Supervised Representation Learning with Double Encoding-Layer Autoencoder for Transfer Learning

FUZHEN ZHUANG, XIAOHU CHENG, and PING LUO, Key Lab of Intelligent Information Processing of Chinese Academy of Sciences (CAS), Institute of Computing Technology, CAS, and University of Chinese Academy of Sciences
SINNO JIALIN PAN, Nanyang Technological University, Singapore
QING HE, Key Lab of Intelligent Information Processing of Chinese Academy of Sciences (CAS), Institute of Computing Technology, and University of Chinese Academy of Sciences

Transfer learning has gained a lot of attention and interest in the past decade. One crucial research issue in transfer learning is how to find a good representation for instances of different domains such that the divergence between domains can be reduced with the new representation. Recently, deep learning has been proposed to learn more robust or higher-level features for transfer learning. In this article, we adapt the autoencoder technique to transfer learning and propose a supervised representation learning method based on double encoding-layer autoencoder. The proposed framework consists of two encoding layers: one for embedding and the other one for label encoding. In the embedding layer, the distribution distance of the embedded instances between the source and target domains is minimized in terms of KL-Divergence. In the label encoding layer, label information of the source domain is encoded using a softmax regression model. Moreover, to empirically explore why the proposed framework can work well for transfer learning, we propose a new effective measure based on autoencoder to compute the distribution distance between different domains. Experimental results show that the proposed new measure can better reflect the degree of transfer difficulty and has stronger correlation with the performance from supervised learning algorithms (e.g., Logistic Regression), compared with previous ones, such as KL-Divergence and Maximum Mean Discrepancy. Therefore, in our model, we have incorporated two distribution distance measures to minimize the difference between source and target domains in the embedding representations. Extensive experiments conducted on three real-world image datasets and one text data demonstrate the effectiveness of our proposed method compared with several state-of-the-art baseline methods.

CCS Concepts: • **Computing methodologies** → **Transfer learning**; • **Information systems** → **Clustering and classification**; • **Computer systems organization** → *Neural networks*;

Additional Key Words and Phrases: Double encoding-layer autoencoder, representation learning, distribution difference measure

This work is supported by the National Natural Science Foundation of China (No. 61773361, 61473273, 91546122, 61573335, 61602438), Guangdong provincial science and technology plan projects (No. 2015 B010109005), 2015 Microsoft Research Asia Collaborative Research Program, and the Youth Innovation Promotion Association CAS 2017146.
Authors' addresses: F. Zhuang, X. Cheng, P. Luo, and Q. He, Key Lab of Intelligent Information Processing of Chinese Academy of Sciences (CAS), Institute of Computing Technology, CAS, and University of Chinese Academy of Sciences, Kexueyuan Nanlu #6, Zhongguancun, Haidian District, Beijing; emails: {zhuangfz, chengxh}@ics.ict.ac.cn, luop@ict.ac.cn, heq@ics.ict.ac.cn; S. J. Pan, Nanyang Technological University, Singapore; email: sinnopan@ntu.edu.sg.

https://doi.org/10.1145/3108257

ACM Reference format:
Fuzhen Zhuang, Xiaohu Cheng, Ping Luo, Sinno Jialin Pan, and Qing He. 2017. Supervised Representation Learning with Double Encoding-Layer Autoencoder for Transfer Learning. *ACM Trans. Intell. Syst. Technol.* 9, 2, Article 16 (October 2017), 17 pages.
https://doi.org/10.1145/3108257

1 INTRODUCTION

Transfer learning aims to adapt knowledge from an auxiliary source domain to a target domain with little or without any label information to build a target prediction model of good generalization performance. In the past decade, a vast amount of attention has been paid on developing methods to transfer knowledge effectively across domains (Pan and Yang 2010). A crucial research issue in transfer learning is how to reduce the difference between the source and target domains while preserving original data properties. Among different approaches to transfer learning, the feature-based transfer-learning methods have proven to be superior for the scenarios where original raw data between domains are very different, while the divergence between domains can be reduced. A common objective of feature-based transfer-learning methods is to learn a transformation to project instances from different domains to a common latent space where the degree of distribution mismatch of the projected instances between domains can be reduced (Blitzer et al. 2006; Dai et al. 2007a; Pan et al. 2008, 2011; Zhuang et al. 2014).

