Neural Processing Letters
https://doi.org/10.1007/s11063-022-10828-3

Partial Domain Adaptation by Progressive Sample Learning of Shared Classes

Lei Tian[1,2] · **Yongqiang Tang**[1] · **Wensheng Zhang**[1,2]

Accepted: 4 April 2022

Abstract

Traditional domain adaptation (DA) research generally assume that the source and target domains have the same label set. However, in many real-world applications, there exists a more general and practical situation where target label set is just a subset of source label set, which is formulated as partial domain adaptation (PDA) problem. Compared with DA, PDA is more vulnerable to negative transfer due to the mismatch of label sets. In this paper, we propose a novel PDA method based on Progressive sample Learning of Shared Classes (PLSC), which contains two main parts: shared classes identification and progressive target sample learning. The shared classes identification component aims to exclude source-private classes and merely allow source samples within shared classes to participate in the progress of knowledge transfer. To achieve this goal, following the separation and alignment assumptions in DA, we minimize the sum of the distances from both source and target samples to their corresponding source class centers, and then design an adaptive threshold to determine the shared classes. Furthermore, considering the misleading of target samples that deviate from the source class centers, we propose to progressively include target samples for subspace learning by introducing self-paced learning mechanism. Extensive experiments verify the superiority of our method against the existing counterparts.

Keywords Partial domain adaptation · Domain adaptation · Transfer learning · Self-paced learning · Low-dimensional subspace learning

✉ Yongqiang Tang
yongqiang.tang@ia.ac.cn

Lei Tian
tianlei2017@ia.ac.cn

Wensheng Zhang
zhangwenshengia@hotmail.com

[1] Institute of Automation, Chinese Academy of Sciences, Beijing, China

[2] School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China

Published online: 25 April 2022

1 Introduction

In the machine learning community, one general assumption is that the training data and the test data follow an identical feature distribution. However, this assumption may be violated in many usual situations. Besides, it is time consuming and expensive to collect and annotate massive training data. Thus, there is a strong motivation to transfer the knowledge from a well-annotated source domain to an unlabeled target domain that has a different feature distribution. To this end, considerable efforts have been devoted to domain adaptation (DA) [1]. The goal of DA is to minimize the distribution discrepancy between two domains [2], such that the classifier trained on source domain can be directly applied to the target domain. So far, DA has been applied to various tasks, such as object recognition [3, 4], face recognition [3, 5] and person re-identification [6, 7].

Although DA has been applied in various tasks successfully, it often assumes that the source domain and the target domain share the same label set. However, in practical applications, this assumption is hard to hold as the target label set is unknown, and it is difficult and burdensome to find a source domain with the identical label set as the given target domain. To surmount this issue, partial domain adaptation (PDA) [8] is naturally introduced, which assumes that the target label set is a subset of the source label set. The difference between DA and PDA is shown in Fig. 1. Compared with DA, PDA is more general and practical since PDA can be applied to many problems when a large-scale dataset (e.g., ImageNet [9] and MS COCO [10]) is utilized to form the source domain. In this paper, we call the classes existing in both domains as *shared classes* [11], and the classes only appearing in source domain as *source-private classes* [12].

Due to the mismatch of label sets in PDA, directly aligning the feature distributions between source and target domains would result in serious negative transfer [8]. To remedy this issue, in recent years, several PDA methods have been proposed, which can be roughly divided into three categories. The first category aims to increase the importance of shared classes between two domains and in the meantime reduce the importance of source-private classes. For example, Li et al. [13] use a weighted class-wise alignment loss to learn the different significance of source classes automatically based on the target output probability distribution. Different from this strategy, the second class of methods develop weighting mechanism from sample level. For instance, Cao et al. [14] utilize the decision scores of a domain classifier to develop a weighting scheme to quantify the transferability of each source sample. The third class borrows advantages of the aforementioned two kinds of methods, and combines them to tackle the PDA problem. For example, Kim et al. [12] employ an adaptive graph adversarial network to integrate class-level feature propagation and sample-level transferability.

Despite the impressive performance achieved by these methods, the samples from source-private classes still take part into the process of domain adaptation, which may bring about adverse effects on knowledge transfer of two domains [8]. To handle the negative transfer issue caused by source-private classes, it is supposed to identify the shared classes as accurately as possible and only allow source samples within shared classes to make contribution to the DA process. To achieve goal, in this paper, we present a simple but effective way to identify the shared classes. Our proposal is inspired by the *separation assumption* and *alignment assumption* in DA [15]. The separation assumption supposes that the source data or the target data are discriminatively clustered in a suitable feature space, while the alignment assumption argues that in the suitable feature space, the clusters corresponding to the identical class in two domains are geometrically close.

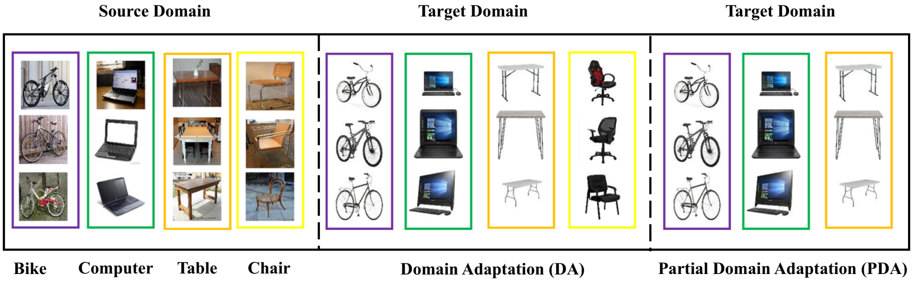


Fig. 1 Illustration of standard Domain Adaptation (DA) and Partial Domain Adaptation (PDA). In DA, the label sets of two domains are the same, while in PDA, the target label set is a subset of the source label set

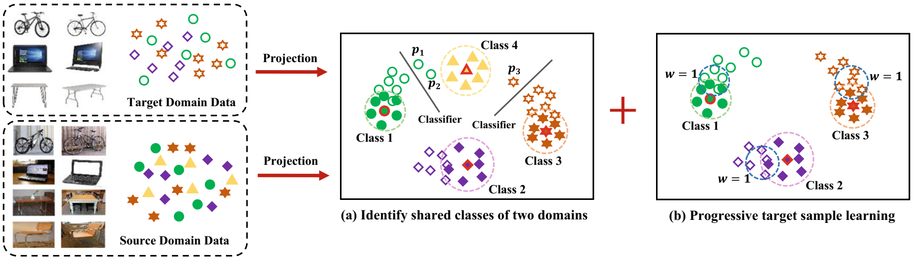


Fig. 2 Illustration of our motivation. Our proposal contains two main parts: **a** shared classes identification and **b** progressive target sample learning

Following the separation assumption, we aim to seek a low-dimensional subspace where the sum of the distance from each source sample to its class center is minimized, such that the source samples of different classes can be well-separated. Meanwhile, according to the alignment assumption, in the subspace, we minimize the sum of the distance from each target sample to its corresponding source class center, so that the target clusters can be aligned with its corresponding source classes more closely. Ideally, in the projection subspace, each target sample can be assigned an accurate class label. However, in real-world applications, as shown in Fig. 2a, the target data generally contain noises, which may make some target samples (see p_1, p_2 and p_3) be away from the source class centers. In such case, these samples are misallocated a label of source-private classes, which hinders the correctness of shared classes identification. To solve this issue, we design an adaptive threshold (> 0) to recognize the shared classes, i.e., one class is considered as shared class only when the number of samples in it is larger than the threshold.

On the basis of excluding source-private classes, from Fig. 2b, we can observe that in the shared classes, the distance from each target sample to its corresponding source class center is significantly different and in the unsupervised setting of PDA the samples deviated from source class centers are unable to provide convincing guidance for projection matrix learning. As a result, how to alleviate the misleading of outlying target samples in shared classes is another problem we concern. To tackle this issue, we introduce the self-paced learning mechanism into our proposal. Specifically, we progressively involve target samples into the process of subspace learning from easy to hard. The difficulty of the samples is determined by the Euclidean distances from them to their corresponding source class centers. By such self-paced strategy, we can obtain a projection matrix with higher quality.

Based on the above introduction, in this paper, a novel PDA method based on Progressive sample Learning of Shared Classes (PLSC) is proposed. The main contributions of this paper are summarized as follows:

1. We propose a novel method to identify shared classes for PDA problem. Inheriting the assumptions of separation and alignment in DA, in the projection subspace, we jointly minimize the sum of the distances from both source and target samples to their corresponding source class centers, and naturally design an adaptive threshold to determine the shared classes.
2. To relieve the misleading of deviated target samples in shared classes, we further introduce the self-paced learning mechanism, which enables our PLSC to gradually add target samples into the process of subspace learning from easy to hard. In this way, a projection matrix with higher quality can be achieved.
3. Extensive experiments on Office31, Office-Home, ImageCLEF and Visda2017 datasets are conducted, and the experimental results validate the superiority and effectiveness of our method.

