Hindawi
Computational Intelligence and Neuroscience
Volume 2022, Article ID 1582624, 18 pages
https://doi.org/10.1155/2022/1582624

Research Article

TSTELM: Two-Stage Transfer Extreme Learning Machine for Unsupervised Domain Adaptation

Shaofei Zang,[1] **Xinghai Li**,[1] **Jianwei Ma**,[1] **Yongyi Yan**,[1] **Jiwei Gao**,[1] **and Yuan Wei**[2]

[1]*College of Information Engineering, Henan University of Science and Technology, Luoyang 471000, China*
[2]*College of Vehicle and Traffic Engineering, Henan University of Science and Technology, Luoyang 471000, China*

Correspondence should be addressed to Jianwei Ma; majianwei_haust@163.com

Received 18 January 2022; Revised 21 June 2022; Accepted 23 June 2022; Published 18 July 2022

Academic Editor: Friedhelm Schwenker

As a single-layer feedforward network (SLFN), extreme learning machine (ELM) has been successfully applied for classification and regression in machine learning due to its faster training speed and better generalization. However, it will perform poorly for domain adaptation in which the distributions between training data and testing data are inconsistent. In this article, we propose a novel ELM called two-stage transfer extreme learning machine (TSTELM) to solve this problem. At the statistical matching stage, we adopt maximum mean discrepancy (MMD) to narrow the distribution difference of the output layer between domains. In addition, at the subspace alignment stage, we align the source and target model parameters, design target cross-domain mean approximation, and add the output weight approximation to further promote the knowledge transferring across domains. Moreover, the prediction of test sample is jointly determined by the ELM parameters generated at the two stages. Finally, we investigate the proposed approach in classification task and conduct experiments on four public domain adaptation datasets. The result indicates that TSTELM could effectively enhance the knowledge transfer ability of ELM with higher accuracy than other existing transfer and non-transfer classifiers.

1. Introduction

In the current era of big data, the classification model constructed by machine learning can help human quickly identify and annotate a large number of images, texts, audios, and signal data rapidly generated by the Internet, sensors, and computers. Mining information from these data helps understand the relationship between things. Support vector machine (SVM) [1], k-nearest neighbor (kNN) [2], naive Bayes [3, 4], decision tree [5], logistic regression [6], and many other classifiers with high accuracy appear and have attracted much attention. Huang et al. [7] proposed extreme learning machine (ELM) which is a better classifier with powerful nonlinear fitting and approximation capabilities [8, 9] and has been widely studied and applied in brain-computer interfaces [10, 11], medical diagnosis [12, 13], fault diagnosis [14], hyperspectral [15], and other fields.

ELM initializes randomly the input weight and bias and obtains optimal output weight by solving a least-squares problem [7–9, 16], which has the advantages of faster learning speed and better generalization, therefore becomes a hot research topic. There are many variants of ELM put forward both in theories and applications to enhance its performance for handling problems in different situations. In order to solve the problem that ELM is sensitive to the input weights and biases, Li et al. [17] proposed a WOA-ELM algorithm which applied the whale optimization algorithm (WOA) to optimize the input weights and biases of ELM for its performance improvement. In response to the class imbalance problem, weighted extreme learning machine (WELM) [18–20] was proposed, in which different weights are assigned for each training sample based on two different strategies. SMOTE based on class-specific extreme learning machine (SMOTE-CSELM) [21] was also presented by exploiting the benefit of both the minority oversampling and the class-specific regularization. To improve the generalization power and prevent the overtraining of ELM,

some methods combined ensemble learning with it to improve its robustness, including voting-based extreme learning machine (V-ELM) [22, 23], AdaBoost extreme learning machine [24–26], and extreme ensemble of ELMs (EEoELMs) [27]. Moreover, affected by deep learning, ELMs with deep structure occur for high accuracy. ML-ELM [28, 29] was presented to resolve the time-consuming issue of deep learning and achieved faster speed and higher generalization than stacked autoencoders, deep belief network, and deep Boltzmann machine. Hierarchical ELM (H-ELM) [30, 31] was proposed to enhance the universal approximation capability of ELM. The kernel-based multilayer ELM (ML-KELM) [32] integrated the kernel learning technique into the ML-ELM and achieved a faster learning speed and a better recognition performance. Although the above ELM models have achieved great success in classification and regression tasks, they will degrade when training samples and test samples are taken from different domains with different distributions (i.e., cross-domain tasks).

To handle this problem, domain adaptation (DA) [33–35], as an important branch of transfer learning, has attracted wide attention, in which efficient classifier is obtained with the help of the knowledge from source domain, which is different but related to target domain. L. Zhang and D. Zhang [36] put forward the domain adaptation extreme learning machine (DAELM) framework by extending ELM to handle domain adaptation problems for gas identification and drift compensation of E-nose system. Adaptive ELM (AELM) [37] was proposed by introducing the manifold regularization term into ELM for image classification. Zang et al. [38] proposed a supervised extreme learning machine called transfer extreme learning machine with output weight alignment (TELM-OWA), which aligned the output weight matrix of the ELM between domains and added the approximation between the inter-domain ELM parameters for knowledge transferring. However, these approaches are developed to solve semi-supervised domain adaptation problems because they require few labeled samples from the target domains. Due to its high cost in collecting labels and labeling samples, cross-domain ELM (CDELM) [39], domain space transfer ELM (DST-ELM) [40], cross-domain extreme learning machine (CdELM) [41], and extreme learning machine based on maximum weighted mean discrepancy (ELM-MWMD) [42] are proposed respectively for unsupervised domains by minimizing the classification loss and applying the maximum mean discrepancy (MMD) strategy on the prediction results. In the above methods, the supervised ELM model usually outperforms the unsupervised ones with the help of a few labeled samples from the target domain.

In this article, inspired by pioneering works [38, 42], we propose a novel method denoted as two-stage transfer extreme learning machine (TSTELM), in which there are two stages of domain adaptation: statistical matching and subspace alignment. At the statistical matching stage, we first learn a domain adaptation ELM classifier via utilizing the MMD to simultaneously minimize the marginal and conditional distribution between domains. In addition, at the subspace alignment stage, we use a transformation matrix to align the output weights of inter-domain ELM models, simultaneously put forward target cross-domain mean approximation, and add an output weight approximation term into the objective function. Then we can obtain the other domain adaptation ELM. Finally, we fuse the DAELM parameters from two stages to realize the label prediction of test samples. TSTELM is illustrated in Figure 1. Extensive experiments have been conducted on real-world image and text datasets, and the results verify that our approach outperforms several existing domain adaptation and non-domain adaptation methods.

In this article, TSTELM realizes knowledge transferring at two stages, and its contributions are summarized as follows:

(1) Similar to [39], our method is to use MMD proved as a general statistical distribution discrepancy measure, to minimize the marginal and conditional distribution discrepancy of the outputs of hidden lays of ELMs from two domains, which effectively extends ELM for unsupervised domain adaptation. Therefore, we can obtain one DAELM.

(2) Based on the first DAELM and inspired by [42], we introduce the output weight alignment, design target cross-domain mean approximation, and add the output weight approximation constraint into traditional ELM for enhancing knowledge transfer across domains. Hence, we can learn the other DAELM. It is worth emphasizing that we present target cross-domain mean approximation referred to [35] to adapt the distribution of the target domain for consistency with the source domains.

(3) At prediction stage, the above two DAELMs jointly determined the category of test samples. Our proposed method performs image classification experiments on object recognition and text datasets. The results verify its effectiveness and advantages.

(4) Compared with other the state-of-the-art DAELMs, our research has some distinct properties: (1) Many technologies including MMD, output weight alignment, output weight approximation, and target cross-domain mean approximation are utilized to jointly realize the efficient knowledge transfer across domains at two stages. (2) Output weight alignment organically bridges the DAELMs from the statistical matching stage and subspace alignment stage. (3) Joint decision from two DAELMs facilitates our approach to obtain robustness and high accuracy.

The rest of this article is organized as follows: In Section 2, we briefly review domain adaptation and ELM. We then present the proposed TSTELM in Section 3. In Section 4, the experiment and analysis are presented. Finally, Section 5 is the conclusion of this article.

2. A Brief Review of the Domain Adaptation and Extreme Learning Machine

2.1. Domain Adaptation. When the training and testing data are drawn from different distributions, a classifier directly

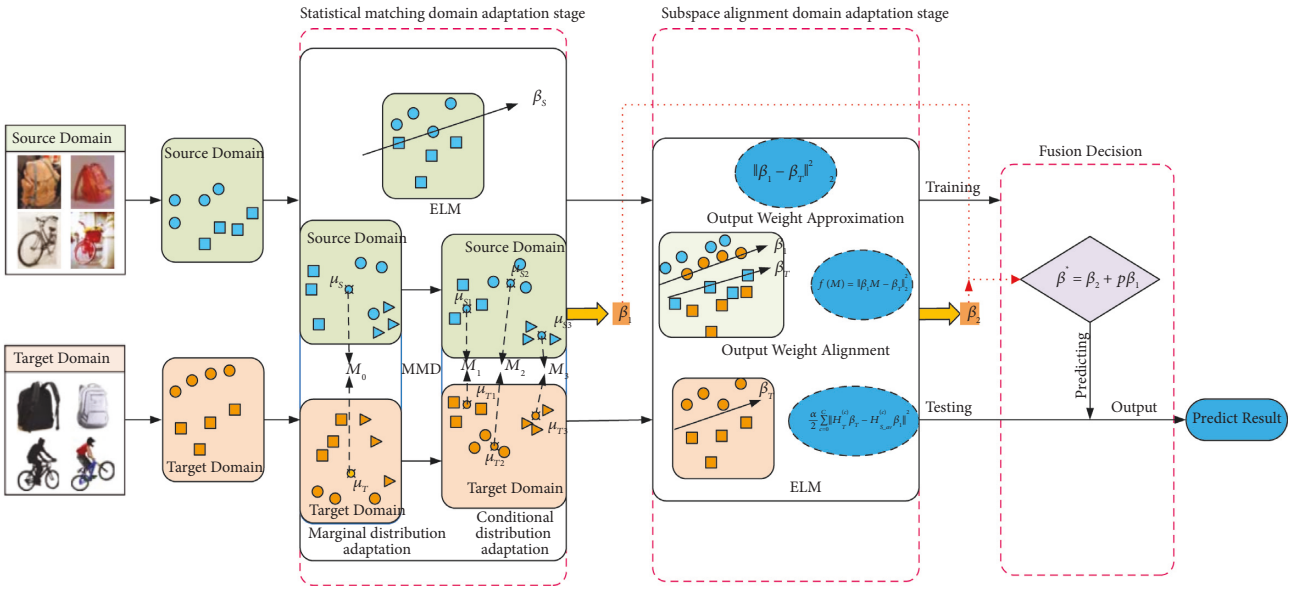

FIGURE 1: An illustration of TSTELM. (1) At the statistical matching domain adaptation stage, we use the MMD to simultaneously minimize the marginal and conditional distribution between domains in output layers and get a DAELM. (2) At the subspace alignment domain adaptation stage, we align the output weights of source and target ELM model, simultaneously design target cross-domain mean approximation term, and add the output weight approximation term into the objective function. We can obtain the other DAELM. (3) In fusion decision, we fuse the DAELM parameters from two stages to realize the label prediction of test samples.

learned on training data would have a poor performance for testing data. Domain adaptation is developed to deal with this scenario, in which an excellent classifier could be obtained from the source domain with rich labeled samples and perform well in the target domain task. The source domain and the target domain have different distributions but some correlations.