Recently, because of the power on learning *high-level features*, deep learning has been applied to transfer learning (Xavier and Bengio 2011; Chen et al. 2012; Joey Tianyi Zhou and Yan 2014). Xavier and Bengio (2011) proposed to learn robust features with stacked denoising autoencoders (SDA) (Vincent et al. 2010) on the union of data of a number of domains. The learned new features are considered as high-level features and used to represent both the source and target domain data. Finally, standard classifiers are trained on the source domain labeled data with the new representations and make predictions on the target domain data. Chen et al. (2012) extended the work of SDA, and proposed the marginalized SDA (mSDA) for transfer learning. mSDA addresses two limitations of SDA: highly computational cost and lack of scalability with high-dimensional features.

Though the goal of previous deep-learning-based methods for transfer learning is trying to learn a more powerful representation to reduce the difference between domains, most of them did not explicitly minimize the distribution distance between domains when learning the representation. Therefore, the learned feature representation can not guarantee the reduction of distribution difference. Moreover, most previous methods are unsupervised, which thus fail to encode discriminative information into the representation learning.

In the previous work (Zhuang et al. 2015), we proposed a supervised representation learning method for transfer learning based on double encoding-layer autoencoder. Specifically, the proposed method, named *T*ransfer *L*earning with *D*ouble encoding-layer *A*utoconders (TLDA), is shown in Figure 1. In TLDA, there are two encoding and decoding layers, respectively, where the encoding and decoding weights are shared by both the source and target domains. The first encoding layer is referred to as the embedding layer, where the distributions of the embedded instances between source and target domains are enforced to be similar by minimizing the KL divergence (Kullback 1987). The second encoding layer is referred to as the label encoding layer, where the source domain label information is encoded using a softmax regression model (Friedman and Rob 2010), which can naturally handle multiple classes. It is worth mentioning that the encoding weights are also used for the final classification model in the second encoding layer.

In this article, we further investigate why our proposed double encoding-layer autoencoder can work for transfer learning. One of the most important issues in transfer learning is how to measure

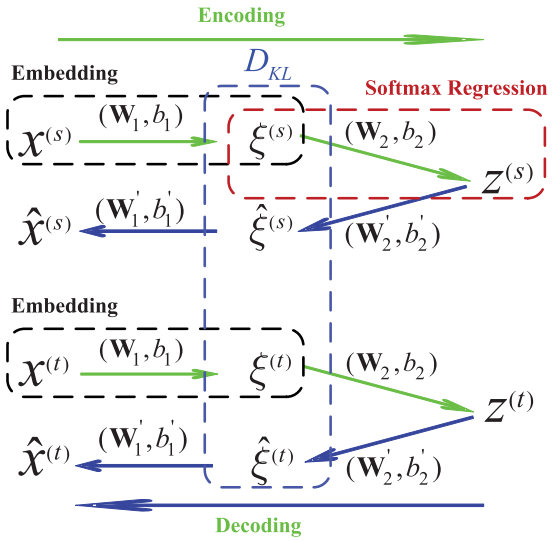


Fig. 1. The framework of TLDA.

the distance difference between different domains. Though the past decade has witnessed many research works devoted to transfer-learning algorithms, how to design an effective distance measure is still an open and challenging problem. To that end, we propose to adapt autoencoder (AE) to measuring the distance between different domains. Specifically, we first run autoencoder code over the source domain data to derive the encoding and decoding weights, which are then applied to target domain data. Finally, the distance measure is defined as the reconstruction error on target domain data. In other words, if the reconstruction error on target domain data is small, which means that the learnt encoding and decoding weights from source domain can fit well on target domain data. Therefore, their distributions are regarded to be similar, vice versa. Experimental results show that the proposed new measure can better reflect the degree of transfer difficulty and has stronger correlation with the performance from supervised learning algorithms (e.g., Logistic Regression), compared with previous ones, such as KL-Divergence and Maximum Mean Discrepancy (MMD). In the proposed framework TLDA, the encoding and decoding weights are shared across different domains for knowledge transfer, which means that autoencoder is also used to draw the distribution of embedded instances between source and target domains to be more similar. Overall, both KL-divergence and Autoencoder are considered to draw the distribution closer, which leads to the improvement of our framework. Furthermore, we also conduct additional experiments on a real-world text dataset, which again validate the effectiveness of the proposed model.