The rest of this paper is organized as follows. In Sect. 2, we review some works about PDA and self-paced learning. Section 3 elaborates our PLSC, the optimization procedure and complexity analysis. We conduct extensive PDA experiments in Sect. 4. Finally, Sect. 5 concludes this paper.

2 Related Works

2.1 Partial Domain Adaptation

Domain adaptation (DA) aims to transfer the knowledge from a well-labeled source domain to an unlabeled target domain [1], which follows a different distribution. Existing DA approaches can be grouped as three categories: instance reweighting [16, 17], feature adaptation [5, 18] and classifier adaptation [2, 19]. The instance reweighting methods aim to assign source samples with different weights based on their similarities with target samples, such that the distribution shift between two domains can be reduced. The classifier adaptation methods manage to adapt the classifier trained on source domain data to target domain data. The feature adaptation methods are probably the most popular one. This kind of methods aim to find a common feature space to reduce the distribution discrepancy between two domains. Critically, conventional methods assume that the source and target domains share the same label set, which may be violated in many practical applications. Recently, Cao et al. [8] propose the partial domain adaption (PDA), where the source label set is large enough to completely cover the target label set. Traditional DA methods could be vulnerable to negative transfer in PDA due to the mismatch of label sets between the source and target domains [8].

One way to solve PDA problem is to decrease the influence of the source-private classes and enhance the influences of the shared classes. For example, Cao et al. [20] develop a class-level weighting mechanism to down-weigh the samples of the source-private classes. Li et al. [13] propose the deep residual correction network to intrinsically address the inherent problem in PDA and introduce a weighted class-wise alignment loss to identify the shared classes. To entirely circumvent negative transfer, Wang et al. [21] detect and remove the source-private classes progressively, and employ the label propagation algorithm to assign the pseudo-labels for target domain data.

Different from the above strategy to select source samples from class-level, Zhang et al. [11] propose a sample-level weighting mechanism to recognize the source samples that are potentially from the shared classes. Cao et al. [14] integrate the discriminative information to quantify the transferability of source examples and down-weight the negative transfer of samples from the source-private classes upon the source classifier and the domain discriminator. Wu et al. [22] design a deep reinforcement learning based source samples selector for PDA, which owns the ability to automatically keep or filter out source samples based on their feature representations.

Recently, to borrow the advantages of the class-level strategy and the sample-level strategy, several works have integrated them to address the PDA problem. For instance, a reweighting network is designed by Li et al. [23] to provide class-level weights for source samples and sample-level weights for target samples. Kim et al. [12] propose to unify sample-level transferability and class-level feature propagation to solve PDA problem, which is based on adaptive graph adversarial networks, such that intra-domain and inter-domain structures between data samples can be fully exploited.

Although the above weight based methods can achieve promising performance, all source samples participate in the domain adaptation process. Consequently, the samples from source-private classes can still make adverse impact for PDA. Different from these methods, our PLSC aims to identify the shared classes and only uses the source samples within the shared classes to train a classifier for assigning the target pseudo-labels, which can relieve the negative transfer caused by the source-private classes. The SCS-LP method in [21] also employs the similar idea. However, our PLSC is significantly different from it. First, we employ a totally different strategy to learn the low-dimensional subspace. Specifically, SCS-LP uses the supervised locality preserving projection technique, while our PLSC learns the subspace under the guidance of general separation and alignment assumptions in DA [15], which makes our PLSC more suitable for PDA problem. Second, our PLSC further considers the noises in target data and designs an adaptive threshold larger than zero to identify the shared classes, which is more practical and owns better generalization capacity. Third, PLSC further introduces the self-paced learning mechanism, which can progressively select target samples to learn a better low-dimensional subspace.

2.2 Self-paced Learning

The goal of self-paced learning (SPL) [24] is to gradually incorporate the training samples from easy to hard to learn the model. This learning paradigm is inspired by the learning process of humans that gradually include easy to complex samples into training [25]. Supposing the training data is $\mathbf{X} = \{(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2), \cdots, (\mathbf{x}_n, \mathbf{y}_n)\}$ and the training model is f parameterized by $\boldsymbol{\theta}$, the general optimization problem of SPL can be stated as:

$$\min_{\boldsymbol{\theta},\mathbf{w}} \sum_{i=1}^{n} w_i L(f_{\boldsymbol{\theta}}(\mathbf{x}_i), \mathbf{y}_i) + h(\lambda, w_i) \tag{1}$$

where $L(\cdot)$ is the loss function for the given problem. $h(\lambda, w_i)$ denotes the SPL regulation term, which is independent of $L(\cdot)$ and has various definitions according to different problems. $\mathbf{w} = [w_1, w_2, \cdots, w_n]^{\mathrm{T}}$ represents the weight variable, and w_i reflects the complexity of sample $\mathbf{x}_i$. λ is the learning pace to control the model age, which progressively increases to incorporate more samples to the training process. When $h(\lambda, w_i) = -\lambda w_i$ and $w_i \in \{0, 1\}$, the optimization problem (1) degenerates into the hard-weight form, and we have:

Table 1 Notations and descriptions

Notations	Descriptions	Notations	Descriptions
$\mathbf{X}_s$	Source domain data	$\mathbf{X}_t$	Target domain data
$\mathbf{X}$	Data matrix for all samples	$\mathbf{1}_{p\times q}$	A $p \times q$ matrix with all elements as 1
n_s	Number of source samples	n_t	Number of target samples
$\mathbf{P}$	Projection matrix	m	Dimension of the original space
d	Dimension of the projected space	$\mathbf{H}$	Centering matrix
$\mathbf{I}_d$	A $d \times d$ identity matrix	C_s	Number of source classes
$\mathcal{Y}$	Shared classes set of two domains	$\mathbf{W}$	Weight matrix for self-paced learning
$\widehat{\mathbf{Y}}_t$	Pseudo-labels of target samples	$\boldsymbol{\mu}_c$	Class centroid of the c-th source class
n_s^c	Number of source samples in class c	n_t^c	Number of target samples in class c

$$\begin{aligned} &\min_{\boldsymbol{\theta},\mathbf{w}} \sum_{i=1}^{n} w_i L(f_{\boldsymbol{\theta}}(\mathbf{x}_i), \mathbf{y}_i) - \lambda w_i \\ &s.t.\ w_i \in \{0, 1\} \end{aligned} \tag{2}$$

When the model parameter $\boldsymbol{\theta}$ is fixed, the optimal solution of problem (2) with respect to w_i is:

$$w_i = \begin{cases} 1, & \text{if } L(f_{\boldsymbol{\theta}}(\mathbf{x}_i), \mathbf{y}_i) \le \lambda; \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

In addition, with $\mathbf{w}$ fixed, the optimization problem (2) with respect to $\boldsymbol{\theta}$ becomes a weighted loss minimization problem, which can be readily solved with the optimization algorithm for the original problem.

SPL has been widely studied in recent years. Supancic et al. [26] apply SPL to the long-term tracking tasks. Jiang et al. [27] consider to employ the prior information and develop a self-paced curriculum learning framework. To avoid the standard SPL to suffer from the class imbalance issue, Ren et al. [28] propose two novel soft-weighting schemes, which can assign weights and select samples locally for each class. Meng et al. [29] prove that solving the optimization problem of SPL in an alternative way is equivalent to solving a roust loss minimization problem via a majorization-minimization algorithm, which helps to provide a theoretical understanding for SPL. Different from the above works which obtain the weights of samples from losses, Shu et al. [30] propose to learn a weighting function directly based on deep neural networks. SPL has shown excellent performance in various tasks, such as feature selection [31, 32], clustering [33, 34] and person re-identification [35, 36].

3 Proposed Method

In this section, the notations frequently used in this paper and their descriptions are first introduced. Then, we describe our proposed method in detail. Next, the optimization procedure of our proposal is shown. Finally, we provide the complexity analysis about our method.

3.1 Notations

In this paper, we focus on the unsupervised PDA problem. There exists two domains in PDA including the well-labeled source domain and the unlabeled target domain. We denote the source domain data as $D_s = \{\mathbf{x}_s^i, y_s^i\}_{i=1}^{n_s}$, where $\mathbf{x}_s^i \in \mathbb{R}^{m\times 1}$ is a sample of source domain and its label is y_s^i, and n_s denotes the number of source samples. The target domain data are denoted as $D_t = \{\mathbf{x}_t^i\}_{i=1}^{n_t}$, where $\mathbf{x}_t^i \in \mathbb{R}^{m\times 1}$ represents a sample in target domain, and n_t denotes the number of target samples. The matrix notations of source and target data are denoted as $\mathbf{X}_s \in \mathbb{R}^{m\times n_s}$ and $\mathbf{X}_t \in \mathbb{R}^{m\times n_t}$, respectively. In PDA, the source label set is a subset of target label set. In Table 1, we summarize the frequently-used notations and their descriptions.

