domain. The objective is to align the cross-domain marginal and conditional data distributions by minimizing the Maximum Mean Discrepancy (MMD).

- DAMA_*sup* (Domain Adaptation Using Manifold Alignment [30]) is originally designed for semi-supervised HDA problems. Similar to our proposed CDSPP, it also aims to learn two projection matrices to map source and target domain data to a common subspace where the manifolds of data from two domains are aligned. We adapt it for supervised HDA by considering only labelled data when constructing the feature similarity matrix $\boldsymbol{W}$, the label based similarity matrix $\boldsymbol{W}^s$ and dissimilarity matrix $\boldsymbol{W}^d$. Different from the suggestion in the original paper, we use an optimal $\mu = 0.1$ throughout our experiments since this setting achieves the best performance.

For semi-supervised HDA, we compare with DAMA [30], SHFA [23], CDLS [15], PA [19], TIT [20], STN [38], DDACL[39], SSAN [22] and DAMA+, our extension of DAMA by incorporating it into our iterative learning framework (c.f. Section 3.4).

- DAMA [30] is employed in the semi-supervised HDA experiments in its original form except the hyper-parameter μ is set as 0.1 as our experimental results show empirically it gives the optimal performance.

- SHFA (Semi-supervised HFA [23]) is an extension of HFA. It takes advantage of the unlabelled target-domain data by replacing the SVM in HFA with a Transductive SVM (T-SVM) [7] model.

- CDLS [15] is designed for semi-supervised HDA. As described above, it aims to learn a projection matrix $\boldsymbol{A}$ to map source-domain data into the target domain so that cross-domain data can be aligned. When unlabelled target-domain data are available in the semi-supervised HDA, the unlabelled data are pseudo-labelled by the supervised version CDLS_*sup*. Subsequently, the pseudo-labelled data are used to update the projection $\boldsymbol{A}$. The processes are repeated for multiple iterations. In particular, the instances are weighted by learnable weights when constructing the objective function.

- PA (Progressive Alignment [19]) and TIT (Transfer Independently Together [20]) share

a similar framework to CDLS but employ different algorithms of transformation matrix learning (involving MMD, graph embedding and regularisation) and different instance weight estimation strategies. The unlabelled target-domain data are also pseudo-labelled to optimize the transformation matrices iteratively.

- STN (Soft Transfer Network [38]) jointly learns a domain-shared classifier and a domain-invariant subspace in an end-to-end manner. The network model is learned by optimising the objective similar to those in the aforementioned works, i.e., MMD. Besides, the unlabelled target-domain data are used by the soft-label strategy.

- DDACL (Discriminative Distribution Alignment with Cross-entropy Loss [39]) trains an adaptive classifier by both reducing the distribution divergence and enlarging distances between class centroids.

- SSAN (Simultaneous Semantic Alignment Network [22]) employs an implicit semantic correlation loss to transfer the correlation knowledge of source categorical prediction distributions to the target domain. A triplet-centroid alignment mechanism is explicitly applied to align feature representations for each category by leveraging target pseudo-labels. Note that the results of best accuracy of the test samples throughout the training process were reported in [22], we argue that this is not achievable in practice since the labels of test samples are not available during training. Instead, we report the results achieved in the last iterations in our experiments.

- DAMA+ is our adaptation of the original DAMA by incorporating the DAMA algorithm into our proposed iterative learning framework with selective pseudo-labelling. Specifically, we use the supervised version of DAMA described above to initialise the projection matrices and get the pseudo-labels of unlabelled target-domain data. The selected most confidently pseudo-labelled target-domain data will contribute to the update of projection matrices in the next iteration of learning. Finally, the optimal projection matrices and predicted target-domain data labels are obtained.

- CDSPP+PCA is a variant of CDSPP by applying PCA to the original features and CDSPP is subsequently applied to the low-dimensional features. This pre-processing is specially designed for handcrafted features in the MRC and NUS-ImageNet datasets and 50 principal components are reserved for all features.

In all experiments, we use the optimal parameters suggested in the original papers for the comparative methods if not otherwise specified whilst set the hyper-parameters of CDSPP empirically as d equal to the number of classes in the dataset, $\alpha = 10$ and $T = 5$. More details of hyper-parameter value selection will be discussed later.

4.3. Comparison Results

Although there exist fixed experimental protocols in terms of the number of labelled samples used for training as shown in Table 1, there is no standard data splits publicly available to follow. As will be demonstrated in our experimental results, selecting different samples for training can lead to significant performance variance. We generate data splits randomly in our experiments[1]. To mitigate the biases caused by the data selection, ten random data splits are generated for each adaptation task. We report the mean and standard deviation of the classification accuracy over these ten trials for each adaptation task. The results for all comparative methods are reproduced using the same data splits for a direct comparison. The implementations released by the authors are employed in our experiments. As a result, the results in this paper are not comparable with those reported in other papers since different sample selections have been used in our experiments. Our experimental results of both supervised and semi-supervised HDA on four datasets are shown in Tables 2-5 from which we can obtain the following insights.

Table 2 (except the last column) lists the comparison results on the MRC dataset. The baseline method SVM_t achieves an accuracy of 67.0% using only 10 labelled target domain samples per class for training. The labelled source domain data can benefit the performance with proper domain adaptation but the improvement is marginal for both HFA and our proposed CDSPP. The supervised version of CDLS uses PCA to learn a subspace from the target domain, hence the dimensionality of subspace cannot be higher than $n_t - 1$. Due to such limitation, CDLS_sup performs worse than others when the number of labelled target samples is small which is usually the case for HDA problems. For the semi-supervised HDA, DAMA and SHFA perform no better than the baseline method SVM_t which was also observed in existing works [15, 19, 20]. The best performance (71.7%) is achieved by PA [19] and our proposed CDSPP is marginally worse with the average classification accuracy of 68.9%. However, when applying PCA to reduce the text features to a lower dimensionality of 50, the performance of CDSPP is improved from 68.9%

[1]The data splits and code are released: https://github.com/hellowangqian/cdspp-hda

Table 2: Mean(std) of classification accuracy (%) over ten trials for cross-language and tag-to-image adaptation under supervised (denoted by $*$) and semi-supervised settings (each column represents one Source $\rightarrow$ Target adaptation task).