In summary, the main contributions of this article are highlighted as follows:

(1) For the representation learning for transfer learning, we newly propose to use the double encoding-layer autoencoder to learn common latent representations of source and target domains, in which the encoding and decoding weights are shared across domains for knowledge transfer.
(2) The label information from source domain is tactfully incorporated by softmax regression model, whose model parameters are shared with the second-layer encoding weights.

(3) To empirically analyze why TLDA can work, we further develop a new distance measure based on autoencoder, which can better reflect the degree of transfer difficulty. Thus, in TLDA there indeed two distribution difference measures are considered to enforce the distributions of two domains to be similar.
(4) Extensive experiments conducted on three real-world image datasets and one text data demonstrate the effectiveness of our proposed method compared with several state-of-the-art baseline methods.

The remainder of this article is organized as follows. Related work are first summarized in Section 2, and some preliminary knowledge is introduced in Section 3. Section 4 details the problem formulation and model learning. In Section 5, we conduct extensive experiments on image and text classification problems to demonstrate the effectiveness of the proposed model. Finally, Section 6 concludes the article.

2 RELATED WORK

Since we adopt the transfer-learning techniques for transfer learning in this work, we first would like to introduce some deep-learning methods for representation learning, and then the most related works of transfer learning.

Poultney et al. (2006) proposed an unsupervised method with an energy-based model for learning sparse and overcomplete features. In their method, the decoder produces accurate reconstructions of the patches, while the encoder provides a fast prediction of the code without the need for any particular preprocessing of the inputs. Vincent and Manzagol (2008) proposed Denoising autoencoders to learn a more robust representation from an artificially corrupted input, and further proposed Stacked denoising autoencoders (Vincent et al. 2010) to learn useful representations through a deep network. Joey Tianyi Zhou and Yan (2014) proposed a deep-learning approach to heterogeneous transfer learning based on an extension of mSDA, where instances in the source and target domains are represented by heterogeneous features. In their proposed method, the bridge between the source and target domains with heterogeneous features is built based on the corresponding information of instances between the source and target domains, which is assumed to be given in advance. Tzeng et al. (2015) proposed a new CNN architecture to exploit unlabeled and sparsely labeled target domain data, which simultaneously optimizes for domain invariance to facilitate domain transfer and uses a soft label distribution matching loss to transfer information between tasks. Also, Ganin and Lempitsky (2015) proposed a new approach to domain adaptation that can make full use of large amount of labeled data from the source domain and large amount of unlabeled data from the target domain in a deep architecture. The most related work is learning transferable features with deep adaptation networks (Long et al. 2015, 2016), they proposed a unified deep adaptation framework for jointly learning transferable representation and classifier to enable scalable domain adaptation, by taking the advantages of both deep learning and optimal two-sample matching. The main difference is that our model contains only two encoding-layer under a supervised learning framework, which does not need to tune the depth of networks. Of course, it would be promising to achieve better results by making deeper networks.

Transfer learning has attracted much attention in the past decade. To reduce the difference between domains, two categories of transfer-learning approaches have been proposed. One is based on the instance level, which aims to learn weights for the source domain labeled data, such that the re-weighted source domain instances look similar to the target domain data instances (Dai et al. 2007b; Gao et al. 2008; Xing et al. 2007; Jiang and Zhai 2007; Zhuang et al. 2010; Crammer et al. 2012). The other is based on the feature representation level, which aims to learn a new feature representation for both the source and target domain data, such that with the new feature