3.2 Problem Formulation

Our proposed PLSC contains two main parts: *(1) shared classes identification, (2) progressive target sample learning*. The proposed shared classes identification method includes three items: *source samples separation, target samples alignment and shared classes identification with adaptive threshold*. Next, we will introduce our PLSC in detail.

Source samples separation: In DA, there is a general separation assumption that the source data or the target data are discriminatively clustered in a suitable feature space [15]. Intuitively, for well-labeled source samples, it is expected to thoroughly exploit the precious discriminative information to make source samples well-separated. To achieve this goal, we propose to project the source data into a low-dimensional subspace where the sum of the distance from each source sample to its corresponding class center is minimized. To this end, the objective function of source samples separation can be written as [38]:

$$\min_{\mathbf{P}} \sum\nolimits_{c=1}^{C_s} \sum_{\mathbf{x}_s^i \in D_s^c} \|\mathbf{P}^{\mathrm{T}}\mathbf{x}_s^i - \mathbf{P}^{\mathrm{T}}\mu_c\|_F^2 = \min_{\mathbf{P}} \operatorname{tr}(\mathbf{P}^{\mathrm{T}}\mathbf{X}_s\mathbf{L}_s\mathbf{X}_s^{\mathrm{T}}\mathbf{P}) \tag{4}$$

where $\mathbf{P} \in \mathbb{R}^{m\times d}$ is the projection matrix, D_s^c denotes the source samples in the c-th class, μ_c represents the corresponding class centroid and $\mathbf{L}_s = \mathbf{I} - \mathbf{Y}_s(\mathbf{Y}_s^{\mathrm{T}}\mathbf{Y}_s)^{-1}\mathbf{Y}_s^{\mathrm{T}}$. C_s is the number of source classes. $\mathbf{Y}_s \in \mathbb{R}^{n_s\times C_s}$ denotes the label matrix of source samples with each element defined as $(\mathbf{Y}_s)_{ij} = 1$ if $y_s^i = j$, and $(\mathbf{Y}_s)_{ij} = 0$ otherwise.

Target samples alignment: DA usually assumes that in an appropriate space, the clusters corresponding to the identical class in two domains are geometrically close, which is known as alignment assumption [15]. Then, it is expected to make the clusters of two domains with respect to the same class be aligned closely. For this purpose, we propose to minimize the sum of distance from each target sample to its corresponding source class center for the low-dimensional subspace learning. In light of this, the objective function of target samples alignment can be formulated as:

$$\begin{aligned}
&\min_{\mathbf{P}} \sum_{c\in\mathcal{Y}} \sum_{\mathbf{x}_t^i \in D_t^c} \|\mathbf{P}^{\mathrm{T}}\mathbf{x}_t^i - \mathbf{P}^{\mathrm{T}}\mu_c\|_F^2 \\
&= \min_{\mathbf{P}} \sum_{c\in\mathcal{Y}} \sum_{\mathbf{x}_t^i \in D_t^c} (\mathbf{P}^{\mathrm{T}}\mathbf{x}_t^i(\mathbf{x}_t^i)^{\mathrm{T}}\mathbf{P} - \mathbf{P}^{\mathrm{T}}\mathbf{x}_t^i\mu_c^{\mathrm{T}}\mathbf{P} - \mathbf{P}^{\mathrm{T}}\mu_c(\mathbf{x}_t^i)^{\mathrm{T}}\mathbf{P} + \mathbf{P}^{\mathrm{T}}\mu_c\mu_c^{\mathrm{T}}\mathbf{P}) \\
&= \min_{\mathbf{P}} \operatorname{tr}(\mathbf{P}^{\mathrm{T}}\mathbf{X}_t\mathbf{X}_t^{\mathrm{T}}\mathbf{P}) - \operatorname{tr}(\mathbf{P}^{\mathrm{T}}\mathbf{X}_t\mathbf{A}^{\mathrm{T}}\mathbf{S}^{\mathrm{T}}\mathbf{B}^{\mathrm{T}}\mathbf{X}_s^{T}\mathbf{P}) \\
&\quad - \operatorname{tr}(\mathbf{P}^{\mathrm{T}}\mathbf{X}_s\mathbf{B}\mathbf{S}\mathbf{A}\mathbf{X}_t^{\mathrm{T}}\mathbf{P}) + \operatorname{tr}(\mathbf{P}^{\mathrm{T}}\mathbf{X}_s\mathbf{B}\mathbf{S}\mathbf{A}\mathbf{A}^{\mathrm{T}}\mathbf{S}^{\mathrm{T}}\mathbf{B}^{\mathrm{T}}\mathbf{X}_s\mathbf{P}))
\end{aligned} \tag{5}$$

where $\mathcal{Y}$ is the shared classes set of two domains, D_t^c denotes target samples in the c-th class, which is defined based on the pseudo-labels of target samples. In our experiments, we use a linear SVM[1] classifier to initialize the pseudo-labels of target samples. We denote $C_I = |\mathcal{Y}|$ and the pseudo-label of target sample $\mathbf{x}_t^i$ as $\widehat{y}_t^i$. Then, $\mathbf{S} \in \mathbb{R}^{C_s \times C_I}$ is a shared class indicator matrix, whose each element is $R_{ij} = 1$ if the i-th source class is the j-th shared class, and $R_{ij} = 0$ otherwise. $\mathbf{A} \in \mathbb{R}^{C_I \times n_t}$ is defined as $A_{ji} = 1$ if $\widehat{y}_t^i = j$, and $A_{ji} = 0$ otherwise. $\mathbf{B} \in \mathbb{R}^{n_s \times C_s}$ is a constant matrix and each entry is calculated as $B_{ij} = \frac{1}{n_s^c}$ if $y_s^i = j$, and $B_{ij} = 0$ otherwise, where n_s^c is the number of source samples in the c-th class.

Shared classes identification with adaptive threshold: In the ideal case, by combining Eq. (4) and Eq. (5), we can learn a suitable subspace where the separation and alignment assumptions can be well-satisfied simultaneously. In other words, after projecting the data of two domains into the subspace constructed by $\mathbf{P}$, we can utilize the source samples to train a standard classifier, e.g., SVM, to assign pseudo-labels for target samples. Ideally, the pseudo-label set is ought to keep consistent with the ground-truth shared label set. However, in practice, noises can be inevitably included in target data, which may make some samples deviate from the source class centers. As a result, these samples would be misclassified into source-private classes, which may cause the wrong identification of shared classes. To solve this issue, we propose an adaptive threshold strategy that first counts the number of target sample for each source class, and then determines a class to be one of the shared classes if the corresponding number is larger than an adaptive value $\gamma * n_t / C$. Finally, we can obtain the following formulation:

$$\mathcal{Y} = \{c \mid n_t^c > \gamma * n_t / C_s\} \tag{6}$$

where n_t^c is the number of target samples in the c-th class. In our experiments, we set the hyper-parameter $\gamma = 0.8$ for all cases.

To obtain a better performance, we alternately learn the low-dimensional subspace and identify the shared classes until convergence. Specifically, after identifying the shared classes, we reuse the labeled source data of all classes and pseudo-labeled target data of shared class to update projection matrix $\mathbf{P}$. And then, the shared classes $\mathcal{Y}$ are determined by Eq. (6). In the iteration process, as shown in Eq. (5), we involve all target samples of shared classes for subspace learning. However, in practical applications, the distances of target samples to their corresponding source class centers are significantly different. The subspace learning step may be misled by the target samples deviated from the source class centers. To tackle this issue, in the following, we further propose to learn target sample progressively.

Progressive Target Sample Learning: We borrow the advantage of self-paced learning mechanism and gradually include the target samples into the low-dimensional subspace learning process. We determine the difficulty of target samples by the Euclidean distances between them and their corresponding source class centers. For the sake of simplicity but without loss of generality, we adopt the hard-weight form for self-paced learning, i.e., the weight is 0 or 1. Then, based on Eq. (5), the objective function of progressive target sample learning can be stated as:

$$\begin{aligned}
&\min_{\mathbf{P},\mathbf{W}} \sum_{c \in \mathcal{Y}} \sum_{\mathbf{x}_t^i \in D_t^c} \|(\mathbf{P}^{\mathrm{T}} \mathbf{x}_t^i - \mathbf{P}^{\mathrm{T}} \mu_c) W_{ii}\|_F^2 - \lambda W_{ii} \\
&= \min_{\mathbf{P},\mathbf{W}} (\mathrm{tr}(\mathbf{P}^{\mathrm{T}} \mathbf{X}_t \mathbf{W} \mathbf{W}^{\mathrm{T}} \mathbf{X}_t^{\mathrm{T}} \mathbf{P}) - \mathrm{tr}(\mathbf{P}^{\mathrm{T}} \mathbf{X}_t \mathbf{W} \mathbf{W}^{\mathrm{T}} \mathbf{A}^{\mathrm{T}} \mathbf{S}^{\mathrm{T}} \mathbf{B}^{\mathrm{T}} \mathbf{X}_s^{T} \mathbf{P})
\end{aligned}$$