Method	EN→SP	FR→SP	GE→SP	IT→SP	Avg	Tag→Image
SVM_t *	67.0(2.4)	67.0(2.4)	67.0(2.4)	67.0(2.4)	67.0	60.6(6.0)
HFA [23] *	68.1(3.0)	68.0(3.0)	68.0(3.0)	68.0(3.0)	68.0	67.5(2.5)
CDLS_sup [15] *	63.0(3.6)	63.4(2.4)	64.0(2.2)	64.6(3.6)	63.8	66.3(3.9)
DAMA_sup *	66.8(2.5)	66.3(3.3)	66.3(3.0)	66.7(2.7)	66.5	66.9(2.6)
CDSPP_sup (Ours) *	67.2(2.8)	67.3(2.9)	67.3(2.9)	67.3(2.8)	67.3	67.2(3.0)
DAMA [30]	67.0(2.5)	66.6(3.1)	66.7(3.0)	67.4(2.8)	66.9	67.0(2.5)
SHFA [23]	66.9(3.7)	66.1(2.7)	67.5(3.1)	67.4(2.2)	67.0	68.1(2.7)
CDLS [15]	69.4(3.0)	69.4(3.0)	69.4(3.2)	69.3(3.1)	69.4	69.6(2.1)
PA [19]	**71.4(2.9)**	**71.6(2.9)**	**71.7(3.0)**	**72.3(2.5)**	**71.7**	70.5(4.0)
TIT [20]	67.1(2.8)	67.6(2.6)	66.1(3.5)	67.8(2.0)	67.2	70.7(3.4)
STN [38]	67.1(3.6)	67.3(2.5)	66.9(3.5)	66.7(3.8)	67.0	74.3(5.2)
DDACL [39]	70.2(3.0)	70.4(3.1)	70.8(3.0)	70.9(3.0)	70.6	73.8(2.8)
SSAN [22]	69.9(2.9)	69.4(2.8)	69.3(4.0)	70.2(2.5)	69.7	71.4(1.2)
DAMA +	68.9(2.1)	68.8(4.0)	68.9(2.7)	68.2(3.5)	68.7	73.4(4.3)
CDSPP (Ours)	69.1(3.2)	69.0(3.6)	68.8(3.2)	68.8(3.0)	68.9	74.7(3.4)
CDSPP+PCA (Ours)	**71.2(3.2)**	**71.7(3.1)**	**71.4(3.0)**	**72.1(3.0)**	**71.6**	**76.5(3.3)**

Table 3: Mean(std) of classification accuracy (%) over ten trials on the Office-Caltech dataset using SURF (source) and $Decaf_6$ (target) features under supervised (denoted by $*$) and semi-supervised settings (each column represents one Source $\rightarrow$ Target adaptation task).

Method	C→C	C→A	C→D	C→W	A→C	A→A	A→D	A→W	D→C	D→A	D→D	D→W	W→C	W→A	W→D	W→W	Avg
SVM_t *	73.6(4.9)	87.9(2.2)	92.3(3.6)	88.4(3.8)	73.6(4.9)	87.9(2.2)	92.3(3.6)	88.4(3.8)	73.6(4.9)	87.9(2.2)	92.3(3.6)	88.4(3.8)	73.6(4.9)	87.9(2.2)	92.3(3.6)	88.4(3.8)	85.5
HFA [23] *	80.1(2.3)	88.9(1.9)	91.6(3.6)	90.7(3.5)	80.2(2.3)	88.9(1.9)	91.5(3.6)	90.5(3.6)	80.2(2.2)	88.8(1.9)	91.8(3.6)	90.7(3.5)	80.2(2.3)	88.8(1.9)	91.5(3.7)	90.6(3.7)	87.8
CDLS_sup [15] *	76.1(2.1)	86.6(3.2)	91.3(4.7)	87.4(3.5)	75.9(3.5)	87.0(2.8)	90.6(3.8)	86.0(3.6)	51.5(4.4)	74.2(2.4)	86.6(3.2)	77.2(5.1)	74.7(4.1)	85.4(3.0)	90.5(3.8)	86.0(3.5)	81.7
DAMA_sup *	78.7(2.4)	87.3(2.2)	91.5(2.6)	88.6(4.3)	77.4(3.2)	85.9(2.4)	90.7(3.3)	88.2(4.1)	79.6(2.2)	88.8(1.6)	90.1(3.6)	89.4(4.1)	78.5(2.6)	87.4(2.0)	89.1(3.1)	88.6(4.7)	86.2
CDSPP_sup (Ours) *	80.3(2.0)	89.0(1.9)	92.0(3.5)	90.7(3.8)	80.3(2.1)	89.1(1.9)	91.7(3.7)	90.7(3.7)	79.8(2.1)	88.9(1.8)	90.4(3.9)	90.1(3.9)	80.4(2.2)	89.0(1.8)	91.5(4.1)	90.6(3.8)	87.8
DAMA [30]	76.6(2.6)	86.2(1.9)	91.0(2.5)	88.2(4.3)	73.6(4.7)	83.3(2.6)	88.8(3.7)	86.5(4.4)	77.5(2.5)	88.4(1.6)	90.7(4.2)	90.1(3.8)	76.1(2.9)	86.0(2.3)	87.7(4.7)	86.8(5.8)	84.8
SHFA [23]	77.1(2.8)	86.2(3.8)	93.0(3.6)	90.0(2.6)	80.5(3.1)	86.7(2.2)	94.3(2.5)	90.0(4.0)	81.6(2.1)	88.5(2.9)	93.5(3.9)	92.0(4.1)	80.5(1.8)	88.5(2.4)	93.5(3.5)	89.5(4.2)	87.8
CDLS [15]	80.6(1.8)	88.8(2.1)	93.0(3.2)	91.1(3.7)	80.6(1.8)	88.8(2.1)	92.0(3.0)	91.0(4.5)	78.4(2.7)	87.2(2.3)	93.0(3.7)	88.9(5.6)	81.0(2.0)	88.6(2.2)	92.1(3.3)	91.4(4.2)	87.9
PA [19]	87.2(1.1)	90.8(1.3)	92.9(3.3)	93.9(3.9)	87.0(1.1)	90.5(1.7)	94.7(2.5)	94.0(3.9)	87.0(1.3)	90.5(2.0)	**94.5(2.8)**	94.3(3.7)	87.0(1.3)	90.7(1.5)	93.4(4.1)	92.8(4.6)	91.3
TIT [20]	84.9(1.7)	89.9(1.6)	94.6(3.1)	92.2(4.3)	84.6(1.5)	89.7(1.7)	94.6(2.2)	92.3(4.9)	82.7(1.5)	88.7(1.9)	94.3(2.7)	92.1(4.0)	84.7(1.6)	89.5(1.8)	92.5(2.8)	92.5(4.3)	90.0
STN [38]	88.2(1.7)	92.4(0.7)	94.4(2.0)	92.8(4.9)	**88.4(1.6)**	92.5(0.7)	95.0(2.0)	93.9(4.1)	87.9(1.7)	92.2(0.5)	94.4(2.5)	93.3(5.0)	**88.2(1.8)**	92.6(0.8)	93.9(3.2)	92.2(5.1)	92.0
DDACL [39]	86.5(1.6)	91.8(0.9)	94.2(2.8)	93.5(3.4)	86.2(1.9)	83.1(11.2)	89.1(5.9)	92.3(3.9)	86.2(1.7)	91.8(1.1)	93.4(3.6)	93.6(3.0)	86.8(1.7)	92.0(0.8)	94.4(3.2)	94.0(3.1)	90.6
SSAN [22]	80.9(8.7)	89.8(2.8)	**95.8(2.0)**	**94.2(2.1)**	84.9(4.7)	89.0(4.0)	93.1(3.6)	93.1(3.1)	81.0(4.7)	90.3(1.5)	93.9(3.6)	82.6(14.7)	84.3(2.2)	86.9(10.0)	93.5(5.2)	**95.0(2.1)**	89.3
DAMA+	88.1(1.7)	**92.7(0.6)**	93.9(1.7)	92.2(4.1)	88.0(1.3)	**92.9(0.6)**	93.9(2.1)	92.8(4.2)	87.7(1.9)	**93.2(0.5)**	92.1(5.3)	94.0(3.3)	88.1(2.1)	**92.7(0.7)**	94.8(1.6)	93.5(3.9)	91.9
CDSPP (Ours)	**88.3(0.7)**	92.3(0.7)	95.6(1.5)	94.1(4.1)	88.1(1.0)	92.6(0.5)	**95.7(1.0)**	**94.6(3.8)**	**88.1(0.6)**	92.7(0.5)	93.5(4.6)	**94.7(3.5)**	88.1(1.0)	92.5(0.5)	**95.7(1.3)**	94.3(3.8)	**92.6**