In the past decades, many researches have conducted to address domain adaptation problems in classification task, which are mainly divided into three parts [35, 43]: (1) Sample-based adaptation. It directly assigns weights to each sample of two domains, which could adapt and minimize the distribution gap between domains. Many such approaches appeared such as domain adaptation (PRDA) [44], TrAdaBoost [45], and Kernel Mean Match (KMM) [46]. (2) Feature-based adaption. It seeks the shared subspace between domains, in which distribution discrepancy is alleviated and knowledge is easily transferred across domains. Transfer component analysis (TCA) [47] and joint distribution adaptation (JDA) [48] take MMD metric as an objective function to find an optimal projected matrix for shared low-dimensional subspace. Liang et al. [49] designed a relaxed domain-irrelevant class clustering (DICE) term and then combined it with MMD to obtain a domain-irrelevant projection for reducing distribution discrepancy between domains. Moreover, DICE was extended to ensemble learning with multiple projection obtained from sampling subsets of source and target domains, which help it achieve better performance. Progressive learning with Confidence-wEighted Targets (PACET) [50] improved DICE by adding a confidence-weight strategy with the posterior probability of target instance. (3) Classifier (or parameter-based) adaptation. Its purpose is to find optimal classifier or its parameter with a well-generalized ability between the source and target domains. Yang et al. [51] presented adaptive support vector machine (Adapt-SVM). It designed a regularizer to minimize the discrepancy between parameters of two classifiers trained on source and target labeled samples and then added into SVM's objective function. Multi-model knowledge transfer (Multi-KT) [52], following the idea of Adapt-SVM, constructs a regularizer to force the parameter of target SVM close to multiple weighted source SVMs. In multiple kernel learning, Wang et al. [53] introduced multiple kernel MMD into the objective function to adapt distribution discrepancy between training samples and test samples, which prevents performance degradation because of inconsistent distribution of datasets and simultaneously obtain a multiple kernel classifier with strong generalization ability. Recently, deep network adaptation and adversarial learning adaptation have become successful in computer version and machine learning. Based on the assumption that samples with the same category are close each to other and the local geometry property of the data can be maintained in neural embedding subspace, Wang et al. [54] proposed the neural embedding match (NEM), which reduces cross-domain distribution divergence by projecting the source and target domains into a common subspace using deep neural network embedding model. In [55], a deep neural network with weighted MMD and the manifold embedding was proposed to handle domain adaptation for hyperspectral image classification. To address the problem in unsupervised partial domain adaptation (PDA), Liang et al. [56] put forward a domain adversarial neural networks called BA^{3}US. It presented balanced adversarial alignment (BAA) and adaptive uncertainty suppression (AUS) to overcome negative transfer and propagation of uncertainty which usually appear in PDA.

In the abovementioned approaches, sample-based adaptation methods are the most efficient ones for knowledge transfer because of direct utilization of source sample, while feature-based adaptation methods are widely applied. Classifier (or parameter)-based adaptation is the most potential one due to past related domain knowledge or experience is integrated into shared parameters of classifier. Deep network adaptation and adversarial learning adaptation strictly belong to feature-based adaptation method but they can extract (deep) domain-invariant feature with strong discrimination. However, these methods also have their own shortcomings. In sample-based adaptation methods, the effective evaluation mechanism about sample importance is a challenge. Generic shared features obtained from different domains is also a difficult task in feature-based adaptation methods. Since the useful information and knowledge from the auxiliary domain to the target domain is not directly applied, classifier (or parameter-based) adaptation is not efficient compared with two formers. Deep network adaptation usually needs massive labeled samples and sufficient computing resources for training deep model, which could hinder its application. Class misalignment and the simultaneous efficiency of feature extractor and discriminator are a challenge for adversarial learning adaptation. In this article, our approach belongs to classifier (or parameter)-based adaptation; it attempts to seek two output weights of shared ELM models across domains for knowledge transferring.

In this article, we propose TSTELM to address problems in the unsupervised domain adaptation, in which the training data come from the source domain with labeled samples and the test data come from the target domain with unlabeled samples. Suppose the source domain dataset is denoted as $\{(\mathbf{x}_{Si}, \mathbf{y}_i)\}_{i=1}^{n_S} \in \mathbf{D}_S$ and the target domain dataset is denoted as $\{(\mathbf{x}_{Tj})\}_{j=1}^{n_T} \in \mathbf{D}_T$, where n_S and n_T represent the number of source and target samples, respectively. The source data and the target data belong to the same feature space $\mathcal{X}_S = \mathcal{X}_T$ and label space $\mathcal{Y}_S = \mathcal{Y}_T$. The data distributions of the source and the target domains should be different but similar, that is the marginal distribution $P(\mathbf{X}_S) \approx P(\mathbf{X}_T)$ and conditional distribution $P(\mathbf{Y}_S|\mathbf{X}_S) \approx P(\mathbf{Y}_T|\mathbf{X}_T)$. In TSTELM, we hope to construct an ELM model using $\{(\mathbf{x}_{s(i)}, \mathbf{y}_{s(i)})\}_{i=1}^{n_S}$ to obtain high accuracy on $\{\mathbf{x}_{Te(k)}\}_{k=1}^{n_{Te}}$.Table 1 summarizes other related notations in domain adaptation problems.

2.2. Extreme Learning Machine. Unlike the conventional feedforward neural networks, ELM is an approach in which two characteristics are contained: (1) Hidden layer parameters (i.e., input weights and the biases) can be randomly initialized. (2) The output layer weight can be solved as the least-squares problem. As a result, it yields faster learning speed and better generalization performance compared with other classifiers.

Given a training set $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{N}$ with N samples, where $\mathbf{y}_i$ is the label corresponding to $\mathbf{x}_i$, and $\mathbf{C}$ is the number of categories. The structure of the ELM with L hidden nodes and activation function $h(x)$ is shown in Figure 2:

Table 1: Notations.

Terminology	Source	Target
Domain	$\mathbf{D}_S = \{\mathcal{X}_S, P(\mathbf{X}_S)\}$	$\mathbf{D}_T = \{\mathcal{X}_T, P(\mathbf{X}_T)\}$
Data	$\mathbf{X}_S = \{(\mathbf{x}_{Si}, \mathbf{y}_i)\}_{i=1}^{n_S}$	$\mathbf{X}_T = \{(\mathbf{x}_{Tj})\}_{j=1}^{n_T}$
Feature space	$\mathcal{X}_S$	$\mathcal{X}_T$
Label space	$\mathcal{Y}_S$	$\mathcal{Y}_T$
Marginal distribution	$P_S(\mathbf{X}_S)$	$P_T(\mathbf{X}_T)$
Conditional distribution	$P(\mathbf{Y}_S\|\mathbf{X}_S)$	$P(\mathbf{Y}_T\|\mathbf{X}_T)$
Task	$\mathbf{T}_S = \{\mathcal{Y}_S, P(\mathbf{Y}_S\|\mathbf{X}_S)\}$	$\mathbf{T}_T = \{\mathcal{Y}_T, P(\mathbf{Y}_T\|\mathbf{X}_T)\}$

In Figure 2, x_i is the input sample, $\mathbf{w}$ is the input layer weight, $\mathbf{b}$ is the hidden layer bias, and the hidden layer output $h(x)$ is calculated as: $h(x_i) = g(\mathbf{w}x_i + \mathbf{b})$. Here, $g(x)$ is the nonlinear activation function, L is the number of nodes in the hidden layer, and β_i is the hidden layer output weight. The outputs of the network are given by:

$$\mathbf{y}_j = \sum_{i=1}^{L} \beta_i h\left(\mathbf{w}_i \cdot \mathbf{x}_j + b_i\right), j = 1, 2, \ldots, N. \tag{1}$$

The above formula can be written in matrix form:

$$\mathbf{Y} = \mathbf{H}\boldsymbol{\beta}, \tag{2}$$

where $\mathbf{H} = \begin{bmatrix} h(\mathbf{w}_1 \cdot \mathbf{x}_1 + b_1) & \cdots & h(\mathbf{w}_L \cdot \mathbf{x}_1 + b_L) \\ \vdots & \ddots & \vdots \\ h(\mathbf{w}_1 \cdot \mathbf{x}_N + b_1) & \cdots & h(\mathbf{w}_L \cdot \mathbf{x}_N + b_L) \end{bmatrix}_{N \times L}$, $\beta = \begin{bmatrix} \beta_1 \\ \vdots \\ \beta_L \end{bmatrix}$, and $\mathbf{Y} = \begin{bmatrix} \mathbf{y}_1 \\ \vdots \\ \mathbf{y}_N \end{bmatrix}$.

By adopting parameter regularization, the ELM can avoid the overfitting problem. Its corresponding objective function can be formulated as

$$\min_{\boldsymbol{\beta}} L_{\text{ELM}} = \frac{1}{2}\|\boldsymbol{\beta}\|^2 + \frac{\theta}{2}\|\mathbf{Y} - \mathbf{H}\boldsymbol{\beta}\|^2, \tag{3}$$

where θ is a penalty constant on the training errors, and $\|\bullet\|^2$ denotes the L2-norm of a matrix or a vector.

The minimization of equation (3) is a regularized least-squares problem. By setting the gradient of equation (3) with respect to β as zero, we have

$$\nabla L_{\text{ELM}} = \boldsymbol{\beta} + \theta\mathbf{H}^T(\mathbf{Y} - \mathbf{H}\boldsymbol{\beta}) = 0. \tag{4}$$

The output weight vector β is obtained according to the Moore–Penrose principle. If $N > L$, the optimal solution of equation (3) is:

$$\boldsymbol{\beta}^* = \left(\mathbf{H}^T\mathbf{H} + \frac{\mathbf{I}_L}{\theta}\right)\mathbf{H}^T\mathbf{Y}, \tag{5}$$

where $\mathbf{I}_L$ is a L-dimensional unit matrix.

If $N \leq L$, the optimal solution of equation (3) is:

$$\boldsymbol{\beta}^* = \mathbf{H}^T\left(\mathbf{H}^T\mathbf{H} + \frac{\mathbf{I}_N}{\theta}\right)\mathbf{Y}, \tag{6}$$

where $\mathbf{I}_N$ is an N-dimensional unit matrix.

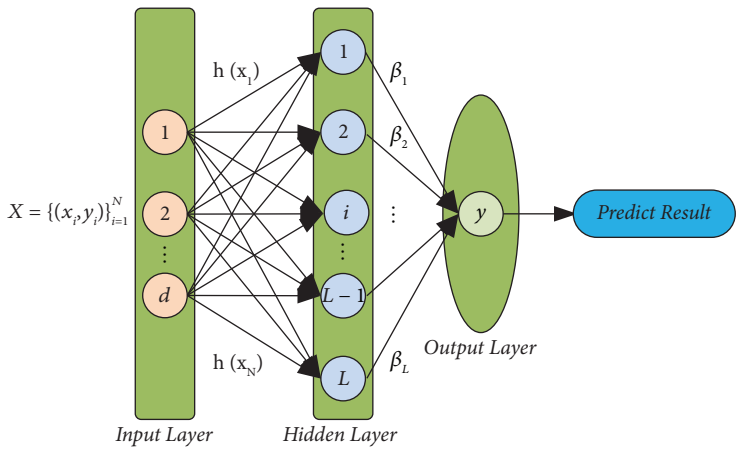

Figure 2: The structure of ELM.