[1] https://www.csie.ntu.edu.tw/~cjlin/liblinear/.

$$-\mathrm{tr}(\mathbf{P}^{\mathrm{T}}\mathbf{X}_s\mathbf{BSAWW}^{\mathrm{T}}\mathbf{X}_t^{\mathrm{T}}\mathbf{P}) + \mathrm{tr}(\mathbf{P}^{\mathrm{T}}\mathbf{X}_s\mathbf{BSAWW}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{S}^{\mathrm{T}}\mathbf{B}^{\mathrm{T}}\mathbf{X}_s\mathbf{P}))$$
$$s.t. \quad W_{ii} \in \{0, 1\} \tag{7}$$

The Overall Formulation of PLSC: To avoid overfitting, we futher impose an F-norm regularization term $\|\mathbf{P}\|_F^2$ on the projection matrix $\mathbf{P}$. So far, by combining Eq. (4), (7) and $\|\mathbf{P}\|_F^2$, we can get the final formulation of our proposed PLSC:

$$\begin{aligned}&\min_{\mathbf{P},\mathbf{W},\mathcal{Y}} \sum_{c=1}^{C_s} \sum_{\mathbf{x}_s^i \in D_s^c} \|\mathbf{P}^{\mathrm{T}}\mathbf{x}_s^i - \mathbf{P}^{\mathrm{T}}\mu_c\|_F^2 \\ &+ \alpha \left(\sum_{c \in \mathcal{Y}} \sum_{\mathbf{x}_t^i \in D_t^c} \|(\mathbf{P}^{\mathrm{T}}\mathbf{x}_t^i - \mathbf{P}^{\mathrm{T}}\mu_c)W_{ii}\|_F^2 - \lambda W_{ii} \right) + \beta\|\mathbf{P}\|_F^2 \\ &s.t. \;\; \mathbf{P}^{\mathrm{T}}\mathbf{XHX}^{\mathrm{T}}\mathbf{P} = \mathbf{I}_d, \;\; W_{ii} \in \{0, 1\}\end{aligned} \tag{8}$$

where $\mathbf{X} = [\mathbf{X}_s, \mathbf{X}_t]$ is the data matrix for all source samples and target samples. $\mathbf{H}$ is centering matrix defined as $\mathbf{H} = \mathbf{I}_{n_s+n_t} - \frac{1}{n_s+n_t}\mathbf{1}_{(n_s+n_t)\times(n_s+n_t)}$. α and β are hyper-parameters. The first constraint is inspired from principal component analysis, which aims to maintain the data property in the projected feature space [5]. We denote $\mathbf{L} = \begin{bmatrix} \mathbf{L}_s + \alpha\mathbf{BSAWW}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{S}^{\mathrm{T}}\mathbf{B}^{\mathrm{T}} & -\alpha\mathbf{BSAWW}^{\mathrm{T}} \\ -\alpha\mathbf{WW}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{S}^{\mathrm{T}}\mathbf{B}^{\mathrm{T}} & \alpha\mathbf{WW}^{\mathrm{T}} \end{bmatrix}$, and then the optimization problem (8) can be reformulated as:

$$\begin{aligned}&\min_{\mathbf{P},\mathbf{W},\mathcal{Y}} \mathrm{tr}(\mathbf{P}^{\mathrm{T}}\mathbf{XLX}^{\mathrm{T}}\mathbf{P}) + \alpha \sum_{c \in \mathcal{Y}} \sum_{\mathbf{x}_t^i \in D_t^c} -\lambda W_{ii} + \beta\|\mathbf{P}\|_F^2 \\ &s.t. \;\; \mathbf{P}^{\mathrm{T}}\mathbf{XHX}^{\mathrm{T}}\mathbf{P} = \mathbf{I}_d, \;\; W_{ii} \in \{0, 1\}\end{aligned} \tag{9}$$

Kernelization: Similar to [5, 18, 37], the proposed PLSC approach can be extended for solving nonlinear problems through kernelization. Suppose the kernel mapping is $\psi : \mathbf{x} \rightarrow \psi(\mathbf{x})$, and then using the kernel tricks, we can obtain the kernel matrix of all samples, i.e., $\mathbf{K} = \psi(\mathbf{X}^{\mathrm{T}})\psi(\mathbf{X}) \in \mathbb{R}^{(n_s+n_t)\times(n_s+n_t)}$. Following [5, 18, 37], the nonlinear version of our proposal can be written as:

$$\begin{aligned}&\min_{\mathbf{P},\mathbf{W},\mathcal{Y}} \mathrm{tr}(\mathbf{P}^{\mathrm{T}}\mathbf{KLK}^{\mathrm{T}}\mathbf{P}) + \alpha \sum_{c \in \mathcal{Y}} \sum_{\mathbf{x}_t^i \in D_t^c} -\lambda W_{ii} + \beta\|\mathbf{P}\|_F^2 \\ &s.t. \;\; \mathbf{P}^{\mathrm{T}}\mathbf{KHK}^{\mathrm{T}}\mathbf{P} = \mathbf{I}_d, \;\; W_{ii} \in \{0, 1\}\end{aligned} \tag{10}$$

It is worth noting that optimization problem (10) and optimization problem (9) have the same formulation, which means they can be solved with the same optimization algorithm. Next, we will describe the detailed optimization procedure for problem (9).

3.3 Optimization Procedure

In the objective function of our PLSC in Eq.(9), we need to optimize three variables $\mathbf{P}$, $\mathbf{W}$ and $\mathcal{Y}$. As the objective function is not jointly convex for all variables, we update each of them alternatively while taking the other variables as constants. Specifically, we solve each subproblem as follows:

Algorithm 1: The Optimization Procedure of PLSC

```
Input: Source domain data $\{\mathbf{x}_s^i, y_s^i\}_{i=1}^{n_s}$; Target domain data $\{\mathbf{x}_t^i\}_{i=1}^{n_t}$; Hyper-parameters $\alpha$, $\beta = 0.01$;
       Subspace dimension $d$; Initial proportion $p_0 = 0.5$; Proportion increasing rate $\eta = 0.1$;
       Maximum iteration $T = 10$.
Output: Target pseudo-labels $\widehat{\mathbf{Y}}_t$; Projection matrix $\mathbf{P}$.
Initialize: Initialize target pseudo-labels $\widehat{\mathbf{Y}}_t$ with a linear SVM classifier in the original space and
initialize the shared classes $\mathcal{Y}_0$ by (6); Initialize $\lambda_0$ in the original space by (15) and $\mathbf{W}$ by (14).
$t = 1$;
while $t \leq T$ do
    // Projection matrix $\mathbf{P}$
    Update $\mathbf{P}$ by solving (12);
    // Assign target pseudo-labels
    Train a linear SVM classifier on the source samples within the shared classes $\mathcal{Y}_{t-1}$ and use it to
    update target pseudo-labels;
    // Shared classes identification
    Identify the shared classes $\mathcal{Y}_t$ according to (6);
    // Update self-paced learning parameter
    Update proportion $p_t$ by (16) and calculate its corresponding self-paced learning parameter $\lambda$ by
    (15);
    // Weight matrix $\mathbf{W}$
    Update weight matrix $\mathbf{W}$ by (14);
    $t = t + 1$;
end
Return Target pseudo-labels $\widehat{\mathbf{Y}}_t$; Projection matrix $\mathbf{P}$.
```

P-subproblem: When **W** is took as a constant matrix and $\mathcal{Y}$ is fixed, we have the following subproblem:

$$\begin{aligned} &\min_{\mathbf{P}} \operatorname{tr}(\mathbf{P}^{\mathrm{T}}\mathbf{X}\mathbf{L}\mathbf{X}^{\mathrm{T}}\mathbf{P}) + \beta\|\mathbf{P}\|_F^2 \\ &s.t.\ \ \mathbf{P}^{\mathrm{T}}\mathbf{X}\mathbf{H}\mathbf{X}^{\mathrm{T}}\mathbf{P} = \mathbf{I}_d \end{aligned} \tag{11}$$

We can readily transform the above optimization problem to a generalized eigenvalue problem as follows:

$$(\mathbf{X}\mathbf{L}\mathbf{X}^{\mathrm{T}} + \beta\mathbf{I}_m)\mathbf{P} = \mathbf{X}\mathbf{H}\mathbf{X}^{\mathrm{T}}\mathbf{P}\mathbf{\Phi} \tag{12}$$

where $\mathbf{\Phi} = \operatorname{diag}(\phi_1, \phi_2, \cdots, \phi_d) \in \mathbb{R}^{d\times d}$ is a diagonal matrix and each diagonal element is a Lagrange Multiplier. Then, we can obtain the optimal **P** by computing the eigenvectors of (12) regarding the d-smallest eigenvalues.

$\mathcal{Y}$-subproblem: In the projection subspace constructed by **P**, we first train a linear SVM classifier on the source samples within the shared classes $\mathcal{Y}_{t-1}$ and use the classifier to assign pseudo-labels for all target samples. Then, we can update the shared label set $\mathcal{Y}_t$ according to (6).