to 71.6%, comparable with the best performance 71.7% achieved by PA. This demonstrates the fact handcrafted text features (i.e. bag-of-features) used in the MRC dataset contain noisy vari-

Table 4: Mean(std) of classification accuracy (%) over ten trials on the Office-Home dataset using VGG16 (source) and ResNet50 (target) features under supervised (denoted by $*$) and semi-supervised settings (each column represents one Source $\rightarrow$ Target adaptation task).

Method	A→A	A→C	A→P	A→R	C→A	C→C	C→P	C→R	P→A	P→C	P→P	P→R	R→A	R→C	R→P	R→R	Avg
SVM$_t$*	51.8(1.2)	41.4(1.6)	71.0(1.4)	65.8(2.3)	51.8(1.2)	41.4(1.6)	71.0(1.4)	65.8(2.3)	51.8(1.2)	41.4(1.6)	71.0(1.4)	65.8(2.3)	51.8(1.2)	41.4(1.6)	71.0(1.4)	65.8(2.3)	57.5
CDLS_sup [15] *	58.7(0.9)	45.7(1.5)	75.0(0.8)	69.8(1.9)	53.4(1.0)	48.6(1.0)	73.9(0.9)	67.8(1.8)	55.0(0.9)	45.9(1.4)	78.0(0.8)	70.2(1.5)	56.5(1.1)	46.8(1.5)	76.2(0.5)	72.4(1.4)	62.1
DAMA_sup *	56.6(2.8)	43.6(2.2)	72.0(1.4)	67.8(2.4)	42.7(4.8)	39.8(5.4)	64.8(5.9)	57.5(4.5)	52.4(3.9)	40.4(4.1)	70.1(5.7)	63.6(3.8)	51.8(3.6)	42.4(3.4)	68.8(5.1)	65.5(4.7)	56.2
CDSPP_sup (Ours) *	60.8(1.2)	49.5(1.1)	76.3(0.8)	71.9(1.8)	59.4(1.4)	50.4(1.0)	76.1(0.9)	71.6(1.8)	59.8(1.2)	49.6(1.1)	78.0(0.9)	72.4(1.4)	60.4(1.3)	49.8(0.9)	76.9(1.0)	73.3(1.6)	64.8
DAMA [30]	55.6(3.3)	43.8(2.1)	71.1(2.1)	66.4(3.5)	43.1(4.7)	39.3(5.2)	62.9(5.7)	56.4(4.7)	52.1(4.1)	40.4(4.6)	69.9(4.3)	64.3(5.3)	51.9(3.6)	42.0(4.3)	68.3(5.0)	65.1(4.5)	55.8
CDLS [15]	62.1(0.9)	46.9(1.2)	76.8(0.7)	71.5(2.3)	55.7(1.3)	47.4(1.2)	76.7(0.6)	70.8(2.0)	56.4(1.1)	47.0(1.2)	77.8(0.6)	71.5(2.0)	56.7(1.2)	47.6(1.3)	77.5(0.4)	72.2(2.0)	63.4
PA [19]	59.8(1.2)	48.2(1.5)	80.0(1.2)	75.5(1.8)	59.8(1.1)	48.2(1.3)	80.0(1.3)	75.4(1.9)	59.5(1.5)	48.2(1.4)	80.0(1.6)	75.7(1.9)	59.6(1.3)	48.2(1.5)	79.9(1.4)	75.7(1.8)	65.8
TIT [20]	55.6(1.0)	44.7(1.3)	74.3(1.0)	70.3(1.8)	56.1(0.9)	45.5(1.1)	74.7(0.7)	70.2(1.7)	55.9(1.1)	45.3(1.3)	74.9(0.9)	70.2(1.8)	55.5(1.5)	44.6(1.4)	74.7(0.8)	69.9(2.0)	61.4
STN [38]	62.6(1.4)	51.2(1.5)	78.7(3.9)	74.5(4.3)	56.1(3.8)	52.2(2.2)	77.0(4.0)	71.1(6.0)	60.7(1.3)	49.3(6.0)	**82.4(1.0)**	75.8(2.8)	61.0(1.3)	50.6(3.2)	80.4(0.9)	75.7(4.4)	66.2
DDACL [39]	50.3(2.2)	39.8(2.4)	59.4(2.8)	56.1(3.4)	45.1(2.0)	36.3(3.0)	60.9(2.9)	56.8(2.0)	40.3(1.5)	34.2(2.3)	55.7(9.1)	43.0(9.9)	41.9(2.4)	36.5(2.0)	52.4(5.1)	51.5(9.2)	47.5
SSAN [22]	50.5(1.9)	40.1(3.0)	70.9(1.8)	63.9(3.0)	43.9(2.9)	42.5(5.0)	67.8(1.2)	61.9(2.9)	44.1(2.6)	38.1(3.5)	77.3(0.9)	66.2(1.3)	45.7(3.9)	38.6(3.8)	71.7(4.0)	68.8(2.5)	55.8
DAMA+	62.1(2.4)	49.0(1.4)	77.7(1.9)	75.0(2.5)	54.0(5.2)	44.7(6.1)	75.6(3.7)	69.0(3.4)	60.9(2.7)	46.9(3.1)	76.9(3.5)	72.5(1.9)	60.3(1.9)	48.6(3.7)	76.7(2.8)	73.4(3.3)	63.9
CDSPP (Ours)	**65.7(1.0)**	**54.8(2.0)**	**81.0(1.5)**	**78.4(1.1)**	**65.0(1.4)**	**55.1(1.6)**	**80.9(1.6)**	**78.5(1.2)**	**65.6(0.4)**	**54.7(1.9)**	81.5(1.1)	**78.8(1.0)**	**65.5(0.9)**	**54.6(1.6)**	**80.9(1.6)**	**79.4(0.9)**	**70.0**