3. Proposed Methods

In this section, the overall architecture of the proposed TSTELM is introduced in detail. As shown in Figure 1, TSTELM consists of three parts: statistical matching stage, subspace alignment stage, and prediction based on weight fusion.

3.1. Statistical Matching Stage. In statistical matching stage, we hope to obtain an ELM for domain adaptation using labeled samples of the source domain and unlabeled samples of the target domain. First, the source data and target data are mapped into the hidden layer space of ELM, and then we could obtain $\mathbf{H}_S = \{(\mathbf{h}_{Si}, \mathbf{y}_i)\}_{i=1}^{n_S}$ and $\mathbf{H}_T = \{(\mathbf{h}_{Tj})\}_{j=1}^{n_T}$, respectively, where $h_S(x_i) = g(\mathbf{w}x_{Si} + \mathbf{b})$ and $h_T(x_j) = g(\mathbf{w}x_{Tj} + \mathbf{b})$, $n = n_S + n_T$, $\mathbf{w} \in R^{d\times L}$ and $\mathbf{b} \in R^{1\times L}$ are randomly generated weights and bias, d is the original spatial dimension of the data.

For labeled source data, we can learn an ELM, that is

$$\min_{\boldsymbol{\beta}_S} : \frac{1}{2}\left\|\boldsymbol{\beta}_S\right\|^2 + \frac{1}{2}\left\|\mathbf{H}_S\boldsymbol{\beta}_S - \mathbf{Y}_S\right\|^2, \tag{7}$$

where β_S is the output weight of the ELM learned on $\{\mathbf{H}_S, \mathbf{Y}_S\}$. Since equation (7) just obtain an ELM classifier using the labeled source samples, it cannot perform well for the target domains due to distribution difference between the source and target domains. Therefore, we adopt MMD between $\mathbf{H}_S$ and $\mathbf{H}_T$ to reduce marginal and conditional distribution difference between domains [42]. MMD minimization is formulated as:

$$\begin{aligned}\mathrm{MMD}(\boldsymbol{\beta}) &= \min_{\boldsymbol{\beta}} \left\| \frac{1}{n_S}\sum_{i=1}^{n_S}\mathbf{h}(\mathbf{x}_i)\boldsymbol{\beta} - \frac{1}{n_T}\sum_{j=1}^{n_T}\mathbf{h}(\mathbf{x}_j)\boldsymbol{\beta} \right\|_{\mathscr{H}}^2 \\ &+ \sum_{c=1}^{C}\left\| \frac{1}{n_S^{(c)}}\sum_{\mathbf{x}_i\in\mathscr{D}_S^{(c)}}\mathbf{h}(\mathbf{x}_i^{(c)})\boldsymbol{\beta} - \frac{1}{n_T^{(c)}}\sum_{\mathbf{x}_j\in\mathscr{D}_T^{(c)}}\mathbf{h}(\mathbf{x}_j^{(c)})\boldsymbol{\beta} \right\|_{\mathscr{H}}^2 \\ &= \min_{\boldsymbol{\beta}} \mathrm{tr}\left(\boldsymbol{\beta}^T\mathbf{H}^T\left(\mathbf{M}_0 + \sum_{c=1}^{C}\mathbf{M}_c\right)\mathbf{H}\boldsymbol{\beta}\right),\end{aligned} \tag{8}$$

where $\mathbf{H} = \mathbf{H}_S^T \cup \mathbf{H}_T^T$; $\mathbf{M}_0$ and $\mathbf{M}_c$ are the MMD matrixes which are as follows:

$$(\mathbf{M}_0)_{ij} = \begin{cases} \dfrac{1}{n_S n_S}, & \mathbf{x}_i, \mathbf{x}_j \in \mathscr{D}_S, \\ \dfrac{1}{n_T n_T}, & \mathbf{x}_i, \mathbf{x}_j \in \mathscr{D}_T, \\ \dfrac{-1}{n_S n_T}, & \text{otherwise}, \end{cases} \tag{9}$$

$$(\mathbf{M}_c)_{ij} = \begin{cases} \dfrac{1}{n_S^{(c)} n_S^{(c)}}, & \mathbf{x}_i^{(c)}, \mathbf{x}_j^{(c)} \in \mathscr{D}_S^{(c)}, \\ \dfrac{1}{n_T^{(c)} n_T^{(c)}}, & \mathbf{x}_i^{(c)}, \mathbf{x}_j^{(c)} \in \mathscr{D}_T^{(c)}, \\ \dfrac{-1}{n_S^{(c)} n_T^{(c)}}, & \begin{cases} \mathbf{x}_i^{(c)} \in \mathscr{D}_S^{(c)} \text{and} \mathbf{x}_j^{(c)} \in \mathscr{D}_T^{(c)}, \\ \mathbf{x}_j^{(c)} \in \mathscr{D}_S^{(c)} \text{and} \mathbf{x}_i^{(c)} \in \mathscr{D}_T^{(c)}, \end{cases} \\ 0, & \text{otherwise}, \end{cases} \tag{10}$$

where $\mathbf{D}_S^{(c)}$ and $\mathbf{D}_T^{(c)}$ are the sample subsets with label category c in the source and target domains; $\mathbf{x}_i^{(c)}$ and $\mathbf{x}_j^{(c)}$ are the samples in $\mathbf{D}_S^{(c)}$ and $\mathbf{D}_T^{(c)}$, respectively; $n_S^{(c)}$ and $n_T^{(c)}$ are the number of samples in $\mathbf{D}_S^{(c)}$ and $\mathbf{D}_T^{(c)}$, respectively.

Here, replacing β_S and β with β_1 and incorporating equations (7) and (8), we can obtain the DAELM at statistical matching stage, and its objective function is

$$\begin{aligned}&\min_{\boldsymbol{\beta}} \frac{1}{2}\left\|\boldsymbol{\beta}_1\right\|^2 + \frac{\theta}{2}\left\|\mathbf{H}_S\boldsymbol{\beta}_1 - \mathbf{Y}_S\right\|^2 \\ &+ \frac{\lambda}{2}\mathrm{Tr}\left(\boldsymbol{\beta}_1^T\mathbf{H}^T\left(\mathbf{M}_0 + \sum_{c=1}^{C}\mathbf{M}_c\right)\mathbf{H}\boldsymbol{\beta}_1\right).\end{aligned} \tag{11}$$

By setting the gradient of equation (11) with respect to β_1 as zero, we have

$$\boldsymbol{\beta}_1 = \begin{cases} \mathbf{H}^T\left(\mathbf{H}\mathbf{E}\mathbf{H}^T + \lambda\left(\mathbf{M}_0 + \sum_{c=1}^{C}\mathbf{M}_c\right)\mathbf{H}\mathbf{H}^{\mathrm{T}} + \frac{\mathbf{I}_L}{\theta}\right)^{-1}\mathbf{E}\mathbf{Y}_S^T, & n > L, \\ \left(\mathbf{H}^T\mathbf{E}\mathbf{H} + \lambda\mathbf{H}^T\left(\mathbf{M}_0 + \sum_{c=1}^{C}\mathbf{M}_c\right)\mathbf{H} + \frac{\mathbf{I}_n}{\theta}\right)^{-1}\mathbf{H}^T\mathbf{E}\mathbf{Y}_S^T, & n \le L, \end{cases} \tag{12}$$

where $\mathbf{E}$ is a diagonal label indicator matrix with each element $E_{ii} = 1$ if $\mathbf{x}_i \in \mathbf{D}_S$, and $E_{ii} = 0$ otherwise.

3.2. Subspace Alignment Stage. At subspace alignment stage, we train a DAELM on labeled samples of the source domains and unlabeled samples of the target domains.

For the target sample, we can learn an ELM from the following formula:

$$\begin{aligned} &\min_{\boldsymbol{\beta}_T} : \frac{1}{2}\left\|\boldsymbol{\beta}_T\right\|^2 + \frac{\alpha}{2}\left\|\mathbf{H}_T\boldsymbol{\beta}_T - \mathbf{H}_{S\_av}\boldsymbol{\beta}_1\right\|^2 \\ &+ \frac{\alpha}{2}\sum_{c=1}^{C}\left\|\mathbf{H}_T^{(c)}\boldsymbol{\beta}_T - \mathbf{H}_{S\_av}^{(c)}\boldsymbol{\beta}_1\right\|^2, \\ &\Leftrightarrow \min_{\boldsymbol{\beta}_T} : \frac{1}{2}\left\|\boldsymbol{\beta}_T\right\|^2 + \frac{\alpha}{2}\sum_{c=0}^{C}\left\|\mathbf{H}_T^{(c)}\boldsymbol{\beta}_T - \mathbf{H}_{S\_av}^{(c)}\boldsymbol{\beta}_1\right\|^2. \end{aligned} \tag{13}$$

Here, inspired by [35], we introduce cross-domain mean approximation to replace the prediction loss. When there are no labeled samples in the target domains (when $c = 0$), we force target data $\mathbf{H}_T$ close to source data mean point $\mathbf{H}_{S\_av}$, which promotes domain adaptation seen from [35]. If the target sample obtains pseudo labels, it is drawn to source data mean point with the same category $\mathbf{H}_{S\_av}^{(c)}$.

In order to further improve cross-domain knowledge transferring, similar to [38], we introduce a transformation matrix $\mathbf{M}$ to align the output weights of ELM between the source domain and the target domain. The function is established as follows:

$$f(\mathbf{M}) = \min\left\|\boldsymbol{\beta}_1\mathbf{M} - \boldsymbol{\beta}_T\right\|_F^2, \tag{14}$$

where $\|\bullet\|_F^2$ is Frobenius norm. It is invariant to the orthogonalization operation, so equation (14) can be rewritten as:

$$\begin{aligned} f(\mathbf{M}) &= \min\left\|\boldsymbol{\beta}_1^T\boldsymbol{\beta}_1 M - \boldsymbol{\beta}_1^T\boldsymbol{\beta}_T\right\|_F^2 \\ &= \min\left\|\mathbf{M} - \boldsymbol{\beta}_1^T\boldsymbol{\beta}_T\right\|_F^2. \end{aligned} \tag{15}$$

Then, we can get the optimal $\mathbf{M}* = \beta_1^T\beta_T$. Let $\beta_a = \beta_1\mathbf{M} = \beta_1\beta_1^T\beta_T$, we can know that β_a is closer to β_T than β_1 and facilitates cross-domain knowledge transfer.