W-subproblem: When **P** and $\mathcal{Y}$ are fixed, the problem (9) becomes:

$$\min_{\mathbf{W}} \sum_{c\in\mathcal{Y}} \sum_{\mathbf{x}_t^i \in D_t^c} \|(\mathbf{P}^{\mathrm{T}}\mathbf{x}_t^i - \mathbf{P}^{\mathrm{T}}\mu_c)W_{ii}\|_F^2 - \lambda W_{ii},\ \ W_{ii} \in \{0, 1\} \tag{13}$$

Denote $l_i = \|\mathbf{P}^{\mathrm{T}}\mathbf{x}_t^i - \mathbf{P}^{\mathrm{T}}\mu_c\|_F^2$, then the optimal weight for each sample is:

$$W_{ii} = \begin{cases} 1, & \text{if } l_i \leq \lambda \wedge c \in \mathcal{Y}; \\ 0, & \text{otherwise.} \end{cases} \tag{14}$$

The parameter λ controls the learning pace of new target examples, which usually iteratively increases during optimization.

Update self-paced learning parameter λ: We set λ to guarantee that $p_t * n_t$ easy target samples are selected, where p_t is the proportion in the t-th iteration. In this iteration, we can calculate the Euclidean distance d_i of each target sample $\mathbf{x}_t^i$ to its corresponding source class center in the low-dimensional subspace. Then, we can sort these distances from small to large. Denote the sorted distances as $d_{\text{sort}}^1, d_{\text{sort}}^2, \cdots, d_{\text{sort}}^{n_t}$, and we have:

$$\lambda = d_{\text{sort}}^{\text{floor}(p_t * n_t)} \tag{15}$$

where floor$(\cdot)$ is the round toward negative infinity function in MATLAB, which rounds the input to the nearest integer less than or equal to the input. At last, the update of self-paced learning parameter λ is transformed to update the proportion p in each iteration by the following function:

$$p_t = \min(p_{t-1} + \eta, 1) \tag{16}$$

where η denotes the proportion increasing rate.

We use a linear SVM[1] to assign the pseudo-labels for target samples. We initialize the proportion to 0.5, i.e. $p_0 = 0.5$, and set the proportion increasing rate $\eta = 0.1$. Algorithm 1 summarizes the optimization procedure of PLSC.

3.4 Complexity Analysis

The optimization Algorithm 1 contains two main parts, project matrix learning and learning pace parameter updating, within T iterations. Note that the time cost to assign target pseudo-labels by a linear SVM is ignored as it can be very fast. Concretely, constructing problem (11) costs $\mathcal{O}(n_s n_t^2)$ and obtaining the projection matrix $\mathbf{P}$ occupies $\mathcal{O}(m^2 d)$. Calculating the distances of target samples to the source clusters and sorting the distances to update the learning pace parameter λ needs a time cost of $\mathcal{O}(n_t^2 \log(n_t))$. Therefore, the overall computational complexity is $\mathcal{O}(T m^2 d + T n_s n_t^2 + T n_t^2 \log(n_t))$.

4 Experiment and Analysis

In this section, we first illustrate the four benchmark datasets used for evaluation. Then, the details of the experimental setup including comparison methods, parameter setting and evaluation metric are described. Next, we show the experimental results of PDA. Finally, several analytical experiments are further conducted to understand our method more deeply.

4.1 Datasets and Descriptions

We evaluate the performance of our method and other methods on four widely used public datasets in PDA: Office31 [39], Office-Home [40], ImageCLEF[2] and Visda2017 [41]. We summarize the overall descriptions of these datasets in Table 2. For simplicity, in our experiments, each PDA task is denoted by S →T, where S represents the source domain and T is the target domain. Next, we will introduce these datasets and the corresponding PDA tasks in detail.

[2] http://imageclef.org/2014/adaptation.

Table 2 Statistics of the four benchmark datasets

Datasets	Subsets (Abbr.)	Samples	*Classes*
Office31	Amazon (A)	2,817	31
	DSLR (D)	498	
	Webcam (W)	795	
Office-Home	Art (Ar)	2,421	65
	Clipart (Cl)	4,379	
	Product (Pr)	4,428	
	RealWorld (Re)	4,357	
ImageCLEF	ImageNet ILSVRC2012 (I)	600	12
	Caltech-256 (C)	600	
	Pascal VOC2012 (P)	600	
Visda2017	train (T)	152,397	12
	validation (V)	55,388	

Office31 [39] includes 4,110 images with 31 classes. These images are collected from three domains: Amazon (A), DSLR (D) and Webcam (W). Amazon domain downloads images from the online merchants. DSLR domain obtains images by a digital SLR camera while Webcam domain captures images by a web camera. In our experiments, we employ the Resnet50 features[3] provided by [42], which are extracted by a Resnet50 model [43] pre-trained on ImageNet. Following [20], we use all 31 classes for source domain and 10 classes shared between Office31 dataset and Caltech-256 dataset [44] for the target domain. We have six PDA tasks, i.e., A→D, A→W, $\cdots$, W→D.

Office-Home [40] consists of 15,585 object images in 65 categories from four domains: Art (Ar), Clipart (Cl), Product (Pr) and RealWorld (Re). Images of Art domain are artistic description of objects. Clipart domain contains clipart images. Images of Product domain have no background. RealWorld domain obtains images by a regular camera. Similar to [20], we utilize the Resnet50 features[3] in our experiments. Besides, the source domain contains images of all 65 categories, while the target domain includes the images of the first 25 categories in alphabetical order. Finally, we can obtain twelve PDA tasks, i.e., Ar→Cl, Ar→Pr, $\cdots$, Re→Pr.

ImageCLEF[2] dataset is first presented in the ImageCLEF Domain Adaptation challenge, which is held in 2014. This dataset contains three different domains: ImageNet ILSVRC2012 (I), Caltech-256 (C), and Pascal VOC2012 (P). Each of the three domains has 12 classes and each class consists of 50 images. Following the previous work [45], for each PDA task, the source domain includes all 600 images and the target domain contains the images of the first six 6 classes in alphabetical order. In our experiments, similar to [45], we utilize the Resnet50 model[4] pretrained on ImageNet to extract the Resnet50 features. Finally, six PDA tasks are established, including I→C, I→P, $\cdots$, P→C.

Visda2017 [41] is first released in the 2017 Visual Domain Adaptation Challenge. This dataset is made up of large number of synthetic images and real images. Specially, the training data has 152,397 synthetic images while the validation data has 55,388 real images. Following

[3] https://github.com/hellowangqian/domainadaptation-capls.

[4] https://github.com/jindongwang/transferlearning/tree/master/code/feature_extractor/for_image_data.

[20], each of the training data and validation data can form a domain, which is abbreviated as T and V, respectively. The resnet50 features[5] provided by [4] are employed for experiments. Like [20], images of all 12 categories comprise the source domain, while the first 6 categories in alphabetical order do the target domain. In our experiments, we have two PDA tasks, T→V and V→T.

4.2 Experimental Setup

Comparison Methods: The proposed method is compared with several PDA methods including:

- *PADA*, Partial Adversarial Domain Adaptation [20], which identifies source-private classes and down-weighs their importance based on label predictions of target samples.
- *ETN*, Example Transfer Network [14], which introduces a progressive weighting strategy to quantify the transferability of each source instance based on its similarity to target domain.
- *SAN*, Selective Adversarial Network [8], which selects the source-privates classes based on the output of C_s class-wise domain discriminators.
- *DRCN*, Deep Residual Correction Network [13], which plugs a residual block into a unified network to further capture the feature discrepancy and develops a weighting scheme to identify the shared classes.
- *AGAN*, Adaptive Graph Adversarial Networks [12], which designs a class-relational graph module to achieve structure-aware domain alignments and a sample-level commonness predictor to compute the commonness for each sample.
- RTNet$_{adv}$, Reinforced Transfer Network with the the reinforced data selector into the domain-adversarial training of neural networks [46], which employs the selector to filter out the source-private classes and introduces a state containing high-level information to select sample.
- *DMP*, Discriminative Manifold Propagation [45], which employs the manifold alignment and discriminative embedding to learn domain-invariant features and develops a weighting scheme to alleviate negative transfer from the source-private classes.
- *SCS-LP*, Source Class Selection with Label Propagation [21], which progressively detects and excludes the source-private classes, and employs the label propagation to assign the target pseudo-labels.
- *DRL-DS*, Deep Reinforcement Learning based source Data Selector [22], which utilizes a deep reinforcement learning based source data selector to eliminate the source samples from the source-private classes.