Table 1. The Notation and Denotation

$\mathcal{D}_s, \mathcal{D}_t$	The source and target domains
n_s	The number of instances in source domain
n_t	The number of instances in target domain
m	The number of original features
k	The number of nodes in embedding layer
c	The number of nodes in label layer
$\boldsymbol{x}_i^{(s)}, \boldsymbol{x}_i^{(t)}$	The i-th instance of source and target domains
$\hat{\boldsymbol{x}}_i^{(s)}, \hat{\boldsymbol{x}}_i^{(t)}$	The reconstructions of $\boldsymbol{x}_i^{(s)}$ and $\boldsymbol{x}_i^{(t)}$
$y_i^{(s)}$	The label of instance $\boldsymbol{x}_i^{(s)}$
$\boldsymbol{\xi}_i^{(s)}, \boldsymbol{\xi}_i^{(t)}$	The hidden representations of $\boldsymbol{x}_i^{(s)}$ and $\boldsymbol{x}_i^{(t)}$
$\hat{\boldsymbol{\xi}}_i^{(s)}, \hat{\boldsymbol{\xi}}_i^{(t)}$	The reconstructions of $\boldsymbol{\xi}_i^{(s)}$ and $\boldsymbol{\xi}_i^{(t)}$
$\boldsymbol{z}_i^{(s)}, \boldsymbol{z}_i^{(t)}$	The hidden representations of $\boldsymbol{\xi}_i^{(s)}$ and $\boldsymbol{\xi}_i^{(t)}$
$\boldsymbol{W}_i, \boldsymbol{b}_i$	Encoding weight and bias matrix for layer i
$\boldsymbol{W}'_i, \boldsymbol{b}'_i$	Decoding weight and bias matrix for layer i
$\top$	The transposition of a matrix
$\circ$	The element-wise product of vectors or matrixes

representation the difference between domains can be reduced (Blitzer et al. 2006; Dai et al. 2007a; Pan et al. 2008; Si et al. 2010; Pan et al. 2011; Xavier and Bengio 2011; Chen et al. 2012; Zhuang et al. 2014; Gong et al. 2016). Based on the observation that multi-task share similar feature structures, Liu et al. (2017) presented novel algorithm-dependent generalization bounds for MTL by exploiting the notion of algorithmic stability. There are also some works about transfer metric learning, for example, Luo et al. (2014) proposed a decomposition-based transfer distance metric learning algorithm for image classification, which considered the transfer-learning setting by exploiting the large quantity of side information from certain related, but different source tasks to help with target metric learning.

Among most feature-based transfer-learning methods, only a few methods aim to minimize the difference between domains explicitly in learning the new feature representation. For instance, maximum mean discrepancy embedding (MMDE) (Pan et al. 2008) and transfer component analysis (TCA) (Pan et al. 2011) tried to minimize the distance in distributions between domains in a kernel Hilbert space, respectively. The transfer subspace learning framework proposed by Si et al. (2010) tried to find a subspace, where the distributions of the source and target domain data are similar, through a minimization on the KL divergence of the projected instances between domains. However, they are either based on kernel methods or regularization frameworks, rather than exploring a deep architecture to learn feature representations for transfer learning. Different from previous works, in this article, our proposed TLDA is a supervised representation learning method based on deep learning, which takes distance minimization between domains and label encoding of the source domain into consideration.

3 PRELIMINARY KNOWLEDGE

The frequently used notations are listed in Table 1, and unless otherwise specified, all the vectors are column vectors. In this section, we first review some preliminary knowledge that is used in our proposed framework.

3.1 Autoencoder

The simplest form of an autoencoder (Bengio 2009) is a feed forward neural network with an input layer, an output layer and one or more hidden layers connecting them. But architecturally, an autoencoder with the output layer having the same number of nodes as the input layer, and with the purpose of reconstructing its own inputs. An autoencoder framework usually includes the encoding and decoding processes. Given an input $\boldsymbol{x}$, autoencoder first encodes it to one or more hidden layers through several encoding processes, then decodes the hidden layers to obtain an output $\hat{\boldsymbol{x}}$. Autoencoder tries to minimize the deviation of $\hat{\boldsymbol{x}}$ from the input $\boldsymbol{x}$, and the process of autoencoder with one hidden layer can be summarized as:

$$\text{Encoding}: \boldsymbol{\xi} = f(\boldsymbol{W}_1\boldsymbol{x} + \boldsymbol{b}_1), \tag{1}$$