To align output layer of source ELM to target one, we combine the training error $\|\mathbf{H}_S\beta_a - \mathbf{Y}_S\|^2$, equation (13), and a regular term and replace β_1 with β_a to get:

$$\begin{aligned} J(\boldsymbol{\beta}_a, \boldsymbol{\beta}_T) = \min_{\boldsymbol{\beta}_a, \boldsymbol{\beta}_T} &\frac{1}{2}\left\|\boldsymbol{\beta}_T\right\|^2 + \frac{\theta}{2}\left\|\mathbf{H}_S\boldsymbol{\beta}_a - \mathbf{Y}_S\right\|^2 \\ &+ \frac{\alpha}{2}\sum_{c=0}^{C}\left\|\mathbf{H}_T^{(c)}\boldsymbol{\beta}_T - \mathbf{H}_{S\_av}^{(c)}\boldsymbol{\beta}_1\right\|^2 + \frac{1}{2}\left\|\boldsymbol{\beta}_T - \boldsymbol{\beta}_a\right\|^2, \end{aligned} \tag{16}$$

where $\|\beta_T - \beta_a\|^2$ is a parameter approximation term for facilitating knowledge transfer and preventing negative transfer, and λ and γ are the balance parametesr. We substitute $\beta_a = \beta_1\beta_1^{\mathrm{T}}\beta_T$ to equation (16) and get:

$$\begin{aligned} J(\boldsymbol{\beta}_T) &= \min_{\boldsymbol{\beta}_T} \frac{1}{2}\left\|\boldsymbol{\beta}_T\right\|^2 + \frac{\theta}{2}\left\|\mathbf{H}_S\boldsymbol{\beta}_1\boldsymbol{\beta}_1^{\mathrm{T}}\boldsymbol{\beta}_T - \mathbf{Y}_S\right\|^2 \\ &\quad + \frac{\alpha}{2}\sum_{c=0}^{C}\left\|\mathbf{H}_T^{(c)}\boldsymbol{\beta}_T - \mathbf{H}_{S\_av}^{(c)}\boldsymbol{\beta}_1\right\|^2 + \frac{1}{2}\left\|\boldsymbol{\beta}_T - \boldsymbol{\beta}_1\boldsymbol{\beta}_1^{\mathrm{T}}\boldsymbol{\beta}_T\right\|^2 \\ &= \min_{\boldsymbol{\beta}_T} \frac{1}{2}\left\|\boldsymbol{\beta}_T\right\|^2 + \frac{\theta}{2}\left\|\mathbf{H}_S\boldsymbol{\beta}_1\boldsymbol{\beta}_1^{\mathrm{T}}\boldsymbol{\beta}_T - \mathbf{Y}_S\right\|^2 \\ &\quad + \frac{\alpha}{2}\sum_{c=0}^{C}\left\|\mathbf{H}_T^{(c)}\boldsymbol{\beta}_T - \mathbf{H}_{S\_av}^{(c)}\boldsymbol{\beta}_1\right\|^2 + \frac{1}{2}\left\|\left(\mathbf{I} - \boldsymbol{\beta}_1\boldsymbol{\beta}_1^{\mathrm{T}}\right)\boldsymbol{\beta}_T\right\|^2. \end{aligned} \tag{17}$$

Because $\|(\mathbf{I} - \beta_1\beta_1^{\mathrm{T}})\beta_T\|^2 \le \|(\mathbf{I} - \beta_1\beta_1^{\mathrm{T}})\|^2\|\beta_T\|^2$, we change equation (17) into:

$$\begin{aligned} &J(\boldsymbol{\beta}_T) \\ &= \min_{\boldsymbol{\beta}_T} \frac{1}{2}\left\|\boldsymbol{\beta}_T\right\|^2 + \frac{\theta}{2}\left\|\mathbf{H}_S\boldsymbol{\beta}_1\boldsymbol{\beta}_1^T\boldsymbol{\beta}_T - \mathbf{Y}_S\right\|^2 \\ &\quad + \frac{\alpha}{2}\sum_{c=0}^{C}\left\|\mathbf{H}_T^{(c)}\boldsymbol{\beta}_T - \mathbf{H}_{S\_av}^{(c)}\boldsymbol{\beta}_1\right\|^2 + \frac{1}{2}\left\|\left(\mathbf{I} - \boldsymbol{\beta}_1\boldsymbol{\beta}_1^T\right)\right\|^2\left\|\boldsymbol{\beta}_T\right\|^2 \\ &= \min_{\boldsymbol{\beta}_T} \frac{1}{2}\left\|\begin{pmatrix} \theta\mathbf{H}_S\boldsymbol{\beta}_1\boldsymbol{\beta}_1^T \\ \alpha\mathbf{H}_T \end{pmatrix}\boldsymbol{\beta}_T - \begin{pmatrix} \mathbf{Y}_S \\ \alpha\sum_{c=0}^{C}\mathbf{H}_{S\_av}^{(c)}\boldsymbol{\beta}_1 \end{pmatrix}\right\|^2 \\ &\quad + \frac{\left(\mathbf{I} + \left(\mathbf{I} - \boldsymbol{\beta}_1\boldsymbol{\beta}_1^T\right)^T\left(\mathbf{I} - \boldsymbol{\beta}_1\boldsymbol{\beta}_1^T\right)\right)}{2}\left\|\boldsymbol{\beta}_T\right\|^2. \end{aligned} \tag{18}$$

Let $\mathbf{Q} = \begin{pmatrix} \theta\mathbf{H}_S\beta_1\beta_1^{\mathrm{T}} \\ \alpha\mathbf{H}_T \end{pmatrix}$, $\mathbf{T} = \begin{pmatrix} \mathbf{Y}_S \\ \alpha\sum_{c=0}^{C}\mathbf{H}_{S\_av}^{(c)}\beta_1 \end{pmatrix}$, and $\mathbf{A} = \mathbf{I} + (\mathbf{I} - \beta_1\beta_1^{\mathrm{T}})^T(\mathbf{I} - \beta_1\beta_1^{\mathrm{T}})$, and equation (18) can be simplified as

$$J(\boldsymbol{\beta}_T) = \min_{\beta_T} : \frac{1}{2}\left\|\mathbf{Q}\boldsymbol{\beta}_T - \mathbf{T}\right\|^2 + \frac{\mathbf{A}}{2}\left\|\boldsymbol{\beta}_T\right\|^2. \tag{19}$$

Then:

Input: Source domain $\mathbf{D}_S$, source labels $\mathbf{Y}_S$, target domain $\mathbf{D}_T$, maximum iterations T.
(1) Randomly initialize the input weights $\mathbf{w}$ and biases $\mathbf{b}$ of the ELM network with L hidden nodes; set the trade-off parameters p, α, θ, and λ.
(2) Calculate the matrix $\mathbf{H}_S$ and $\mathbf{H}_T$, obtain $\mathbf{H}_0$ using equation (9).
(3) Compute the optimal weights β_1 by equation (12).
(4) Calculate the matrix $\mathbf{H}_{S\_av}$ and compute the optimal weights β_2 using equation (20).
(5) Construct the MMD matrixes $\mathbf{M}_0$, $\mathbf{M}_c$, and $\mathbf{H}_{S\_av}^{(c)}$.
(6) Compute the optimal weights $\beta^* = \beta_2 + p\beta_1$ and obtain the prediction $\widehat{\mathbf{Y}}_\mathbf{T}$ of $\mathbf{H}_T$ using equation (21).
(7) Repeat step 2–6 until the number of iterations reaches to T or $\widehat{\mathbf{Y}}_\mathbf{T}$ no change.
Output: The output weight β^* and the predicted output $\widehat{\mathbf{Y}}_\mathbf{T}$.

Algorithm 1: TSTELM.

$$\boldsymbol{\beta}_2 = \boldsymbol{\beta}_T^* = \begin{cases} \left(\mathbf{Q}^T\mathbf{Q} + \mathbf{A}\right)^{-1}\mathbf{Q}^T\mathbf{T}, & n > L, \mathbf{I} \text{ in} \mathbf{A} \text{ is an } L - \text{dimensional unit matrix}, \\ \mathbf{Q}^T\left(\mathbf{Q}\mathbf{Q}^T + \mathbf{A}\right)^{-1}\mathbf{T}, & n \le L, \mathbf{I} \text{ in} \mathbf{A} \text{ is an } n - \text{dimensional unit matrix}. \end{cases} \tag{20}$$

3.3. Prediction Based on Output Weight Fusion. In the classification task, a sample $\mathbf{x}_{Te}$ is tested. After β_1 and β_2 are obtained, the output weight of final ELM model is dominated by $\beta^* = \beta_2 + p\beta_1$, and the classification result of $\mathbf{x}_{Te}$ can be obtained:

$$\mathbf{y}_{Te} = \begin{cases} \text{sign}\left(\mathbf{h}_{Te}^T\boldsymbol{\beta}^*\right), & \text{for binary classification}, \\ \text{argmax}\left(\mathbf{h}_{Te}^T\boldsymbol{\beta}^*\right), & \text{for multi} - \text{class classification}, \end{cases} \tag{21}$$

where $\mathbf{h}_{Te} = g(\mathbf{x}_{Te})$ and p is the scale factor to balance β_1 and β_2.

TSTELM can be summarized in Algorithm 1.

3.4. Discussion. In order to solve the problem that the traditional ELM does not perform well in cross-domain tasks, we propose TSTELM and its objective function is equations (8) and (17). It can be seen:

(1) Compared with the classical ELM, TSTELM reduces the distribution difference between domains and transfers knowledge across domains via adopting MMD, output weight alignment, parameter approximation, and $\sum_{c=0}^{C} \|\mathbf{H}_T^{(c)}\beta_T - \mathbf{H}_{S\_av}^{(c)}\beta_1\|^2$.
(2) Though TELM-OWA proposed by Zang et al. [38] also applies output weight alignment and parameter approximation for domain adaptation, it is a supervised domain adaptation algorithm requiring few target labeled samples unlike TSTELM. In addition, TSTELM replaces $\|\mathbf{H}_T\beta_T - \mathbf{Y}_T\|^2$ with $\sum_{c=0}^{C} \|\mathbf{H}_T^{(c)}\beta_T - \mathbf{H}_{S\_av}^{(c)}\beta_1\|^2$, which is different from TELM-OWA.
(3) Different from DAELM-S, DAELM-T, TELM-OWA, and CDELM, in which only one output weight β^* is solved, our method needs the learned weights of two stages to fuse and then make decisions, which shows that TSTELM has strong robustness similar to ensemble learning.
(4) In domain adaptation, we consider samples from the source domains as training datasets and samples from the target domains as testing datasets, so the output matrixes $\mathbf{H}_S$ and $\mathbf{H}_T$ are computed which have time complexity of $O(Ln_Sd)$ and $O(Ln_Td)$. The main time cost of ELM is equation (5) or (6), and the time complexity is:

$$\begin{aligned} &\mathrm{O}\left(n_S^3 + 2Ln_S^2 + mLn_S\right)\text{when}, n_S < \mathrm{L}, \\ &\mathrm{O}\left(L^3 + L^2n_S + m\mathrm{Ln}_S\right)\text{when}, n_S > L. \end{aligned} \tag{22}$$

According to Algorithm 1, the main computation cost of our method is in steps 3 and 4.