Parameter Setting: In current PDA task, target ground-truth labels are unavailable, thus a standard cross-validation procedure can not be performed to select the optimal parameters. For each comparison method, for fairness, we directly cite the results from the original paper. In our methods, there are four hyper-parameters: α, β, γ and d. We fix $\beta = 0.01$, $\gamma = 0.8$ for all datasets, leaving α and d tunable. We obtain the optimal parameters by searching $\alpha \in \{0.5, 1.0, 2.5\}$ and $d \in \{50, 100, 150, 200\}$. We also provide the optimal parameters used in this paper for PDA task: Office31 ($\alpha = 2.5$, $d = 50$), Office-Home ($\alpha = 1.0$, $d = 150$), ImageCLEF ($\alpha = 1.0$, $d = 50$) and Visda2017 ($\alpha = 2.5$, $d = 200$). In our experiments, we use the kernel version of our PLSC on Office31, Office-Home and

[5] https://github.com/LeiTian-qj/CMMS/tree/master/data/Visda2017.

Table 3 Accuracy (%) of all methods on Office31 dataset

Task	PADA	ETN	SAN	DRCN	AGAN	RTNet$_{\text{adv}}$	DMP	DRL-DS	SCS-LP	PLSC
A→D	82.2	95.0	94.3	86.0	94.3	97.6	96.4	96.0	**100.0**	**100.0**
A→W	86.5	94.5	93.9	88.5	97.3	96.2	96.6	96.6	99.0	**99.3**
D→A	92.7	**96.2**	95.1	95.6	95.7	92.3	95.1	95.4	94.3	96.0
D→W	99.3	**100.0**	99.3	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**
W→A	95.4	94.6	88.7	95.8	95.7	95.4	95.4	95.3	95.4	**96.6**
W→D	**100.0**	**100.0**	99.4	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**
Average	92.7	96.7	95.0	94.3	97.2	96.9	97.2	97.2	98.1	**98.7**

ImageCLEF datasets, and Gaussian kernel with kernel width 1.5 is adopted. On Visda2017 dataset, we use the original version due to limited memory space.

Evaluation Metric: Similar to many previous PDA methods [13, 20], we adopt the accuracy of target samples as the evaluation metric, which can be computed by:

$$\text{accuracy} = \frac{\mid \mathbf{x} : \mathbf{x} \in \mathbf{X}_t \cap \tilde{y} = y \mid}{\mid \mathbf{x} : \mathbf{x} \in \mathbf{X}_t \mid} \tag{17}$$

where $\mathbf{x}$ is a target sample with the ground-truth label as y, and $\tilde{y}$ is the label obtained by the proposed PDA method.

4.3 Experimental Results

Results on Office31 Dataset: The experimental results of all methods on Office31 dataset are displayed in Table 3. In Table 3, the highest accuracy for each task is boldfaced. As we can see, our proposal achieves 98.7% average classification performance, which leads the best competitor SCS-LP by 0.6%. Besides, our method works the best for five out of all six tasks. It is worth noting that the second best method SCS-LP does not perform better than our approach on any task. The above results can illustrate the superiority of our method over the counterparts.

Results on Office-Home Dataset: We summarize the classification results of all methods on Office-Home dataset in Table 4. We can observe that our proposal is superior to all competitors with respect to the average classification accuracy. Specifically, our method owns 1.2% improvement against the best competitor SCS-LP in average performance. Our method is the best one for six out of all twelve tasks, while the best competitor SCS-LP only wins three tasks. These results confirm the significant effectiveness of our proposal.

Results on ImageCLEF and Visda2017 Datasets: The classification performances of several competitors and our PLSC on ImageCLEF dataset and Visda2017 dataset are displayed in Table 5. As we can see, on ImageCLEF dataset, our PLSC is the best method on five out of all six tasks and leads the best competitor DMP by 1.8% in average performance, which demonstrates the superiority of our method. Besides, on Visda2017 dataset, our proposal wins both tasks and owns 4.4% improvement over the second best method AGAN in terms of the average accuracy. This phenomenon indicates that our PLSC has excellent capability to deal with PDA tasks with massive samples.

Table 4 Accuracy (%) of all methods on Office–Home dataset

Task	PADA	ETN	SAN	DRCN	AGAN	RTNet$_{adv}$	DMP	DRL-DS	SCS-LP	PLSC
Ar→Cl	52.0	59.2	44.4	54.0	56.4	63.2	59.0	61.0	**65.0**	63.2
Ar→Pr	67.0	77.0	68.7	76.4	77.3	80.1	81.2	80.8	81.2	**85.7**
Ar→Re	78.7	79.5	74.6	83.0	85.1	80.7	86.3	84.5	90.0	**91.6**
Cl→Ar	52.2	62.9	67.5	62.1	74.2	66.7	68.1	**75.5**	70.0	72.1
Cl→Pr	53.8	65.7	65.0	64.5	73.8	69.3	72.8	75.8	**81.7**	80.2
Cl→Re	59.0	75.0	77.8	71.0	81.1	77.2	78.8	80.1	81.7	**82.7**
Pr→Ar	52.6	68.3	59.8	70.8	70.8	71.6	71.2	76.0	70.2	**78.7**
Pr→Cl	43.2	55.4	44.7	49.8	51.5	53.9	57.6	**60.1**	55.4	56.1
Pr→Re	78.8	84.4	80.1	80.5	84.5	84.6	84.9	83.4	82.8	**86.0**
Re→Ar	73.7	75.7	72.2	77.5	79.0	77.4	77.3	79.0	**79.2**	76.6
Re→Cl	56.6	57.7	50.2	59.1	56.8	57.9	61.5	**64.3**	60.3	58.9
Re→Re	77.1	84.5	78.7	79.9	83.4	85.5	82.9	83.2	87.4	**87.6**
Average	62.1	70.5	65.3	69.0	72.8	72.3	73.5	75.3	75.4	**76.6**

Table 5 Accuracy (%) of several methods on ImageCLEF and Visda2017 datasets

Method	I→C	I→P	C→I	C→P	P→I	P→C	Average	Method	T→V	V→T	Average
PADA	94.6	81.7	89.8	77.7	82.1	94.1	88.3	PADA	53.5	76.5	65.0
SAN	95.9	81.6	90.4	78.5	91.1	97.1	89.1	DRCN	58.2	73.2	65.7
DMP	96.7	82.4	94.3	**78.7**	94.5	96.4	90.5	AGAN	67.7	80.5	74.1
PLSC	**98.3**	**85.0**	**96.3**	78.0	**97.3**	**99.0**	**92.3**	PLSC	**74.9**	**82.2**	**78.5**

4.4 Analytical Experiments

In this section, several experiments are further conducted to pursue deeper understanding for our proposed PLSC approach.

Effectiveness of self-paced learning: In our approach, we employ the SPL to gradually select target samples for training. To verify the effectiveness of SPL, we propose a variant of our proposal, which Removes the SPL mechanism and always trains the model with all target samples (PLSC$_{\mathrm{rs}}$). The results of our PLSC and the variant PLSC$_{\mathrm{rs}}$ are shown in Table 6. As we can see, PLSC performs better than PLSC$_{\mathrm{rs}}$ on all four datasets, which demonstrates the effectiveness of SPL. By introducing the SPL, our method can gradually select easy target samples, which can provide more convincing guidance for projection matrix learning, and thus boosting the performance of PDA.

The consistency between identified shared classes by our method and the ground-truth shared classes: In our PLSC, we design an adaptive threshold to identify the shared classes. In Table 7, we display the shared classes identified by our method and the real shared classes on task W→A, Cl→Re, I→P and T→V. We can see on task W→A, the identified shared classes and the real shared classes are the same. On task Cl→Re, the identified classes do not include the 12-th and the 16-th classes while contain an addition class. On task I→P, compared with the real shared classes, the identified classes contain one other class. On task T→V, the identified shared classes do not contain the 6-th class and have two others. As we

Table 6 Average classification accuracy (%) of PLSC$_{\text{rs}}$ and PLSC on four datasets

	Office31	Office-Home	ImageCLEF	Visda2017
PLSC$_{\text{rs}}$	96.1	75.3	91.7	73.2
PLSC	**98.7**	**76.6**	**92.3**	**78.5**

Table 7 The shared class numbers of the two domains selected by our method and the real shared class numbers on task W→A, Cl→Re, I→P and T→V

Task	D→A	Cl→Re	I→P	T→V
Selected	1,2,6,11,12,13,16,17,18,23	1-11,13-15,17-25,30	1-6,12	1-5,8,9
Real	1,2,6,11,12,13,16,17,18,23	1-25	1-6	1-6

Table 8 Average classification accuracy (%) on Office31 dataset with varying p ($p \in \{0.2, 0.3, \cdots, 0.8\}$) and η ($\eta \in \{0.02, 0.04, \cdots, 0.2\}$)

	0.02	0.04	0.06	0.08	0.10	0.12	0.14	0.16	0.18	0.20
0.2	95.6	96.2	97.0	97.6	98.5	97.9	98.5	97.4	97.7	97.7
0.3	96.1	96.6	97.7	97.5	98.5	98.5	97.6	96.9	97.0	98.2
0.4	96.3	97.3	97.8	97.0	98.6	97.9	97.2	97.8	97.1	97.6
0.5	96.4	97.1	97.8	98.7	98.7	97.4	97.8	97.3	97.3	97.1
0.6	96.8	97.7	97.5	97.8	98.6	97.5	97.2	97.5	97.2	97.2
0.7	96.9	96.7	96.9	96.9	96.9	97.0	96.4	96.1	96.2	96.0
0.8	95.5	95.6	95.7	95.2	95.1	94.7	94.7	94.6	94.4	94.6

see, the shared classes identified by our method have little difference with the real shared classes. The above results verify that even in an unsupervised manner, our method owns a great potentiality to identify the real shared classes, which validates the effectiveness of our method.