Table 5: Mean(std) of classification accuracy (%) over ten trials on the Office-Home dataset using ResNet50 (source) and VGG16 (target) features under supervised (denoted by $*$) and semi-supervised settings (each column represents one Source $\rightarrow$ Target adaptation task).

Method	A→A	A→C	A→P	A→R	C→A	C→C	C→P	C→R	P→A	P→C	P→P	P→R	R→A	R→C	R→P	R→R	Avg
SVM$_t$ *	40.3(1.4)	30.5(1.6)	63.3(1.7)	56.3(2.9)	40.3(1.4)	30.5(1.6)	63.3(1.7)	56.3(2.9)	40.3(1.4)	30.5(1.6)	63.3(1.7)	56.3(2.9)	40.3(1.4)	30.5(1.6)	63.3(1.7)	56.3(2.9)	47.6
CDLS_sup [15] *	51.4(1.1)	36.5(1.0)	69.6(1.1)	63.5(2.0)	46.4(1.2)	39.2(1.0)	68.7(1.2)	62.0(1.6)	47.2(1.2)	36.4(0.8)	73.1(1.0)	64.6(1.9)	48.6(1.1)	37.1(1.1)	70.9(1.2)	66.4(2.0)	55.1
DAMA_sup *	46.9(1.8)	35.6(1.8)	65.9(1.4)	60.3(1.8)	43.4(2.4)	32.5(3.7)	60.3(6.0)	56.3(3.0)	44.1(4.0)	31.8(3.6)	62.2(4.0)	56.4(4.0)	45.3(3.2)	34.4(1.6)	60.9(4.6)	60.3(2.1)	49.8
CDSPP (Ours)*	49.7(1.1)	39.2(1.0)	69.5(1.3)	63.7(2.0)	48.3(1.2)	40.4(1.3)	69.5(1.5)	63.4(1.8)	48.5(1.1)	38.9(0.8)	71.3(1.4)	64.1(1.9)	49.0(1.2)	39.4(1.1)	70.1(1.3)	65.0(2.1)	55.6
DAMA [30]	46.7(2.0)	33.6(2.5)	66.2(1.7)	57.8(3.4)	43.1(4.0)	32.0(4.5)	60.2(6.2)	55.7(5.0)	44.3(3.7)	32.0(4.1)	65.5(5.6)	59.8(3.5)	45.3(3.4)	34.8(2.6)	65.0(4.4)	60.9(3.5)	50.2
CDLS [15]	54.9(1.1)	36.6(1.1)	71.1(0.8)	65.9(1.3)	47.8(1.4)	39.8(1.2)	69.5(1.2)	63.6(1.4)	49.7(1.2)	36.8(1.2)	75.6(0.8)	67.9(1.6)	52.3(1.0)	38.5(1.3)	73.1(1.0)	69.6(1.6)	57.0
PA [19]	51.4(1.0)	38.3(1.3)	73.7(1.2)	67.4(1.6)	51.2(1.4)	38.2(1.2)	73.6(1.2)	67.4(1.6)	51.2(1.1)	38.1(1.4)	73.6(1.2)	67.3(1.9)	51.2(0.9)	38.2(1.2)	73.7(1.2)	67.4(1.4)	57.6
TIT [20]	46.8(1.7)	36.4(1.2)	69.4(0.9)	62.5(1.8)	47.0(1.7)	36.4(1.1)	69.3(1.1)	62.0(2.2)	46.8(1.7)	36.4(1.1)	69.8(0.9)	62.4(2.1)	45.9(1.6)	36.0(1.3)	69.4(1.2)	62.5(2.1)	53.7
STN [38]	52.6(1.5)	41.2(2.4)	74.9(1.0)	69.2(1.5)	51.2(1.1)	42.5(1.2)	75.3(1.2)	69.6(1.0)	53.0(1.2)	41.7(1.4)	**77.3(1.2)**	70.7(1.4)	52.7(1.9)	41.7(1.4)	**76.6(1.0)**	71.6(1.3)	60.1
DDACL [39]	33.8(2.3)	27.5(1.6)	52.2(4.0)	46.8(1.6)	31.8(2.3)	24.3(1.6)	50.8(1.9)	44.0(3.5)	32.0(2.7)	23.4(2.8)	49.0(7.9)	39.9(7.3)	32.4(2.7)	24.9(1.6)	46.5(3.7)	45.5(4.4)	37.8
SSAN[22]	42.2(4.1)	30.4(2.3)	61.9(3.7)	56.5(2.6)	37.9(1.6)	32.3(2.3)	62.1(1.5)	53.4(3.4)	38.1(2.0)	29.9(1.7)	69.0(2.9)	58.0(1.9)	37.5(2.3)	29.6(1.7)	63.3(2.2)	57.9(3.5)	47.5
DAMA+	49.1(2.9)	37.5(1.2)	71.1(1.5)	65.4(2.5)	49.7(1.9)	32.9(4.1)	68.3(3.3)	63.2(3.7)	48.9(3.1)	33.3(3.6)	68.1(2.5)	61.4(3.9)	49.9(3.1)	36.3(2.2)	67.1(2.4)	64.6(2.1)	54.2
CDSPP (Ours)	**55.6(1.1)**	**44.7(1.8)**	**75.2(1.6)**	**71.7(1.4)**	**54.5(1.2)**	**46.0(1.6)**	**75.7(1.6)**	**71.4(1.9)**	**54.7(1.2)**	**45.0(1.6)**	76.0(1.8)	**71.8(1.6)**	**55.0(1.3)**	**44.9(2.0)**	75.8(1.8)	**72.1(1.8)**	**61.9**

ables which cannot be well handled by the CDSPP algorithm itself but a pre-processing like PCA suffices to address this issue.