In step 3, we need to compute $\mathbf{M}_0$, $\sum_{c=1}^{C}\mathbf{M}_c$, $(\mathbf{E} + \mathbf{M}_0 + \sum_{c=1}^{C}\mathbf{M}_c)\mathbf{H}\mathbf{H}^{\mathrm{T}}$, $\mathbf{H}^{\mathrm{T}}(\mathbf{E} + \mathbf{M}_0 + \sum_{c=1}^{C}\mathbf{M}_c)\mathbf{H}$, the inverse of matrix with $N \times N$ or $L \times L$ size and $\mathbf{H}^{\mathrm{T}}\mathbf{E}\mathbf{Y}_S$ which has time complexity $O(N^2)$, $O(CN^2)$, $O(N^3{+}LN^2)$, $O(LN^2 + L^2N)$, $O(N^3)$ or $O(L^3)$, and $O(\mathrm{CLN}{+}L\mathrm{N}^2)$. Therefore, the time complexity of step 2 is

$$\begin{aligned} &O\left(2\mathrm{TN}^3 + 2\mathrm{TLN}^2 + T(C+1)N^2 + \mathrm{TCLN}\right)\text{when}, N < \mathrm{L}, \\ &O\left(\mathrm{TL}^3 + 2\mathrm{TLN}^2 + \mathrm{TL}^2N + T(C+1)N^2 + \mathrm{TCLN}\right)\text{when}, N > \mathrm{L}, \end{aligned} \tag{23}$$

where T is the number of iterations.

In step 4, the output weight is determined according to equation (20) and $\mathbf{Q}$ has the same size of $\mathbf{H}$. Therefore, the time complexity of step 4 is:

$$\begin{aligned} &\mathrm{O}\left(TN^3{+}2\mathrm{TLN}^2 + \mathrm{TCLN}\right)\text{when}, N < \mathrm{L}, \\ &\mathrm{O}\left(TL^3 + TL^2N + \mathrm{TCLN}\right)\text{when}, N > L. \end{aligned} \tag{24}$$

Given that TELM-OWA also has two stages to compute the output weight, it has time complexity in the first stage as follows:

$$\begin{aligned} &\mathrm{O}\left(n_S^3 + 2Ln_S^2 + \mathrm{mLn}_S\right)\text{when}, n_S < \mathrm{L}, \\ &\mathrm{O}\left(L^3 + L^2 n_S + \mathrm{mLn}_S\right)\text{when}, n_S > L. \end{aligned} \tag{25}$$

The time complexity of TELM-OWA in the second stage is

$$\begin{aligned} &\mathrm{O}\left(N^3 + 2LN^2 + \mathrm{CLN}\right)\text{when}, N < \mathrm{L}, \\ &\mathrm{O}\left(L^3 + L^2 N + \mathrm{CLN}\right)\text{when}, N > L. \end{aligned} \tag{26}$$

The above analysis indicates that the computational complexity of TSTELM is significantly higher than ELM and TELM-OWA.

4. Experiment and Analysis

In this section, experiments are conducted on four cross-domain datasets including Office + Caltech object recognition, USPS and MNIST digital handwriting, MSRC and VOC2007 object recognition, and Reuters-21578 text dataset for classification, where image datasets are descripted in Table 2. We compare our approach with several related unsupervised classification methods and semi-supervised and unsupervised domain adaptation methods. To be more objective, experiments are implemented on PC with 8 GB memory and Windows 10 operating system and MATLAB 2017b. Every experiment runs 20 times and the average value is recorded. We adopt the accuracy rate to evaluate the performance of every algorithm, and it is

$$\text{Accuracy} = \frac{\text{correctly\_classified\_samples}}{\text{total\_samples}} \times 100\%. \tag{27}$$

4.1. Dataset Description. aOffice + Caltech256 (shown in Figure 3): Office is widely used dataset for visual cross-domain learning, which contains 4,652 images in 31 categories. These images come from 3 realistic aggregated item datasets: Amazon (images download from online chants https://www.amazon.com); DSLR (high-resolution images by a digital SLR camera in realistic environments); and Webcam (low-resolutions images by a simple webcam). Caltech256 is also a standard object recognition dataset which contains 30,607 images from 256 categories.

In this article, we employ the Office + Caltech dataset released by Gong et al. [57]. SURF features are extracted and quantized into an 800-bin histogram with codebooks computed with K-means on a subset of images from Amazon. Then, the histograms are standardized by z-score. We select four domains C (Caltech256), A (Amazon), W (Webcam), and D (DSLR) for experiment, and two different domains are randomly selected as the source and the target domain datasets, and 12 cross-domain tasks for evaluation are constructed, namely $\mathrm{C} \longrightarrow \mathrm{A}$, $\mathrm{C} \longrightarrow \mathrm{W}$, $\mathrm{C} \longrightarrow \mathrm{D}$, ..., and $\mathrm{D} \longrightarrow \mathrm{W}$.

USPS + MNIST (as shown in Figure 4): USPS and MNIST are the two different but related handwritten datasets with 10 categories of 0–9. The USPS dataset contains 7,291 training samples and 2,007 test samples with 16×16 pixels.

Table 2: Description of image dataset.

Dataset	Type	Samples	Dimension	Class	Contains subsets
USPS	Digit	1800	256	10	USPS
MNIST	Digit	2000	256	10	MNIST
MSRC	Object	1269	240	18	MSRC
VOC2007	Object	1530	240	20	VOC
Office	Object	1410	800	10	A, W, D
Caltech	Object	1123	800	10	C

There are 60,000 training images and 10,000 test images with 28×28 pixels in the MNIST database.

During this experiment, we randomly select 1,800 pictures from USPS and 2,000 pictures from MNIST and convert them into 16×16 pixels. We construct two cross-domain tasks, that is USPS as the source domain and MNIST as the target domain (USPS vs MNIST) and vice versa (MNIST vs USPS).

MSRC + VOC2007 (shown in Figure 5): MSRC dataset is provided by Microsoft Cambridge, which consists of 4323 images of 18 object classes. VOC2007 dataset contains 5,011 images annotated with 20 concepts. We can see from Figure 5 that MSRC and VOC2007 have distinct but different distributions. MSRC are standard images as benchmark data for evaluation. VOC2007 is randomly constructed by using the images in the network album.

In our experiments, we construct the domain adaptation dataset MSRC vs VOC in which 6 shared categories are selected including aircraft, birds, cows, family cars, sheep and bicycles. Among them, 1269 images are selected from the MSRC dataset as the source domain dataset and 1530 images are selected from the VOC2007 dataset as the target domain dataset. Then, the source domain and the target domain are exchanged to construct a new set of domain adaptation dataset VOC vs MSRC. We convert all images into 256 gray pixels; 240 dimensions are extracted as the spatial dimension of the sample.

Reuters-21578: Reuters-21578 text dataset is a common dataset for text classification. It contains 21,577 news documents from Reuters in 1987. These documents have been manually labeled by Reuters as five classes, such as "exchanges," "orgs," "people," "places," and "topics," including multiple categories and subclasses. Among them, the largest three categories are "orgs," "people," and "place," which can construct six cross-domain text classification tasks orgs vs people, people vs orgs, orgs vs place, place vs orgs, people vs place, and place vs people. This article makes a more complete evaluation of the algorithm on 6 classification tasks.

4.2. Experimental Settings. To validate the efficiency of TSTELM, we compare it with some other classifiers.

Classifiers for non-domain adaptation: 1NN, SVM, ELM, and SSELM [58] (ELM with graph regularization term for semi-supervised learning).

Classifiers for domain adaptation: TCA1 [47](TCA with 1NN for classification), TCA2 [47] (TCA with SVM for classification), JDA1 [48] (JDA with 1NN for classification),

(a) (b) (c) (d)

Figure 3: Image samples from (b) Amazon, (d) Caltech256, (a) DSLR, and (c) Webcam.

(a) (b)

Figure 4: Image samples from (a) MNIST and (b) USPS.

(a) (b)

Figure 5: Image samples from (a) MSRC and (b) VOC2007.

JDA2 [48](JDA with SVM for classification), DAELM-S [36], DAELM-T [36], ARRLS [59], TELM-OWA [38], JPDA [60], AELM [61], DST-ELM [40], CDELM [39], and TSTELM.

In order to achieve the optimal performance of each algorithm in the experiment, we set SVM penalty parameter belonging to $\{0.1, 0.5, 1, 5, 10, 50, 100\}$, and the penalty parameter $\theta \in [0.001, 0.1]$ in ELM, SSELM, DAELM_S, DAELM_T, TELM-OWA, and TSTELM. TCA(1,2) and JDA(1,2) are combined feature extraction with standard classifier for domain adaptation, in which the dimension of the feature subspace is 100 and the value range of the balanced-constraint parameter of the projection matrix in TCA and JDA is $[0.1, 1]$. The parameters of ARRLS, DAELM_S, DAELM_T, and TELM-OWA are set according to the corresponding literature. In TSTELM, we set $\alpha \in [10^3, 10^4]$, $\lambda \in [10^{-3}, 10^{-4}]$,and $L = 1500$ on Office + Caltech dataset, $L = 3000$ on USPS + MNIST and MSRC + VOC2007 datasets, $L = 5000$ on Reuters-21578 dataset. We cite results of JPDA, AELM, DST-ELM, and CDELM from those corresponding literature.

In fact, DAELM_S, DAELM_T, and TELM-OWA, as supervised models, need a few labeled target samples to induce the target classifier. We test them with 0.5% labeled target samples on USPS + MNIST, MSRC + VOC2007, and Reuters-21578 datasets and 1% labeled target samples on Office + Caltech dataset, to approximate the performance of these methods in unsupervised domain adaptation.

4.3. Experimental Results and Analysis. We test TSTELM on Office + Caltech256, USPS + MNIST, MSRC + VOC2007, and Reuters-21578 datasets, and the comparison results are displayed in Tables 3 and 4 and Figures 4–8, in which the

Table 3: Accuracy of different algorithms on Office + Caltech256 and USPS + MNIST datasets.