Influence of initial proportion and increasing rate when involving target samples progressively: In this part, we investigate the sensitivity of initial proportion p and increasing rate η in Algorithm 1. To be specific, we fix other parameters and vary p in the range of $\{0.2, 0.3, \cdots, 0.8\}$ and η in the range of $\{0.02, 0.04, \cdots, 0.2\}$. The average classification accuracy on Office31 dataset are summarized in Table 8. Carefully looking at this table, we can observe that we can obtain a relative good average performance when setting $p \in [0.2, 0.6]$ and $\eta \in [0.06, 0.12]$. In addition, the results of this table also verify the effectiveness of SPL, since the majority of the average classification accuracies are higher than that of PLSC$_{\text{rs}}$ in Table 6 (i.e., 96.1%). Actually, we set $p = 0.5$ and $\eta = 0.1$ in this paper. It is worth noting that in our PLSC, we set the parameter λ controlling the learning pace to ensure that $p * n_t$ samples are selected. Specifically, we can sort the distances of target samples to their source clusters from small to large and set λ to the value of the floor$(p * n_t)$-th largest distance, where floor$(\cdot)$ is the round toward negative infinity function in MATLAB. The formulation to update λ is (15).

Impact of varying number of target classes: To verify the effectiveness of our method with varying number of target classes, we select task Cl→Pr and compare the performances of our method and the best competitor SCS-LP on this task. We show the results in Fig. 3, where the

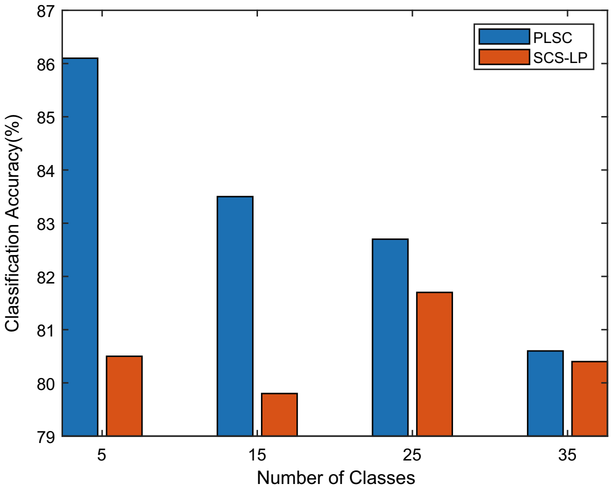


Fig. 3 Classification accuracy with varying number of target classes

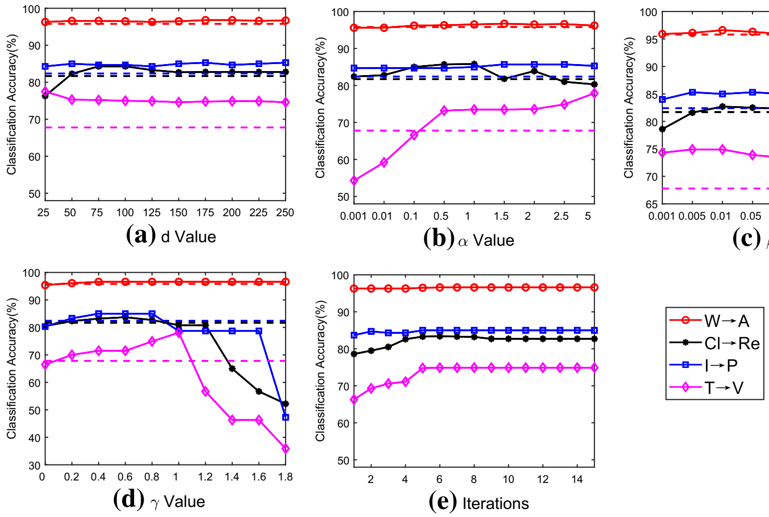
Table 9 Average accuracy (%) of PLSC with differenet kernels on Office31, Office-Home and ImageCLEF datasets. σ represents the width of gaussian kernel

kernel	Office31	Office-Home	ImageCLEF	Avearage
linear	96.1	73.0	92.2	87.1
gaussian ($\sigma = 1.0$)	97.4	73.8	92.2	87.8
gaussian ($\sigma = 1.25$)	97.9	75.0	92.2	88.4
gaussian ($\sigma = 1.5$)	**98.7**	**76.6**	92.3	**89.2**
gaussian ($\sigma = 1.75$)	97.7	75.6	**92.6**	88.6
gaussian ($\sigma = 2.0$)	97.6	75.5	92.4	88.5

number of target classes is set to {5, 15, 25, 35}. Besides, the results of SCS-LP are obtained by running the public codes with default parameters. We can see that our PLSC consistently outperforms SCS-LP when the number of target classes is smaller than 35. Besides, it is observed that our PLSC is more effective for PDA when the label mismatch between two domains is larger.

Influence of kernels configuration: In our experiments, we use the gaussian kernel with 1.5 on Office31, Office-Home and ImageCLEF datasets. In this part, we further conduct experiments to explore the influences of kernels on the performance of our PLSC. Specifically, we run our PLSC with linear kernel and gaussian kernel with different kernel widths. These two kinds of kernel are widely utilized by previous domain adaptation works [5, 18, 38]. In Table 9, we summarize the average accuracies of PLSC with different kernels on Office31, Office-Home and ImageCLEF datasets, where σ represents the width of gaussian kernel. As we can see, the gaussian kernel with $\sigma = 1.5$ achieves the highest average performance. Thus, we choose this kernel in our experiments.

Parameter sensitivity and convergence analysis: In our PLSC, there exists four parameters: d, α, β and γ. We have conducted extensive experiments to investigate the sensitivity of the four parameters. Specifically, we vary one parameter once in a wide range with the other



(a) d Value **(b)** α Value **(c)** β Value **(d)** γ Value **(e)** Iterations

Fig. 4 Parameter sensitivity with respect to d, α, β, γ and convergence analysis

parameters fixed as the optimal values. The results of task W→A, Cl→Re, I→P and T→V are displayed in Fig. 4a–d, where the results of the best competitor for each task are also shown as the dash lines. First, we run our PLSC as d varies $d \in \{25, 50, \cdots, 250\}$. From Fig. 4a, we can find that our PLSC can perform consistently better than the corresponding best competitor when d is located within a wide range [50, 250]. Next, we investigate the influence of α by varying it in a wide range [0.001, 0.01, 0.1, 0.5, 1.0, 1.5, 2.0, 2.5, 5.0]. Theoretically, a small α will make the target samples to source clusters progressively minimization term less ineffective. In such case, each target sample is not near to its corresponding source cluster, which hinders the correctness of target pseudo-labels assignment. By contrast, a large α will dominate the objective function and the source within-class scatter minimization is not performed. Then, the source samples can not be well cluster, which can also effect the accuracy of target pseudo-labels assignment. We empirically observe that when α is located in a reasonable range, i.e., $\alpha \in [0.2, 2.0]$, our PLSC can be superior to the best competitor. Then, we explore the influence of β by setting $\beta \in [0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1.0, 5.0, 10.0]$. As we can see from Fig. 4c, it is infeasible to determine the optimal value of β, since it highly depends on the domain prior knowledge of the datasets. However, we empirically find that,when β is located within the range [0.005, 0.5], our PLSC can obtain better classification results than the most competitive competitor. Next, we vary the value of γ from 0.0 to 1.8 to evaluate its influence. Theoretically, as a result of the noises containing in target data, too small values of γ will make some source-private classes to be identified as shared classes, which causes negative transfer. By contrast, too large values of γ will hinder some real shared classes to be identified as shared classes. A proper value of γ helps to identify the shared classes more accurately, thereby improving the performance of partial domain adaptation. From Fig. 4d, we can discover that $\gamma \in [0.2, 0.8]$ is an optimal choice. In our experiments, to avoid tuning too many parameters, we fix $\beta = 0.01$ and $\gamma = 0.8$ for all tasks. Finally, we depict the convergence analysis in Fig. 4e, where the maximum iteration number is set to 15. It is observed that the proposed method can quickly converge within 10 iterations.