Table 2 (rightmost column) also presents the results of tag-to-image adaptation on the NUS-ImageNet dataset. There is only one adaptation task (i.e. Tag→Image) in this dataset. In the supervised HDA setting, the baseline method SVM_t is outperformed by all three comparative methods with large margins among which HFA achieves the best performance of 67.5% as opposed to the accuracy of 67.2% by our proposed CDSPP_sup. However, HFA is more computationally expensive than others as discussed in [23]. In the semi-supervised HDA setting, our method achieves the best performance with an accuracy of 74.7%. The performance of our CDSPP can be further improved to 76.5% when PCA is applied to reduce the dimensionality of the text features to 50.

Similar results can also be observed in Table 3 for the image classification experiments on Office-Caltech. Both HFA and our CDSPP achieve the same average accuracy of 87.8% in the supervised HDA setting. CDLS_sup performs worse than the baseline method SVM_t again due to the restricted PCA dimensions as discussed above. In the semi-supervised HDA, our CDSPP achieves the best results in 6 out of 16 adaptation tasks and has the highest average accuracy of 92.6%.

The experimental results for the challenging Office-Home dataset are shown in Table 4 and Table 5. The difference between these two tables lies in the features used for the source/target domains are VGG16/ResNet50 and ResNet50/VGG16 respectively. In this experiment, the methods HFA and SHFA are excluded due to their extremely long computation time given the scale of this dataset. It can be seen that CDLS_sup, for the first time, outperforms the baseline method SVM_t on this dataset since the total number of labelled target samples is 195 which no longer restricts the PCA dimension in this algorithm. Two more recent approaches DDACL [39] and SSAN [22], however, perform poorly on this more challenging dataset although they achieve good performance on three simpler datasets. One reasonable explanation is that these two approaches along with many others benefit from the clustering characteristics of the original features and can easily recognize the target samples cluster-wisely. For the more challenging dataset, the classes are prone to overlap in a low-dimensional subspace if the projections are not properly learned. The simultaneous learning of the classifier and feature projections tends to result in an overfitted classifier to the labelled and pseudo-labelled samples and the overfitting can be an

issue when the labelled target samples cannot represent the distribution of their corresponding classes in the subspace. As a result, they suffer from negative adaptation when the pseudo-labels are inaccurate at the beginning and less robustness to the choice of labelled target samples. This also provides evidence for the necessity of new test beds for HDA approaches. In both tables, the best performances were achieved by our CDSPP for most adaptation tasks in both supervised and semi-supervised settings. Specifically, CDSPP achieves an average accuracy of 70.0% when VGG16 and ResNet50 features were employed for source and target domains, significantly better than the second-best performance 66.2% achieved by STN [38]. Similar results can be observed in Table 5, CDSPP achieves the best performance of 61.9% as opposed to the second-best 60.1% by STN [38]. The significant performance improvement gained by CDSPP on the Office-Home dataset is attributed to the fact this dataset is much more challenging than other datasets since it contains much more classes (65 vs 10, 8, 6). We believe Office-Home is a more appropriate testbed for discriminating different HDA methods.

In addition, the performance comparison between DAMA and DAMA+ provide further evidence that the use of the iterative learning framework described in Section 3.4 is beneficial to semi-supervised HDA. On the other hand, the superior performance of CDSPP to DAMA+ across all datasets validates the fact that our CDSPP is essentially different from DAMA as discussed in Section 3.3. In the supervised HDA experiments, CDSPP also outperforms our adaptation of DAMA consistently on four datasets and the performance gap on the challenging Office-Home dataset is particularly significant. The other interesting phenomenon that can be observed from Tables 3-4 is the semi-supervised DAMA (i.e. the original version in [30]) performs no better than its supervised version (i.e. DAMA_sup adapted by ourselves). This demonstrates that the way how DAMA [30] exploits the unlabelled target-domain data is ineffective. By contrast, the selective pseudo-labelling strategy employed in our proposed CDSPP is more effective and can be readily used by other HDA algorithms.

4.4. On the Number of labelled Target Samples

We conducted additional experiments of semi-supervised HDA to compare our proposed CDSLPP with other methods when different numbers of labelled target samples were used for training. Specifically, we set the number of labelled target samples as 5, 10, 15 or 20 for the MRC dataset whilst for the other three datasets the investigated numbers of labelled target samples were within the collection of $\{1,3,5,7,9\}$. For the MRC and NUS-ImageNet datasets, all

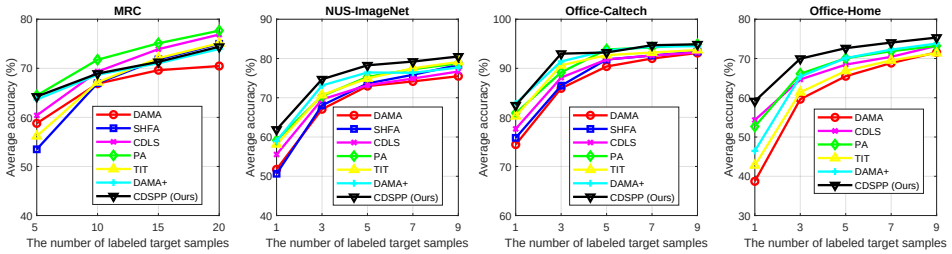

Figure 2: Comparison results when different numbers of labelled target samples are used.

adaptation tasks (i.e. $EN/FR/GE/IT \rightarrow SP$ and $Tag \rightarrow Image$, respectively) were repeated for ten trials with randomly selected data (the same as those used in the previous experiment). To save computational time without loss of generality, we only conducted the first four adaptation tasks for the first three trials for the Office-Caltech ($C \rightarrow C, C \rightarrow A, C \rightarrow D, C \rightarrow W$) and Office-Home ($A \rightarrow A, A \rightarrow C, A \rightarrow P, A \rightarrow R$ with VGG16 and ResNet50 as the source and target features, respectively) datasets in this experiment. For each dataset, the average classification accuracy over all the conducted adaptation tasks in this dataset is reported for comparison.