Methods	Dataset																
	C ⟶ A(1)	C ⟶ W(2)	C ⟶ D(3)	A ⟶ C(4)	A ⟶ W(5)	A ⟶ D (6)	W ⟶ C (7)	W ⟶ A (8)	W ⟶ D(9)	D ⟶ C(10)	D ⟶ A(11)	D ⟶ W(12)	Average	USPS vs MNIST	MNIST vs USPS	Average	Total average
1NN	23.70	25.76	25.48	26.00	29.83	25.48	19.86	22.96	59.24	26.27	28.50	63.39	31.37	44.70	65.94	55.32	34.79
SVM	52.40	40.68	42.04	44.08	41.69	40.76	29.92	34.13	**87.90**	31.43	32.36	73.56	45.91	35.10	54.69	44.9	45.77
ELM	50.63	42.71	42.04	43.01	37.63	42.68	32.24	36.85	80.25	30.10	31.52	71.86	45.13	30.70	51.39	41.05	44.54
SSELM	52.92	45.42	38.85	41.76	35.93	38.85	34.28	37.06	78.98	30.37	32.05	80.00	45.54	29.05	43.28	36.16	44.2
TCA1	55.85	48.81	**51.59**	44.17	40.34	36.94	32.32	38.41	81.53	36.24	39.35	82.37	48.99	35.55	53.34	44.45	48.34
TCA2	55.01	48.48	49.68	**44.35**	41.02	43.31	32.59	38.94	79.62	34.64	38.31	**83.89**	49.15	34.70	52.78	43.74	48.38
JDA1	53.34	46.78	52.87	42.65	36.27	40.76	33.66	47.60	86.89	36.51	43.31	82.73	47.33	34.77	54.92	44.85	49.5
JDA2	54.80	51.86	51.22	41.23	39.66	44.59	35.09	45.51	80.26	33.39	40.92	83.05	50.13	33.86	53.11	43.49	49.18
DAELM_S	50.26	45.42	37.58	42.88	38.31	36.94	34.20	36.44	80.89	32.32	34.35	72.88	45.21	44.36	47.11	45.73	45.28
DAELM_T	27.85	31.86	22.93	25.25	19.32	36.31	**41.90**	**46.49**	53.50	**48.08**	**51.20**	54.24	38.24	31.08	60.32	45.7	39.31
ARRLS	54.91	50.51	44.59	42.48	38.98	38.85	34.73	39.87	78.34	32.24	37.16	82.71	47.95	34.64	52.48	43.56	47.32
JPDA	47.60	45.76	46.5	40.78	40.68	36.94	34.55	33.82	88.54	34.73	34.66	91.19	47.98	59.2	68.94	**64.07**	50.28
AELM	52.40	43.73	47.77	40.16	34.92	35.67	30.54	37.16	86.62	28.67	29.54	83.73	45.91	41.65	62.83	52.24	46.81
DST-ELM (mar)	51.88	46.1	44.59	43.72	44.41	40.13	33.93	36.22	74.52	32.06	34.45	77.97	46.67	43.9	59.83	51.87	47.41
DST-ELM (con)	53.65	**51.53**	49.04	37.49	39.66	**50.96**	35.98	42.07	82.17	29.47	44.05	84.07	50.01	50.85	**73.39**	62.12	51.74
CDELM-M	52.07	51.05	45.86	42.33	42.85	45.86	30.31	39.83	81.15	30.31	35.72	81.76	48.26	46.53	62.47	54.5	49.15
CDELM-C	**56.28**	50.98	43.82	43.17	40.75	41.78	34.44	39.58	83.06	34.44	39.54	84.34	49.35	45.89	42.57	44.23	48.62
TELM-OWA	31.73	32.20	33.12	38.85	38.64	43.31	33.48	38.01	77.07	32.50	36.23	78.64	42.82	**59.55**	59.37	59.46	45.19
TSTELM	56.26	50.47	49.1	42.27	**44.75**	47.22	35.26	39.24	78.34	35.44	40.29	83.38	**50.17**	58.55	69.44	62.96	**52.86**

The bold values in Table 3 is best result in its column.

Table 4: Accuracy of different algorithms on MSRC + VOC2007 and Reuters-21578 datasets.

Methods\dataset	Non-transfer learning algorithm				Transfer learning algorithm								
	1NN	SVM	ELM	SSELM	TCA1	TCA2	JDA1	JDA2	DAELM_S	DAELM_T	ARRLS	TELM-OWA	TSTELM
MSRC vs VOC	35.95	35.10	38.56	37.52	27.91	34.90	27.12	34.77	38.34	33.60	34.64	**41.37**	36.47
VOC vs MSRC	45.23	54.69	58.00	63.20	47.91	54.22	48.31	54.61	58.37	30.61	50.35	66.22	**75.23**
Average	40.59	44.90	48.28	50.36	37.91	44.56	37.71	44.69	48.35	32.10	42.50	53.79	**55.85**
Orgs vs people (1)	72.85	75.25	77.24	80.30	76.49	77.51	80.13	81.79	76.64	48.71	80.55	63.74	**83.27**
People vs orgs (2)	72.03	77.12	79.30	84.72	77.28	79.39	85.13	85.29	78.41	47.72	84.56	64.27	**85.69**
Orgs vs place (3)	67.50	70.18	63.57	69.61	72.39	71.81	75.74	**76.22**	68.53	43.81	72.20	68.09	73.92
Place vs orgs (4)	61.12	63.77	64.57	71.16	68.21	65.75	**76.38**	76.08	68.77	42.50	74.11	74.68	73.52
People vs place (5)	52.65	60.63	57.01	66.02	61.19	60.63	**66.85**	65.83	59.14	42.46	65.18	59.04	61.74
Place vs people (6)	53.39	57.94	58.68	61.65	56.08	51.07	58.12	51.16	58.77	39.93	58.56	60.45	**60.81**
Average	63.26	67.48	66.73	72.24	68.61	67.69	73.73	72.73	68.38	44.19	72.53	65.05	**73.16**
Total average	57.59	61.84	62.12	66.77	60.93	61.91	64.72	65.72	63.37	41.17	65.02	62.23	**68.83**

The bold values in Table 3 is best result in its row.

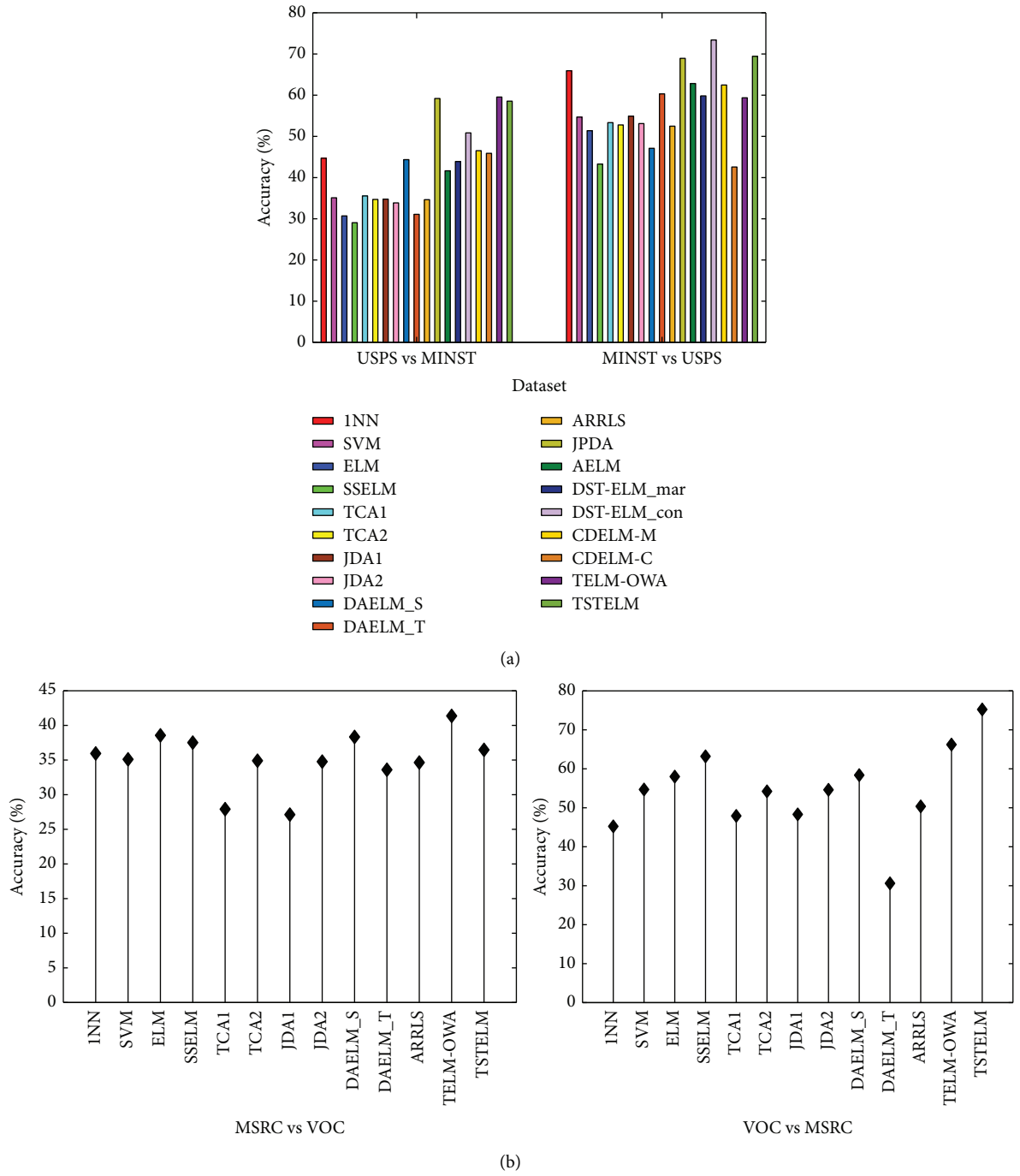

Figure 6: Classification accuracy of different algorithms on (a) USPS + MNIST and (b) MSRC + VOC2007 dataset.

best results of each task are bold. From the comparison of results, we have the following observations.

(1) The total average accuracy of TSTELM across 22 tasks is the highest among all methods, indicating that our approach can effectively deal with domain adaptation problem.

(2) In unsupervised domain adaptation, the proposed method outperforms AELM, DST-ELM, CDELM, TELM-OWA, DAELM_S, and DAELM_T, indicating the superiority of TSTELM in which MMD and output weight alignment are unified into the ELM learning framework to minimize distribution discrepancy between domains. AELM, DAELM_S, and DAELM_T obtain poor results, showing that they are highly dependent on labeled target samples. 1NN, SVM, and ELM perform unsuccessfully because of the problem of domain shift. SSELM performs better than ELM due to that the original geometry information of data is mined.

(3) TCA(1,2), JDA(1,2), and JPDA are better than 1NN and SVM, showing the importance of shared feature

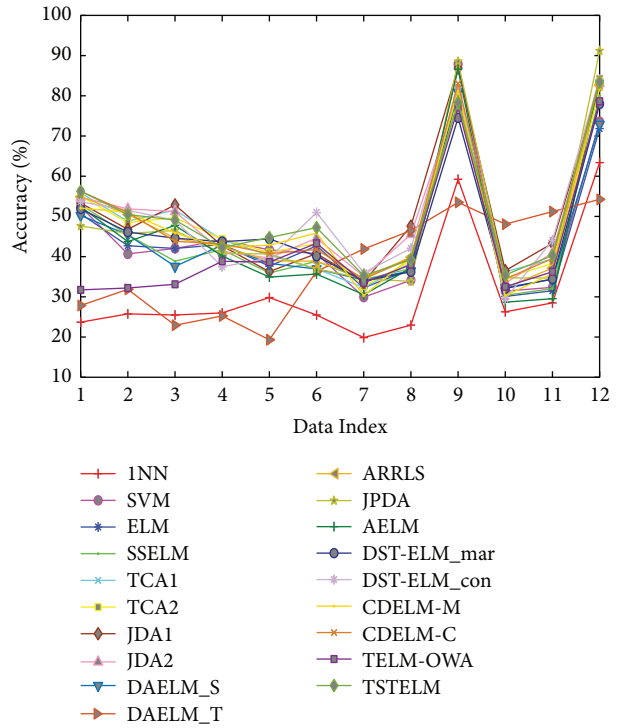

Figure 7: Classification accuracy of different algorithms on Office + Caltech256 dataset.

extraction across domains. JPDA and JDA(1,2) are generally higher than TCA(1,2), which indicates the superiority of simultaneously reducing the marginal and conditional distribution discrepancy. ARRLS performs well because of MMD and preserves the manifold consistency at the same time.