$$\text{Decoding}: \hat{\boldsymbol{x}} = f(\boldsymbol{W}'_1\boldsymbol{\xi} + \boldsymbol{b}'_1), \tag{2}$$

where f is a nonlinear activation function (the sigmoid function, $f(u) = \frac{1}{1+e^{-u}}$, is adopted in this article), $\boldsymbol{W}_1 \in \mathbb{R}^{k\times m}$ and $\boldsymbol{W}'_1 \in \mathbb{R}^{m\times k}$ are weight matrices, $\boldsymbol{b}_1 \in \mathbb{R}^{k\times 1}$ and $\boldsymbol{b}'_1 \in \mathbb{R}^{m\times 1}$ are bias vectors, and $\boldsymbol{\xi} \in \mathbb{R}^{k\times 1}$ is the output of the hidden layer. Given a set of inputs $\{\boldsymbol{x}_i\}_{i=1}^n$, the reconstruction error can be computed by $\sum_{i=1}^n \|\hat{\boldsymbol{x}}_i - \boldsymbol{x}_i\|^2$. The goal of autoencoder is to learn the weight matrices $\boldsymbol{W}_1$ and $\boldsymbol{W}'_1$, and the bias vectors $\boldsymbol{b}_1$ and $\boldsymbol{b}'_1$ by minimizing the reconstruction error as follows,

$$\min_{\boldsymbol{W}_1, \boldsymbol{b}_1, \boldsymbol{W}'_1, \boldsymbol{b}'_1} \sum_{i=1}^{n} \|\hat{\boldsymbol{x}}_i - \boldsymbol{x}_i\|^2. \tag{3}$$

3.2 Softmax Regression

The softmax regression model (Friedman and Rob 2010) is a generalization of the logistic regression model for multi-class classification problems, where the class label y can take more than two values, that is, $y \in \{1, 2, \ldots, c\}$ (where $c \geq 2$ is the number of class labels). For a test instance $\boldsymbol{x}$, we can estimate the probabilities of each class that $\boldsymbol{x}$ belongs to as follows,

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \begin{bmatrix} p(y_i = 1|\boldsymbol{x};\boldsymbol{\theta}) \\ p(y_i = 2|\boldsymbol{x};\boldsymbol{\theta}) \\ \vdots \\ p(y_i = c|\boldsymbol{x};\boldsymbol{\theta}) \end{bmatrix} = \frac{1}{\sum_{j=1}^{c} e^{\boldsymbol{\theta}_j^\top \boldsymbol{x}}} \begin{bmatrix} e^{\boldsymbol{\theta}_1^\top \boldsymbol{x}} \\ e^{\boldsymbol{\theta}_2^\top \boldsymbol{x}} \\ \vdots \\ e^{\boldsymbol{\theta}_c^\top \boldsymbol{x}} \end{bmatrix}, \tag{4}$$

where $\sum_{j=1}^{c} e^{\boldsymbol{\theta}_j^\top \boldsymbol{x}}$ is a normalized term, and $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_c$ are the model parameters.

Given the training set $\{\boldsymbol{x}_i, y_i\}_{i=1}^n$, $y_i \in \{1, 2, \ldots, c\}$, the solution of softmax regression can be derived by minimizing the following optimization problem:

$$\min_{\boldsymbol{\theta}} \left(-\frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{c} 1\{y_i = j\} \log \frac{e^{\boldsymbol{\theta}_j^\top \boldsymbol{x}_i}}{\sum_{l=1}^{c} e^{\boldsymbol{\theta}_l^\top \boldsymbol{x}_i}} \right), \tag{5}$$

where $1\{\cdot\}$ is an indicator function, whose value is 1 if the expression is true, otherwise 0. Once the model is trained, one can compute the probability of instance $\boldsymbol{x}$ belonging to a label j using Equation (4) and assign its class label as

$$y = \max_j \frac{e^{\boldsymbol{\theta}_j^\top \boldsymbol{x}}}{\sum_{l=1}^{c} e^{\boldsymbol{\theta}_l^\top \boldsymbol{x}}}. \tag{6}$$