The experimental results are shown in Figure 2 from which we can draw some conclusions. (1) The performance of all methods is improved with the increase of labelled target samples since more labelled target samples provide additional information for the training. (2) The performance margins between different methods decrease when more labelled target samples are used for training. This phenomenon demonstrates these methods have different capabilities of cross-domain knowledge transfer which is of vital importance when there are limited labelled data in the target domain. (3) Our proposed CDSPP algorithm outperforms the others in three out of four datasets regardless of the number of labelled target samples. The superiority of CDSPP to other methods is more significant when less labelled target samples are available. (4) On the MRC dataset, our method performs the best when 5 labelled target samples are used but outperformed by CDLS [15] and [19] when more labelled target samples are available.

4.5. On the Effect of Hyper-parameters

In all our experiments described above, we empirically set the dimensionality of the common subspace d equal to the number of classes in the dataset and set the hyper-parameters $\alpha = 10$ (c.f. Eq.(8)) and the number of iterations $T = 5$ (c.f. Algorithm 2). In this experiment, we will show

how these values were selected and the fact that our algorithm is not sensitive to these hyper-parameters across all the datasets. Similar to the experimental settings in the previous section, we repeated all the adaptation tasks for ten trials for the MRC and NUS-ImageNet datasets and repeated the first four adaptation tasks for the first three trials for the Office-Caltech and Office-Home datasets to save time without loss of generality. The average accuracy over all the investigated adaptation tasks is reported for each dataset when a specific hyper-parameter value is used.

Firstly, we investigate the effect of the subspace dimension d. The values of d were from the set $\{2,4,6,8,10,16,32,64/65,128,256,512\}$ which contains the class numbers of four datasets (i.e. 6, 8, 10 and 65) as well as other candidate values less or greater than the class numbers. The experimental results are shown in the left graph of Figure 3. It is not hard to see that the best performance can be achieved when the value of d is no less than the number of classes in each dataset. A greater value of d does not further improve the performance but a smaller value of d leads to a significant performance drop. As a result, it is easy to select an optimal value of the subspace dimension for our proposed CDSPP.

Subsequently, We investigate the effect of the regularization parameter α in Eq.(8) by conducting experiments with the values of α selected from $\{0.01,0.1,1,10,100,1000\}$. The experimental results are shown in the middle graph of Figure 3 from which we can see that the optimal values of α should be between 10 and 100 across all datasets. A smaller value of α leads to performance drops for all datasets except Office-Caltech. This validates the necessity of the regularization term in Eq.(8) in our method and it is not very sensitive to the value of α. Similar findings have been validated in the traditional LPP algorithm by Wang and Chen [34].

Finally, we are concerned about the number of iterations T by setting $T = \{1,3,5,7,9,11,15,21\}$. The right-side graph in Figure 3 shows that the CDSPP algorithm performs generally well when $T \geq 5$. Increasing the number of iterations further can only improve the performance on the NUS-ImageNet dataset very marginally but will increase the computational cost significantly. As a result, we selected $T = 5$ as the optimal value in all our experiments.

4.6. Qualitative Evaluation

To give an intuitive explanation of how our algorithm can align two heterogeneous domains progressively, we take the tag-to-image adaptation task in the NUS-ImageNet dataset as an example and visualise the distribution of samples in the learned subspace. As shown in Figure 4(a),

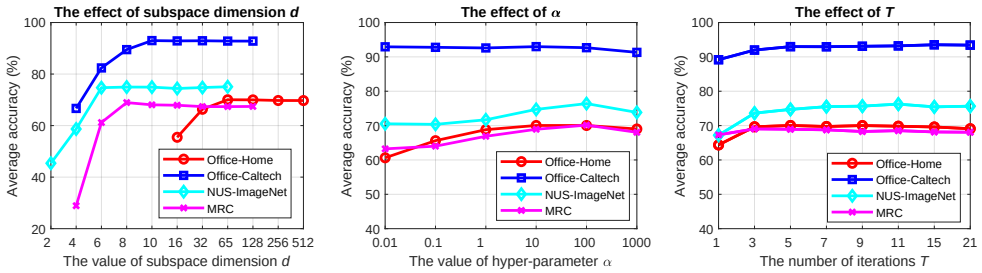

Figure 3: Performance sensitivity to hyper-parameters.

the original features from the two domains are independent of each other although the clustering characteristics are evident. Figure 4(b) illustrates how the three labelled target samples ("circles") are pulled closer to the corresponding source classes ("squares") after the first iteration of CDSPP. More importantly, due to the property of structure preservation of CDSPP, the unlabelled target samples ("crosses") are also moving towards their corresponding source clusters. In Figure 4(c), we can see more target samples are pseudo-labelled ("crosses" within "circles") and the source and target domains are further aligned. Such progressive pseudo-labelling and domain alignment are enhanced in Figure 4(d) and no significant improvement can be observed in the following iterations (e) and (f). This is consistent with the recognition results achieved by our CDSPP in this particular experiment (i.e. from the first to the fifth iteration, recognition accuracy is 70.1%, 76.7%, 79.1%, 78.9% and 79.0%, respectively).

It is obvious that the clustering of eight classes has converged after the third iteration and the two domains are relatively well aligned. The samples which are misclassified in the final iteration are those located in the overlapping regions of two classes. The overlap comes from the original features as shown in Figure 4(a) and can be mitigated in different ways. The best way is to extract more discriminative features to avoid such distribution overlap from the beginning which, however, is beyond our focus of this paper. Alternatively, one can use a more capable domain adaptation algorithm such as our proposed CDSPP to mitigate the class overlap by learning the most discriminative features from the original ones. In addition, the choice of labelled target samples also makes a difference. Taking a closer look at Figure 4(a), we can see one of the three randomly selected labelled target samples for class 5 is far away from the target cluster of class 5. When this outlier is pulled closer to the source cluster of class 5, some samples from class 2 and

class 6 are also mistakenly pulled close to the source cluster of class 5 as shown in Figure 4(b). These observations also imply it is important to choose the most representative target samples to label for improved performance in practice.