We check the execution times of some methods on MNIST vs USPS, and the results are reported in Table 5. It can be seen: (1) The speed of the methods based on ELM are significantly faster than other methods, and ELM is the fastest. (2) TSTELM consumes more time than TELM-OWA, ELM, SSELM, DAELM_S, and DAELM_T, because of label refinement iterative process. TELM-OWA is more time-consuming than ELM, SSELM, DAELM_S, and DAELM_T as a result of solving β_S^* and β_T^*. (3) Since constructing the Laplacian matrix is the most time-consuming, SSELM is relatively inefficient. (4) TCA(1,2) and JDA(1,2) cost more time than 1NN and SVM because of additional feature extraction process. (5) JDA(1,2) has the highest time cost because it applies an iterative manner to refine the target pseudo label and extract cross-domain shared feature.

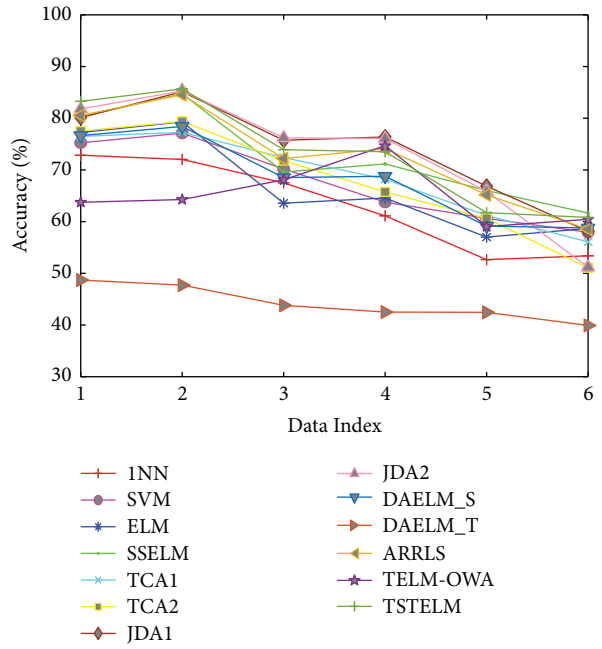

Figure 8: Classification accuracy of different algorithms on Reuters-21578 dataset.

4.4. Parameter Analysis. To evaluate the effects of scale factor (p); number of hidden layer nodes(L); parameter α, λ, and θ on TSTELM, we conduct some experiments on org vs people, MSRC vs VOC, MNIST vs USPS, and A vs D. The results are shown in Figures 9(a)–9(f). It can be seen that: (1) With the increase of p, the trend in TSTELM accuracy goes up first and then goes down on all test datasets and achieves optimal results when $p \in [0.1, 1]$, as shown in Figure 9(a). It can be known that the results of joint decision of β_1 and β_2 are better than their separate decisions. (2) As shown in Figure 9(b), TSTELM accuracy increases first and then decreases with the number of L on all test datasets. Although a large network forces the ELM network to behave better on output function approximation, time cost of the algorithm

Table 5: Average runtime of all the methods on MNIST vs USPS.

Algorithm	1NN	SVM	ELM	SSELM	TCA1	TCA2	JDA1	JDA2	DAELM_S	DAELM_T	ARRLS	TELM-OWA	TSTELM
Time (s)	0.55	8.79	0.37	3.49	4.72	5.47	48.32	53.6	0.81	0.64	2.08	3.7	36.5

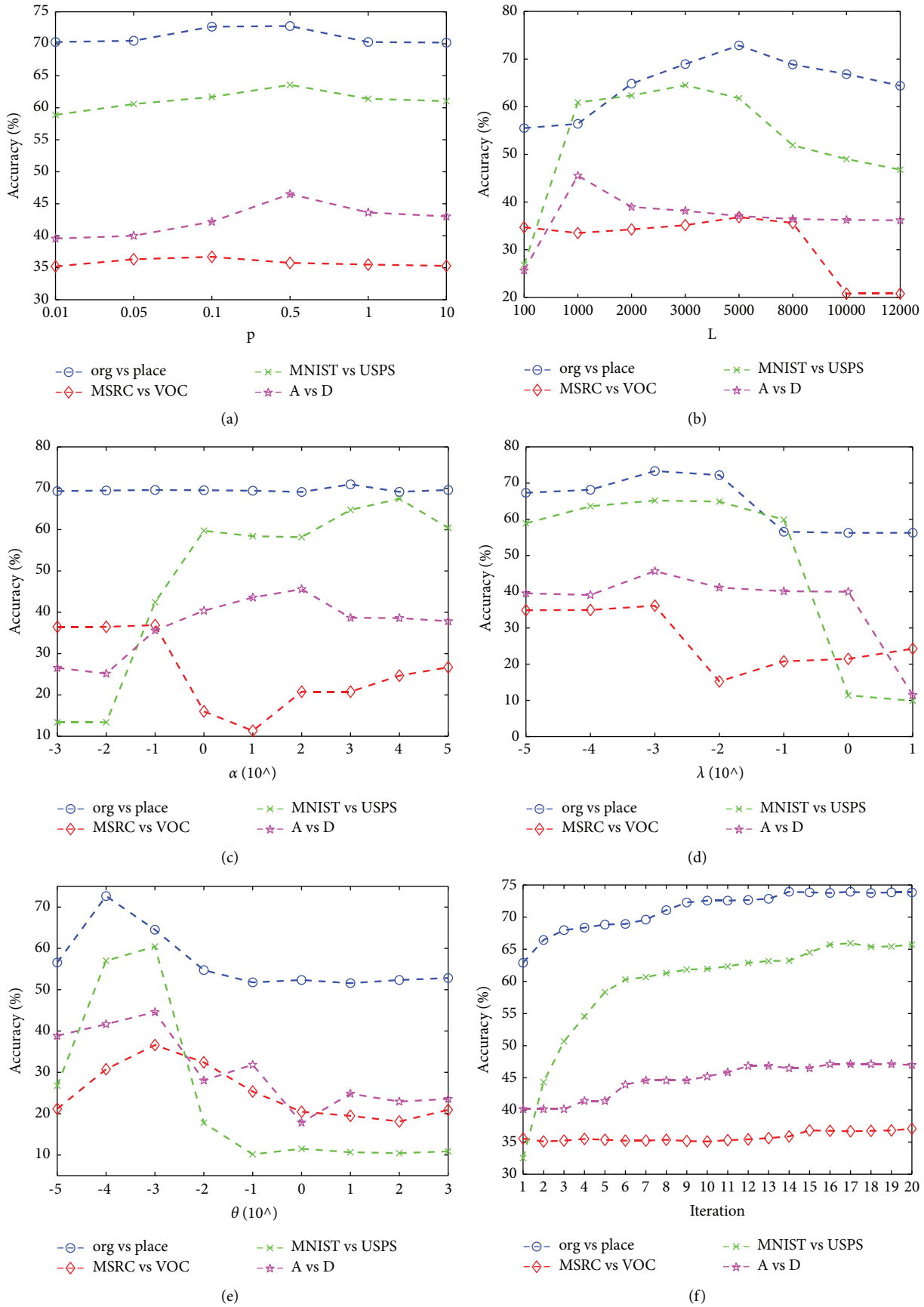

Figure 9: Classification accuracy of TSTELM with respect to scale factor (p); number of hidden layer nodes (L); parameter α, λ, and θ; and iteration.

and required memory become large, and too many hidden nodes will hurt the ELM performance of domain adaptation because of better output function approximation. (3) In Figures 9(c)–9(e), with the gradual increase of parameter α, λ, and θ, the accuracy increases first and then decreases and takes different optimal values on different test datasets to achieve the optimal accuracy, which indicates that the control term of these parameters are beneficial to TSTELM when the parameter values are reasonable. (4) We also provide the classification accuracy varying with the iteration number, and result is shown in Figure 9(f). It shows that the accuracy is increasing iterative with the number of iterations and finally converges after several iterations, which verifies that TSTELM has strong robustness.

5. Conclusion

To handle the problem that traditional ELM does not perform well in unsupervised domain adaptation, we in this article propose TSTELM including two domain adaptation stages. At the statistical matching stage, MMD is introduced into ELM learning frame to simultaneously minimize the marginal and conditional distribution between domains. At the subspace alignment stage, subspace alignment strategy, cross-domain mean approximation, and output weight approximation are adopted to further adjust the distribution consistency between domains. Finally, parameters of learned ELM models at two stages are fused and used to predict test samples. Extensive experiments have been conducted on real-world image and text datasets, and the results show that TSTELM has higher accuracy and better generalization performance. In the future, we will make further research that TSTELM is improved by stacking it into deep structure model for extracting deep feature.

Data Availability

The data used to support the findings of this study can be found at https://github.com/jindongwang/transferlearning/blob/master/data/dataset.md.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China, under Grant nos. 62073124 and 51805149, Key Scientific Research Projects of Universities in Henan Province, under grant no. 22A120005, and National Aviation Fund Projects, under Grant no 201701420002.

References

[1] A. Kammoun and M. S. AlouiniFellow, "On the precise error analysis of support vector machines," *IEEE Open Journal of Signal Processing*, vol. 2, pp. 99–118, 2021.

[2] M. M. Kumbure, P. Luukka, and M. Collan, "A new fuzzy k-nearest neighbor classifier based on the Bonferroni mean," *Pattern Recognition Letters*, vol. 140, pp. 172–178, 2020.

[3] Y. Fang, L. Xu, J. Peng, H. Yang, A. Wong, and D. A. Clausi, "Unsupervised Bayesian classification of a hyperspectral image based on the spectral mixture model and Markov random field," *Ieee Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 11, no. 9, pp. 3325–3337, 2018.

[4] A. Zollanvari and E. R. Dougherty, "Optimal Bayesian classification with vector autoregressive data dependency," *IEEE Transactions on Signal Processing*, vol. 67, no. 12, pp. 3073–3086, 2019.

[5] M. Li, H. Xu, and Y. Deng, "Evidential decision tree based on belief entropy," *Entropy*, vol. 21, no. 9, p. 897, 2019.

[6] D. Deb Nath, "Building an efficient classification model: a comparison of logistics regression and artificial neural network," *Journal of Management*, vol. 4, pp. 39–44, 2017.

[7] G. B. Huang, Q. Y. Zhu, and C. K. Siew, "Extreme learning machine: theory and applications," *Neurocomputing*, vol. 70, no. 1-3, pp. 489–501, 2006.

[8] J. Wang, S. Lu, S. H. Wang, and Y. D. Zhang, "A Review on Extreme Learning Machine," *Multimedia Tools and Applications*, pp. 1–50, 2021.

[9] G. Feng, G. B. Huang, Q. Lin, and R. Gay, "Error minimized extreme learning machine with growth of hidden nodes and incremental learning," *IEEE Transactions on Neural Networks*, vol. 20, no. 8, pp. 1352–1357, 2009.

[10] Y. Gu and L. Hua, "A novel smart motor imagery intention human-computer interaction model using extreme learning machine and eeg signals," *Frontiers in Neuroscience*, vol. 15, Article ID 685119, 2021.