3.3 Kullback-Leibler Divergence

Kullback-Leibler (KL) divergence (Kullback 1987), also known as the relative entropy, is a non-symmetric measure of the divergence between two probability distributions. Given two probability distributions $\boldsymbol{P} \in \mathbb{R}^{k\times 1}$ and $\boldsymbol{Q} \in \mathbb{R}^{k\times 1}$, the KL divergence of $\boldsymbol{Q}$ from $\boldsymbol{P}$ is the information lost when $\boldsymbol{Q}$ is used to approximate $\boldsymbol{P}$ (Liddle et al. 2010), defined as $D_{KL}(\boldsymbol{P}||\boldsymbol{Q}) = \sum_{i=1}^{k} \boldsymbol{P}(i) \ln(\frac{\boldsymbol{P}(i)}{\boldsymbol{Q}(i)})$. In this article, we adopt the symmetrized version of KL-divergence, $KL(\boldsymbol{P}, \boldsymbol{Q}) = D_{KL}(\boldsymbol{P}||\boldsymbol{Q}) + D_{KL}(\boldsymbol{Q}||\boldsymbol{P})$, to measure the divergence for classification problems, smaller value of KL divergence indicates more similar of two distributions. Thus, we use the KL divergence to measure the difference between two data domains when they are embedded to the same latent space.

4 ADAPT DOUBLE ENCODING-LAYER AUTOENCODER TO TRANSFER LEARNING

4.1 Problem Formalization

Given two domains $\boldsymbol{D}_s$ and $\boldsymbol{D}_t$, where $\boldsymbol{D}_s=\{\boldsymbol{x}_i^{(s)}, y_i^{(s)}\}|_{i=1}^{n_s}$ is the source domain labeled data with $\boldsymbol{x}_i^{(s)} \in \mathbb{R}^{m\times 1}$, and $y_i^{(s)} \in \{1, \ldots, c\}$, while $\boldsymbol{D}_t = \{\boldsymbol{x}_i^{(t)}\}|_{i=1}^{n_t}$ is the target domain with unlabeled data. Here, n_s and n_t are the numbers of instances in $\boldsymbol{D}_s$ and $\boldsymbol{D}_t$, respectively.

As shown in Figure 1, there are three factors to be taken into consideration for representation learning. Therefore, the objective to be minimized in our proposed learning framework for transfer learning can be formalized as follows:

$$\begin{aligned}\mathcal{J} &= J_r(\boldsymbol{x}, \hat{\boldsymbol{x}}) + \alpha\Gamma(\boldsymbol{\xi}^{(s)}, \boldsymbol{\xi}^{(t)}) + \beta\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\xi}^{(s)}) \\ &\quad +\gamma\Omega(\boldsymbol{W}, \boldsymbol{b}, \boldsymbol{W}', \boldsymbol{b}').\end{aligned} \tag{7}$$

The first term of the objective is the reconstruction error for both source and target domain data, which can be defined as

$$J_r(\boldsymbol{x}, \hat{\boldsymbol{x}}) = \sum_{r\in\{s,t\}} \sum_{i=1}^{n_r} ||\boldsymbol{x}_i^{(r)} - \hat{\boldsymbol{x}}_i^{(r)}||^2, \tag{8}$$

where

$$\boldsymbol{\xi}_i^{(r)} = f(\boldsymbol{W}_1\boldsymbol{x}_i^{(r)} + \boldsymbol{b}_1), \boldsymbol{z}_i^{(r)} = f(\boldsymbol{W}_2\boldsymbol{\xi}_i^{(r)} + \boldsymbol{b}_2), \tag{9}$$

$$\hat{\boldsymbol{\xi}}_i^{(r)} = f(\boldsymbol{W}'_2\boldsymbol{z}_i^{(r)} + \boldsymbol{b}'_2), \hat{\boldsymbol{x}}_i^{(r)} = f(\boldsymbol{W}'_1\hat{\boldsymbol{\xi}}_i^{(r)} + \boldsymbol{b}'_1). \tag{10}$$