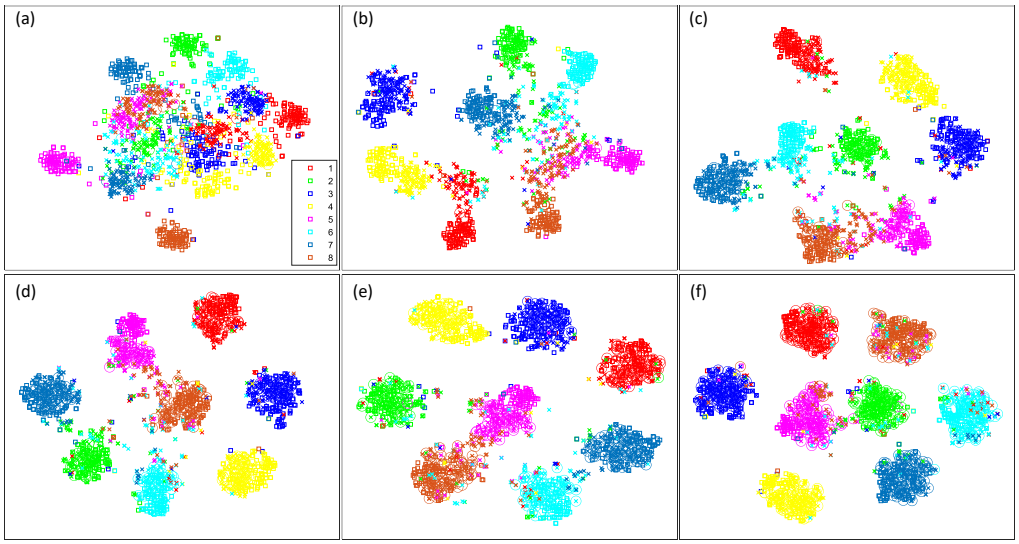

Figure 4: Visualisation of the learned subspace for the NUS-ImageNet dataset (i.e. the tag to image adaptation task) using the proposed CDSPP, best view in colour. (Results are from one of the ten trials with a specific random seed; eight classes 1-8 are represented by different colours; "squares": labelled source samples; "crosses": unlabelled target samples; "circles": labelled or pseudo-labelled target samples; (a) the original features learned by two separate PCA projections independently; (b)-(f) projections in the subspace learned by CDSPP after 1st-5th iteration.)

4.7. On the Computational Efficiency

We compare the computational efficiency of different methods by calculating the time cost of each method in the experiments. The experiments are conducted on a laptop with an Intel Core i5-7300HQ CPU @ 2.5 GHz and 32 GB memory. For neural network based methods STN and SSAN, the Nvidia Titan Xp GPUs are used. The results are shown in Table 6. The computational time is calculated by averaging the time for all adaptation tasks (i.e. 4, 1, 16 and 16 tasks for MRC, NUS-ImageNet, Office-Caltech and Office-Home respectively) over three trials. By comparison, our proposed CDSPP is generally the most efficient method on three out of four datasets. The exception on Office-Caltech is because CDLS and TIT use dimensionality reduction such as PCA to reduce the dimensionality of Decaf features from 4096 to a much lower value whilst our CDSPP uses the original 4096-dimensional features. From Table 6 we can also

Table 6: Computation time (s) of different methods on four datasets (the total time of all adaptation tasks in each dataset is calculated).

Method	MRC	NUS-ImageNet	Office-Caltech	Office-Home
DAMA [30]	46	7	58	477
SHFA [23]	917	25	255	Inf
CDLS [15]	168	**6**	**47**	272
PA [19]	617	30	121	3991
TIT [20]	175	11	52	1740
STN [38]	2734	343	7134	40857
DDACL [39]	622	169	2940	3421
SSAN [22]	9520	1229	13245	47145
DAMA +	49	21	288	1390
CDSPP (Ours)	**16**	7	161	**256**

see different methods have the varying capability of scaling to larger datasets (e.g., from NUS-ImageNet to Office-Home) in terms of both feature dimensionality and the number of samples. In particular, SHFA takes an excessively long time before completing one single adaptation task of Office-Home in our experiment hence is marked as Inf in the table. STN and SSAN take the most time across all datasets since neural networks are trained for a large number of iterations which is generally much less efficient compared with our CDSPP which can be solved by eigen-decomposition.

5. Conclusion and Future Work

We propose a novel algorithm CDSPP for HDA and extend it to the semi-supervised setting by incorporating it into an iterative learning framework. Experimental results on several benchmark datasets demonstrate the proposed CDSPP is not only computationally efficient but also can achieve state-of-the-art performance on four datasets. We also investigate the effect of the number of labelled target samples in the performance of different methods and found that the use of too many labelled target samples will suppress the performance distinction among different methods. The newly introduced benchmark dataset Office-Home for HDA is proved a proper

testbed for HDA since it is more challenging with much more classes than others and the performances of investigated methods on this dataset are more significantly varied. In addition, the proposed method for HDA is not sensitive to hyper-parameters and it is easy to select optimal hyper-parameter values across varying datasets.

One limitation of the proposed method is that its performance relies on the quality of pre-extracted features. As we have observed in our experiments on the MRC dataset, proper pre-processing of features can affect the domain adaptation performance significantly. One direction of future work to address this issue is to unify the feature extraction neural networks and domain adaptation. For HDA, the source and target domains are different either in the data modality (e.g., text and image) or in the feature space. As a result, two individual neural networks are needed for feature extraction before feeding the features into the domain adaptation module. Our selective pseudo-labelling strategy described in this paper can also be easily applied to exploit the unlabelled target-domain data when training the unified neural networks for HDA.

References

[1] Amini, M., Usunier, N., and Goutte, C. (2009). Learning from multiple partially observed views-an application to multilingual text categorization. In *NeurIPS*, pages 28–36.

[2] Atapour-Abarghouei, A. and Breckon, T. P. (2018). Real-time monocular depth estimation using synthetic data with domain adaptation via image style transfer. In *CVPR*, pages 2800–2810.

[3] Chen, C., Chen, Z., Jiang, B., and Jin, X. (2019a). Joint domain alignment and discriminative feature learning for unsupervised deep domain adaptation. In *AAAI*.

[4] Chen, C., Xie, W., Huang, W., Rong, Y., Ding, X., Huang, Y., Xu, T., and Huang, J. (2019b). Progressive feature alignment for unsupervised domain adaptation. In *CVPR*, pages 627–636.

[5] Chen, W.-Y., Hsu, T.-M. H., Tsai, Y.-H. H., Wang, Y.-C. F., and Chen, M.-S. (2016). Transfer neural trees for heterogeneous domain adaptation. In *ECCV*, pages 399–414. Springer.

[6] Chua, T.-S., Tang, J., Hong, R., Li, H., Luo, Z., and Zheng, Y. (2009). Nus-wide: a real-world web image database from national university of singapore. In *ACM international conference on image and video retrieval*, page 48.

[7] Collobert, R., Sinz, F., Weston, J., and Bottou, L. (2006). Large scale transductive svms. *Journal of Machine Learning Research*, 7(Aug):1687–1712.

[8] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255.

[9] Deng, W.-Y., Dong, Y.-Y., Liu, G.-D., Wang, Y., and Men, J. (2019). Multiclass heterogeneous domain adaptation via bidirectional ecoc projection. *Neural Networks*, 119:313–322.