[11] Q. She, J. Zou, M. Meng, Y. Fan, and Z. Luo, "Balanced Graph-based regularized semi-supervised extreme learning machine for EEG classification," *International Journal of Machine Learning and Cybernetics*, vol. 12, no. 4, pp. 903–916, 2021.

[12] J. Xia, D. Yang, H. Zhou et al., "Evolving kernel extreme learning machine for medical diagnosis via a disperse foraging sine cosine algorithm," *Computers in Biology and Medicine*, vol. 141, Article ID 105137, 2022.

[13] M. Eshtay, H. Faris, and N. Obeid, "Improving extreme learning machine by competitive swarm optimization and its application for medical diagnosis problems," *Expert Systems with Applications*, vol. 104, pp. 134–152, 2018.

[14] Z. Li, W. Jiang, S. Zhang, Y. Sun, and S. Zhang, "A hydraulic pump fault diagnosis method based on the modified ensemble empirical mode decomposition and wavelet kernel extreme learning machine methods," *Sensors*, vol. 21, no. 8, p. 2599, 2021.

[15] U. Ergul and G. Bilgin, "MCK-ELM: multiple composite kernel extreme learning machine for hyperspectral images," *Neural Computing & Applications*, vol. 32, no. 11, pp. 6809–6819, 2020.

[16] G. B. Huang, H. Zhou, X. Ding, and R. Zhang, "Extreme learning machine for regression and multiclass classification," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 42, no. 2, pp. 513–529, 2012.

[17] L. L. Li, J. Sun, M. L. Tseng, and Z. G. Li, "Extreme learning machine optimized by whale optimization algorithm using insulated gate bipolar transistor module aging degree evaluation," *Expert Systems with Applications*, vol. 127, pp. 58–67, 2019.

[18] H. Zhu, G. Liu, M. Zhou, Y. Xie, A. Abusorrah, and Q. Kang, "Optimizing Weighted Extreme Learning Machines for imbalanced classification and application to credit card fraud detection," *Neurocomputing*, vol. 407, pp. 50–62, 2020.

[19] W. Zong, G. B. Huang, and Y. Chen, "Weighted extreme learning machine for imbalance learning," *Neurocomputing*, vol. 101, pp. 229–242, 2013.

[20] J. Li, X. Shi, Z. H. You et al., "Using weighted extreme learning machine combined with scale-invariant feature transform to predict protein-protein interactions from protein evolutionary information," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 17, no. 5, pp. 1546–1554, 2020.

[21] B. S. Raghuwanshi and S. Shukla, "SMOTE based class-specific extreme learning machine for imbalanced learning," *Knowledge-Based Systems*, vol. 187, Article ID 104814, 2020.

[22] M. Mengcan, C. H. E. N. Xiaofang, and X. I. E. Yongfang, "Constrained voting extreme learning machine and its application," *Journal of Systems Engineering and Electronics*, vol. 32, no. 1, pp. 209–219, 2021.

[23] L. Zhai and J. Zhai, "Fault diagnosis for oil-filled transformers using voting based extreme learning machine," *Cluster Computing*, vol. 22, no. 4, pp. 8363–8370, 2019.

[24] A. O. Abuassba, D. Zhang, and X. Luo, "A heterogeneous AdaBoost ensemble based extreme learning machines for imbalanced data," *International Journal of Cognitive Informatics and Natural Intelligence 2019*, vol. 13, no. 3, pp. 19–35, 2019.

[25] M. Sharifmoghadam and H. Jazayeriy, "Breast cancer classification using AdaBoost-extreme learning machine," in *Proceedings of the Iranian Conference on Signal Processing and Intelligent Systems (ICSPIS 2019)*, pp. 1–8, IEEE, Shahrood, Iran, December 2019.

[26] H. Ge, W. Sun, M. Zhao, K. Zhang, L. Sun, and C. Yu, "Multi-grained cascade AdaBoost extreme learning machine for feature representation," in *Proceedings of the 2019 International Joint Conference on Neural Networks (IJCNN 2019)*, pp. 1–8, IEEE, Budapest, Hungary, July 2019.

[27] E. G. Mansoori and M. Sara, "Extreme ensemble of extreme learning machines," *Statistical Analysis and Data Mining: The ASA Data Science Journal*, vol. 14, no. 2, pp. 116–128, 2021.

[28] J. Zhang, Y. Li, W. Xiao, and Z. Zhang, "Non-iterative and fast deep learning: multilayer extreme learning machines," *Journal of the Franklin Institute*, vol. 357, no. 13, pp. 8925–8955, 2020.

[29] L. L. C. Kasun, H. Zhou, G. B. Huang, and C. M. Vong, "Representational learning with extreme learning machine for big data," *IEEE Intelligent Systems*, vol. 28, no. 6, pp. 31–34, 2013.

[30] R. Li, X. Wang, Y. Song, and L. Lei, "Hierarchical extreme learning machine with L21-norm loss and regularization," *International Journal of Machine Learning and Cybernetics*, vol. 12, no. 5, pp. 1297–1310, 2021.

[31] H. G. Han, L. D. Wang, and J. F. Qiao, "Hierarchical extreme learning machine for feedforward neural network," *Neurocomputing*, vol. 128, pp. 128–135, 2014.

[32] C. M. Wong, C. M. Vong, P. K. Wong, and J. Cao, "Kernel-based multilayer extreme learning machines for representation learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 3, pp. 757–762, 2016.

[33] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2009.

[34] G. Wilson and D. J. Cook, "A survey of unsupervised deep domain adaptation," *ACM Transactions on Intelligent Systems and Technology*, vol. 11, no. 5, pp. 1–46, 2020.

[35] S. F. Zang, Y. H. Cheng, X. S. Wang, Q. Yu, and G. S. Xie, "Cross domain mean approximation for unsupervised domain adaptation," *IEEE Access*, vol. 8, pp. 139052–139069, 2020.

[36] L. Zhang and D. Zhang, "Domain adaptation extreme learning machines for drift compensation in E-nose systems," *IEEE Transactions on Instrumentation and Measurement*, vol. 64, no. 7, pp. 1790–1801, 2015.

[37] L. Zhang, Z. He, and Y. Liu, "Deep object recognition across domains based on adaptive extreme learning machine," *Neurocomputing*, vol. 239, pp. 194–203, 2017.

[38] S. Zang, Y. Cheng, X. Wang, and Y. Yan, "Transfer extreme learning machine with output weight Alignment," *Computational Intelligence and Neuroscience*, vol. 2021, Article ID 6627765, 14 pages, 2021.

[39] S. Li, S. Song, G. Huang, and C. Wu, "Cross-domain extreme learning machines for domain adaptation," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 49, no. 6, pp. 1194–1207, 2018.

[40] Y. Chen, S. Song, S. Li, L. Yang, and C. Wu, "Domain space transfer extreme learning machine for domain adaptation," *IEEE Transactions on Cybernetics*, vol. 49, no. 5, pp. 1909–1922, 2018.

[41] Y. Liu, L. Zhang, P. L. Deng, and Z. He, "Common subspace learning via cross-domain extreme learning machine," *Cognitive Computation*, vol. 9, no. 4, pp. 555–563, 2017.

[42] Y. Si, J. Pu, S. F. Zang, and L. Sun, "Extreme learning machine based on maximum weighted mean discrepancy for unsupervised domain adaptation," *IEEE Access*, vol. 9, pp. 2283–2293, 2021.

[43] L. Zhang and X. Gao, "Transfer Adaptation Learning: A Decade Survey," 2022, http://arxiv.org/abs/1903.04687.

[44] S. Li, S. Song, and G. Huang, "Prediction reweighting for domain adaptation," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 7, pp. 1682–1695, 2017.

[45] L. Yan, R. Zhu, Y. Liu, and N. Mo, "TrAdaBoost based on improved particle swarm optimization for cross-domain scene classification with limited samples," *Ieee Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 11, no. 9, pp. 3235–3251, 2018.

[46] A. Gretton, A. Smola, J. Huang, M. Schmittfull, K. Borgwardt, and B. Schölkopf, "Covariate shift by kernel mean matching," *Dataset Shift in Machine Learning*, vol. 3, no. 4, pp. 131–160, 2008.

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

[48] M. Long, J. Wang, G. Ding, J. Sun, and P. S. Yu, "Transfer feature learning with joint distribution adaptation," in *Proceedings of the IEEE International Conference on Computer Vision(ICCV)*, pp. 2200–2207, Sydney, Australia, December 2013.

[49] J. Liang, R. He, Z. Sun, and T. Tan, "Aggregating randomized clustering-promoting invariant projections for domain adaptation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 5, pp. 1027–1042, 2019.

[50] J. Liang, R. He, Z. Sun, and T. Tan, "Exploring uncertainty in pseudo-label guided unsupervised domain adaptation," *Pattern Recognition*, vol. 96, Article ID 106996, 2019.

[51] J. Yang, R. Yan, and A. G. Hauptmann, "Adapting SVM classifiers to data with shifted distributions," in *Proceedings of the Seventh IEEE International Conference on Data Mining Workshops (ICDMW2007)*, pp. 69–76, Omaha, NE, USA, October 2007.

[52] T. Tommasi, F. Orabona, and B. Caputo, "Learning categories from few examples with multi model knowledge transfer," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 5, pp. 928–941, 2013.

[53] Z. Wang, B. Du, W. Tu, L. Zhang, and D. Tao, "Incorporating distribution matching into uncertainty for multiple kernel active learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 33, no. 1, pp. 128–142, 2021.

[54] Z. Wang, B. Du, and Y. Guo, "Domain adaptation with neural embedding matching," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 7, pp. 2387–2397, 2019.

[55] Z. Wang, B. Du, Q. Shi, and W. Tu, "Domain adaptation with discriminative distribution and manifold embedding for hyperspectral image classification," *IEEE Geoscience and Remote Sensing Letters*, vol. 16, no. 7, pp. 1155–1159, 2019.

[56] J. Liang, Y. Wang, D. Hu, R. He, and J. Feng, "A balanced and uncertainty-aware approach for partial domain adaptation," in *Proceedings of the European Conference on Computer Vision (ECCV2020)*, pp. 123–140, Glasgow, UK, November 2020.

[57] B. Gong, Y. Shi, and F. Sha, "Geodesic flow kernel for unsupervised domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition(CVPR2012)*, pp. 2066–2073, Providence, RI, USA, June 2012.

[58] G. Huang, S. Song, J. N. D. Gupta, and C. Wu, "Semi-supervised and unsupervised extreme learning machines," *IEEE Transactions on Cybernetics*, vol. 44, no. 12, pp. 2405–2417, 2014.

[59] M. Long, J. Wang, G. Ding, S. J. Pan, and P. S. Yu, "Adaptation regularization: a general framework for transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 5, pp. 1076–1089, 2014.

[60] W. Zhang and D. Wu, "Discriminative joint probability maximum mean discrepancy (DJP-MMD) for domain adaptation," in *Proceedings of the 2020 International Joint Conference on Neural Networks (IJCNN2020)*, pp. 1–8, Glasgow, UK, July 2020.

[61] M. Uzair and A. Mian, "Blind domain adaptation with augmented extreme learning machine features," *IEEE Transactions on Cybernetics*, vol. 47, no. 3, pp. 651–660, 2017.