For these two encoding layers, the first one is called as embedding layer to find good representation with an output $\boldsymbol{\xi} \in \mathbb{R}^{k\times 1}$ of k nodes ($k \le m$), while the second one is called as label layer to encode label information with an output $\boldsymbol{z} \in \mathbb{R}^{c\times 1}$ of c nodes (equals to the number of class labels). The output of first layer is the input for the second hidden layer. Here, the softmax Regression is used as the regularization item on source domain to incorporate label information. In addition, the output of the second layer is used as the prediction results for target domain. The third hidden layer $\hat{\boldsymbol{\xi}} \in \mathbb{R}^{k\times 1}$ is the reconstruction of the embedding layer with the corresponding weight matrix and bias vector $\boldsymbol{W}'_2 \in \mathbb{R}^{k\times c}$ and $\boldsymbol{b}'_2 \in \mathbb{R}^{k\times 1}$. Finally, $\hat{\boldsymbol{x}} \in \mathbb{R}^{m\times 1}$ is the reconstruction of $\boldsymbol{x}$ with $\boldsymbol{W}'_1 \in \mathbb{R}^{m\times k}$ and $\boldsymbol{b}'_1 \in \mathbb{R}^{m\times 1}$.

The second term in the objective Equation (7) is the KL divergence of embedded instances between the source and target domains, which can be written as

$$\Gamma(\boldsymbol{\xi}^{(s)}, \boldsymbol{\xi}^{(t)}) = D_{KL}(\boldsymbol{P}_s||\boldsymbol{P}_t) + D_{KL}(\boldsymbol{P}_t||\boldsymbol{P}_s), \tag{11}$$

where

$$P'_s = \frac{1}{n_s}\sum_{i=1}^{n_s}\xi_i^{(s)}, \quad P_s = \frac{P'_s}{\sum P'_s}, \tag{12}$$

$$P'_t = \frac{1}{n_t}\sum_{i=1}^{n_t}\xi_i^{(t)}, \quad P_t = \frac{P'_t}{\sum P'_t}. \tag{13}$$

The goal of minimizing the KL divergence is to ensure the embedded source and target data distributions to be similar in the embedding space.

The third term in the objective Equation (7) is the loss function of softmax regression to incorporate the label information of the source domain into the embedding space. Specifically, this term can be formalized as follows:

$$\mathcal{L}(\boldsymbol{\theta}, \xi^{(s)}) = -\frac{1}{n_s}\sum_{i=1}^{n_s}\sum_{j=1}^{c} 1\{y_i^{(s)} = j\}\log\frac{e^{\boldsymbol{\theta}_j^\top \xi_i^{(s)}}}{\sum_{l=1}^{c} e^{\boldsymbol{\theta}_l^\top \xi_i^{(s)}}},$$

where $\boldsymbol{\theta}_j^\top (j \in \{1, \ldots, c\})$ is the jth row of $\boldsymbol{W}_2$.
Finally, the last term in the objective Equation (7) is an regularization on model parameters, which is defined as follows:

$$\begin{aligned}\Omega(\boldsymbol{W}, \boldsymbol{b}, \boldsymbol{W}', \boldsymbol{b}') = {} & \|\boldsymbol{W}_1\|^2 + \|\boldsymbol{b}_1\|^2 + \|\boldsymbol{W}_2\|^2 + \|\boldsymbol{b}_2\|^2 \\ & + \|\boldsymbol{W}'_1\|^2 + \|\boldsymbol{b}'_1\|^2 + \|\boldsymbol{W}'_2\|^2 + \|\boldsymbol{b}'_2\|^2.\end{aligned}$$

The trade-off parameters α, β, and γ are positive constants to balance the effect of different terms to the overall objective.