[10] Fang, W.-C. and Chiang, Y.-T. (2018). A discriminative feature mapping approach to heterogeneous domain adaptation. *Pattern Recognition Letters*, 106:13–19.

[11] Gong, B., Shi, Y., Sha, F., and Grauman, K. (2012). Geodesic flow kernel for unsupervised domain adaptation. In *CVPR*, pages 2066–2073. IEEE.

[12] He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *CVPR*, pages 770–778.

[13] He, X. and Niyogi, P. (2004). Locality preserving projections. In *NeurIPS*, pages 153–160.

[14] Huang, J., Zhou, Z., Shang, J., and Niu, C. (2020). Heterogeneous domain adaptation with label and structural consistency. *Multimedia Tools and Applications*, pages 1–21.

[15] Hubert Tsai, Y.-H., Yeh, Y.-R., and Frank Wang, Y.-C. (2016). Learning cross-domain landmarks for heterogeneous domain adaptation. In *CVPR*, pages 5081–5090.

[16] Li, H., Pan, S. J., Wan, R., and Kot, A. C. (2019a). Heterogeneous transfer learning via deep matrix completion with adversarial kernel embedding. In *AAAI*.

[17] Li, J., Chen, E., Ding, Z., Zhu, L., Lu, K., and Shen, H. T. (2020a). Maximum density divergence for domain adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

[18] Li, J., Jing, M., Lu, K., Zhu, L., and Shen, H. T. (2019b). Locality preserving joint transfer for domain adaptation. *IEEE Transactions on Image Processing*, 28(12):6103–6115.

[19] Li, J., Lu, K., Huang, Z., Zhu, L., and Shen, H. T. (2018a). Heterogeneous domain adaptation through progressive alignment. *IEEE transactions on neural networks and learning systems*, 30(5):1381–1391.

[20] Li, J., Lu, K., Huang, Z., Zhu, L., and Shen, H. T. (2018b). Transfer independently together: a generalized framework for domain adaptation. *IEEE transactions on Cybernetics*, 49(6):2144–2155.

[21] Li, J., Lu, K., Zhu, L., and Li, Z. (2017). Locality-constrained transfer coding for heterogeneous domain adaptation. In *Australasian database conference*, pages 193–204. Springer.

[22] Li, S., Xie, B., Wu, J., Zhao, Y., Liu, C. H., and Ding, Z. (2020b). Simultaneous semantic alignment network for heterogeneous domain adaptation. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 3866–3874.

[23] Li, W., Duan, L., Xu, D., and Tsang, I. W. (2013). Learning with augmented features for supervised and semi-supervised heterogeneous domain adaptation. *IEEE transactions on PAMI*, 36(6):1134–1148.

[24] Mozafari, A. S. and Jamzad, M. (2016). A svm-based model-transferring method for heterogeneous domain adaptation. *Pattern Recognition*, 56:142–158.

[25] Patel, V. M., Gopalan, R., Li, R., and Chellappa, R. (2015). Visual domain adaptation: A survey of recent advances. *IEEE signal processing magazine*, 32(3):53–69.

[26] Petersen, K. B., Pedersen, M. S., et al. (2008). The matrix cookbook. *Technical University of Denmark*, 7(15):510.

[27] Shen, C. and Guo, Y. (2018). Unsupervised heterogeneous domain adaptation with sparse feature transformation. In *ACML*, pages 375–390.

[28] Simonyan, K. and Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In *ICLR*.

[29] Venkateswara, H., Eusebio, J., Chakraborty, S., and Panchanathan, S. (2017). Deep hashing network for unsupervised domain adaptation. In *CVPR*, pages 5018–5027.

[30] Wang, C. and Mahadevan, S. (2011). Heterogeneous domain adaptation using manifold alignment. In *IJCAI*.

[31] Wang, J., Feng, W., Chen, Y., Yu, H., Huang, M., and Yu, P. S. (2018). Visual domain adaptation with manifold

embedded distribution alignment. In *ACMMM*, pages 402–410.

[32] Wang, Q. and Breckon, T. P. (2020). Unsupervised domain adaptation via structured prediction based selective pseudo-labeling. In *AAAI*.

[33] Wang, Q., Bu, P., and Breckon, T. P. (2019). Unifying unsupervised domain adaptation and zero-shot visual recognition. In *IJCNN*.

[34] Wang, Q. and Chen, K. (2017). Zero-shot visual recognition via bidirectional latent embedding. *International Journal of Computer Vision*, 124(3):356–383.

[35] Weiss, K., Khoshgoftaar, T. M., and Wang, D. (2016). A survey of transfer learning. *Journal of Big Data*, 3(1):9.

[36] Wu, H., Zhu, H., Yan, Y., Wu, J., Zhang, Y., and Ng, M. K. (2021). Heterogeneous domain adaptation by information capturing and distribution matching. *IEEE Transactions on Image Processing*, 30:6364–6376.

[37] Yan, Y., Li, W., Ng, M. K., Tan, M., Wu, H., Min, H., and Wu, Q. (2017). Learning discriminative correlation subspace for heterogeneous domain adaptation. In *IJCAI*, pages 3252–3258.

[38] Yao, Y., Zhang, Y., Li, X., and Ye, Y. (2019). Heterogeneous domain adaptation via soft transfer network. In *ACMMM*, pages 1578–1586.

[39] Yao, Y., Zhang, Y., Li, X., and Ye, Y. (2020). Discriminative distribution alignment: A unified framework for heterogeneous domain adaptation. *Pattern Recognition*, 101:107165.

[40] Zhang, Y., Tang, H., Jia, K., and Tan, M. (2019). Domain-symmetric networks for adversarial domain adaptation. In *CVPR*, pages 5031–5040.

[41] Zhao, S., Li, B., Yue, X., Gu, Y., Xu, P., Hu, R., Chai, H., and Keutzer, K. (2019). Multi-source domain adaptation for semantic segmentation. In *NeurIPS*, pages 7285–7298.

[42] Zhou, H. and Chen, K. (2019). Transferable positive/negative speech emotion recognition via class-wise adversarial domain adaptation. In *ICASSP*, pages 3732–3736.

[43] Zhou, J. T., Pan, S. J., and Tsang, I. W. (2019a). A deep learning framework for hybrid heterogeneous transfer learning. *Artificial Intelligence*.

[44] Zhou, J. T., Tsang, I. W., Pan, S. J., and Tan, M. (2019b). Multi-class heterogeneous domain adaptation. *Journal of Machine Learning Research*, 20(57):1–31.