5 Conclusion

In this paper, we propose a novel method named PLSC for solving PDA problem. In PLSC, we borrow the idea of separation and alignment assumptions in DA to identify the shared classes. To instantiate these two assumptions, we propose to minimize the sum of the distances from both source and target samples to their corresponding source class centers. Considering the fact that in practical applications the noises in target data may result in wrong identification of shared classes, we design an adaptive threshold strategy to determine the shared classes. Additionally, to relieve the misleading of target samples that deviate from their corresponding source class centers, we further introduce the self-paced learning mechanism into our PLSC to progressively select target samples for projection matrix learning. Extensive experiments on Office31, Office-Home, ImageCLEF and Visda2017 datasets validate the superiority of our method against the current PDA methods. In the future, we will manage to apply our strategy to deep learning scenario and elaborately design an end-to-end method.

Acknowledgements The authors are thankful for the financial support by the the Key-Area Research and Development Program of Guangdong Province 2019B010153002 and the National Natural Science Foundation of China 62106266, U1936206, 61961160707 and 61976212.

References

1. Pan S, Yang Q (2010) A survey on transfer learning. IEEE Trans Knowl Data Eng 22(10):1345–1359
2. Long M, Wang J, Ding G, Pan SJ, Yu PS (2013) Adaptation regularization: a general framework for transfer learning. IEEE Trans Knowl Data Eng 26(5):1076–1089
3. Wang J, Li X, Du J (2019) Label space embedding of manifold alignment for domain adaption. Neural Process Lett 49:375–391
4. Tian L, Tang Y, Hu L, Ren Z, Zhang W (2019) Domain adaptation by class centroid matching and local manifold self-learning. IEEE Trans Image Process 29:9703–9718
5. Long M, Wang J, Ding G, Sun J, Yu PS (2013) Transfer feature learning with joint distribution adaptation. In: IEEE international conference on computer vision (ICCV), pp 2200–2207
6. Zhang C, Tang Y, Zhang Z, Li D, Yang X, Zhang W (2020) Improving domain-adaptive person re-identification by dual-alignment learning with camera-aware image generation. IEEE Trans Circuits Syst Video Technol 31(11):4334–4346
7. Bai Z, Wang Z, Wang J, Hu D, Ding E (2021) Unsupervised multi-source domain adaptation for person re-identification. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), pp 12914–12923
8. Cao Z, Long M, Wang J, Jordan M (2018) Partial transfer learning with selective adversarial networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), pp 2724–2732
9. Deng J, Dong W, Socher R, Li LJ, Li K, Li F-F (2009) Imagenet: a large-scale hierarchical image database. In: Proceedings of computer vision and pattern recognition (CVPR), pp 248–255
10. Lin TY, Maire M, Belongie S, Hays J, Perona P, Ramanan D, Dollar P, Zitnick CL (2014) Microsoft coco: common objects in context. In: Proceedings of European conference on computer vision (ECCV), pp 740–755
11. Zhang J, Ding Z, Li W, Ogunbona P (2018) Importance weighted adversarial nets for partial domain adaptation. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), pp 8156–8164
12. Kim Y, Hong S (2021) Adaptive graph adversarial networks for partial domain adaptation. IEEE Trans Circuits Syst Video Technol 32:172–182
13. Li S, Liu C, Lin Q, Wen Q, Su L, Huang G, Ding Z (2021) Deep residual correction network for partial domain adaptation. IEEE Trans Pattern Anal Mach Intell 43(7):2329–2344
14. Cao Z, You K, Long M, Wang J, Yang Q (2019) Learning to transfer examples for partial domain adaptation. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), pp 2985–2994

15. Shi Y, Sha F (2012) Information-theoretical learning of discriminative clusters for unsupervised domain adaptation. In: Proceedings of the 29th international conference on international conference on machine learning, pp 1275–1282
16. Long M, Wang J, Ding G, Sun J, Yu PS (2014) Transfer joint matching for unsupervised domain adaptation. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), pp 1410–1417
17. Sugiyama M, Krauledat M, Muller KR (2007) Covariate shift adaptation by importance weighted cross validation. J Mach Learn Res 1010(8):985–1005
18. Li S, Song S, Huang G, Ding Z, Wu C (2018) Domain invariant and class discriminative feature learning for visual domain adaptation. IEEE Trans Image Process 27(9):4260–4276
19. Wang J, Feng W, Chen Y, Yu H, Huang M, Yu PS (2018) Visual domain adaptation with manifold embedded distribution alignment. In: Proceedings of ACM international conference on multimedia, pp 402–410
20. Cao Z, Ma L, Long M, Wang J (2018) Partial adversarial domain adaptation. In: Proceedings of the European conference on computer vision (ECCV), pp 135–150
21. Wang Q, Breckon T P (2021) Source class selection with label propagation for partial domain adaptation. In: IEEE international conference on image processing (ICIP), pp 769–773
22. Wu K, Wu M, Yang J, Chen Z, Li Z, Li X (2021) Deep reinforcement learning boosted partial domain adaptation. In: Proceedings of the thirtieth international joint conference on artificial intelligence, pp 3192–3199
23. Li L, Wang Z, He H (2020) Dual alignment for partial domain adaptation. IEEE Trans Cybern 51(7):3404–3416
24. Kumar MP, Packer B, Daphne K (2010) Self-paced learning for latent variable models. In: Advances in neural information processing systems, pp 1–9
25. Jiang L, Meng D, Yu S, Lan Z, Shan S, Hauptmann AG (2014) Self-paced learning with diversity. In: Advances in neural information processing systems, vol 27, pp 2078–2086
26. Supancic JS, Ramanan D (2013) Self-paced learning for long-term tracking. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), pp 2379–2386
27. Jiang L, Meng D, Zhao Q, Shan S, Hauptmann AG (2015) Self-paced curriculum learning. In: AAAI conference on artificial intelligence
28. Ren Y, Zhao P, Sheng Y, Yao D, Xu Z (2017) Robust softmax regression for multi-class classification with self-paced learning. In: International joint conference on artificial intelligence
29. Meng D, Zhao Q, Jiang L (2017) A theoretical understanding of self-paced learning. Inf Sci 414:319–328
30. Shu J, Xie Q, Yi L, Zhao Q, Zhou S, Xu Z, Meng D (2019) Meta-weight-net: learning an explicit mapping for sample weighting. In: Advances in neural information processing systems, pp 1919–1930
31. Li Y, Ma C, Tao Y, Hu Z, Su Z, Liu M (2021) A robust cost-sensitive feature selection via self-paced learning regularization. Neural Process Lett 1–18
32. Zheng W, Zhu X, Wen G, Zhu Y, Yu H, Gan J (2020) Unsupervised feature selection by self-paced learning regularization. Pattern Recognit Lett 132:4–11
33. Tang Y, Xie Y, Yang X, Niu J, Zhang W (2021) Tensor multielastic kernel self-paced learning for time series clustering. IEEE Trans Knowl Data Eng 33(3):1223–1237
34. Chen R, Tang Y, Tian L, Zhang C, Zhang W (2021) Deep convolutional self-paced clustering. Appl Intell 52:4858–4872
35. Huang W, Liang C, Yu Y, Wang Z, Ruan W, Hu R (2018) Self-paced multi-task learning. In: Proceedings of the thirty-second AAAI conference on artificial intelligence, pp 2273–2280
36. Zhou S, Wang J, Meng D, Xin X, Li Y, Gong Y, Zheng N (2018) Deep self-paced learning for person re-identification. Pattern Recognit 76:739–751
37. Pan SJ, Tsang IW, Kwok JT, Yang Q (2011) Domain adaptation via transfer component analysis. IEEE Trans Neural Netw 22(2):199–210
38. Liang J, He R, Sun Z, Tan T (2019) Aggregating randomized clustering promoting invariant projections for domain adaptation. IEEE Trans Pattern Anal Mach Intell 41(5):1027–1042
39. Saenko K, Kulis B, Fritz M, Darrell T (2010) Adapting visual category models to new fomains. In: Proceedings of the European conference on computer vision (ECCV), pp 213–226
40. Venkateswara H, Eusebio J, Chakraborty S, Panchanathan S (2017) Deep hashing network for unsupervised domain adaptation. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), pp 5018–2027
41. Peng X, Usman B, Kaushik N, Hoffman J, Wang D, Saenko K (2017) Visda: the visual domain adaptation challenge. arXiv preprint arXiv:1710.06924
42. Wang Q, Breckon TP (2020) Unsupervised domain adaptation via structured prediction based selective pseudo-labeling. In: The thirty-fourth AAAI conference on artificial intelligence (AAAI), pp 6243–6250

43. He K, Zhang X, Ren S, Sun J (2017) Deep residual learning for image 1084 recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), pp 770–778
44. Griffin G, Holub A, Perona P (2007) Caltech-256 object category dataset
45. Luo Y, Ren C, Dai D, Yan H (2022) Unsupervised domain adaptation via discriminative manifold propagation. IEEE Trans Pattern Anal Mach Intell 44:1653–1669
46. Chen Z, Chen C, Cheng Z, Jiang B, Fang K, Jin X (2020) Selective transfer with reinforced transfer network for partial domain adaptation. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), pp 12706–12714