4.2 Model Learning

To minimize the problem of Equation (7) with respect to $\boldsymbol{W}_1$, $\boldsymbol{b}_1$, $\boldsymbol{W}_2$, $\boldsymbol{b}_2$, $\boldsymbol{W}'_2$, $\boldsymbol{b}'_2$, $\boldsymbol{W}'_1$, and $\boldsymbol{b}'_1$, we adopt the gradient descent method to derive the solution. For succinctness, we first introduce some intermediate variables as follows:

$$\begin{aligned} A_i^{(r)} &= \left(\hat{\boldsymbol{x}}_i^{(r)} - \boldsymbol{x}_i^{(r)}\right) \circ \hat{\boldsymbol{x}}_i^{(r)} \circ \left(1 - \hat{\boldsymbol{x}}_i^{(r)}\right), \\ B_i^{(r)} &= \hat{\xi}_i^{(r)} \circ \left(1 - \hat{\xi}_i^{(r)}\right), \\ C_i^{(r)} &= \boldsymbol{z}_i^{(r)} \circ \left(1 - \boldsymbol{z}_i^{(r)}\right), \\ D_i^{(r)} &= \xi_i^{(r)} \circ \left(1 - \xi_i^{(r)}\right). \end{aligned}$$

The partial derivatives of the objective Equation (7) w.r.t. $\boldsymbol{W}_1$, $\boldsymbol{b}_1$, $\boldsymbol{W}_2$, $\boldsymbol{b}_2$, $\boldsymbol{W}'_2$, $\boldsymbol{b}'_2$, $\boldsymbol{W}'_1$, and $\boldsymbol{b}'_1$ can be computed as follows, respectively,

$$\begin{aligned}\frac{\partial \mathcal{J}}{\partial \boldsymbol{W}_1} = {} & \sum_{i=1}^{n_s} 2\boldsymbol{W}'^{\top}_1 A_i^{(s)} \circ (\boldsymbol{W}_2^\top (\boldsymbol{W}'^{\top}_2 B_i^{(s)} \circ C_i^{(s)})) \circ D_i^{(s)} \boldsymbol{x}_i^{(s)\top} \\ & + \sum_{i=1}^{n_t} 2\boldsymbol{W}'^{\top}_1 A_i^{(t)} \circ (\boldsymbol{W}_2^\top (\boldsymbol{W}'^{\top}_2 B_i^{(t)} \circ C_i^{(t)})) \circ D_i^{(t)} \boldsymbol{x}_i^{(t)\top} \\ & + \frac{\alpha}{n_s}\sum_{i=1}^{n_s} D_i^{(s)} \circ \left(1 - \frac{P_t}{P_s} + ln\left(\frac{P_s}{P_t}\right)\right) \boldsymbol{x}_i^{(s)\top} \end{aligned} \tag{14}$$

$$+\frac{\alpha}{n_t}\sum_{i=1}^{n_t} D_i^{(t)} \circ \left(1 - \frac{\boldsymbol{P}_s}{\boldsymbol{P}_t} + ln\left(\frac{\boldsymbol{P}_t}{\boldsymbol{P}_s}\right)\right)\boldsymbol{x}_i^{(t)\top} + 2\gamma \boldsymbol{W}_1$$
$$-\frac{\beta}{n_s}\sum_{i=1}^{n_s}\sum_{j=1}^{c} 1\{y_i^{(s)} = j\}\left(\boldsymbol{W}_{2j}^{\top} - \frac{\boldsymbol{W}_2^{\top} e^{\boldsymbol{W}_2 \xi_i^{(s)}}}{\sum_l e^{\boldsymbol{W}_{2l}\xi_i^{(s)}}}\right) \circ D_i^{(s)} \boldsymbol{x}_i^{(s)\top},$$

$$\begin{aligned}\frac{\partial \mathcal{J}}{\partial \boldsymbol{W}_{2j}} = &\sum_{i=1}^{n_s} 2\boldsymbol{W}_{2j}'^{\top}(\boldsymbol{W}_1'^{\top} A_i^{(s)} \circ B_i^{(s)}) \circ C_{ij}^{(s)} \boldsymbol{\xi}_i^{(s)\top} \\ &+ \sum_{i=1}^{n_t} 2\boldsymbol{W}_{2j}'^{\top}(\boldsymbol{W}_1'^{\top} A_i^{(t)} \circ B_i^{(t)}) \circ C_{ij}^{(t)} \boldsymbol{\xi}_i^{(t)\top} \\ &- \frac{\beta}{n_{sj}}\left(\sum_{i=1}^{n_{sj}} \boldsymbol{\xi}_i^{(s)\top} - \sum_{i=1}^{n_s} \frac{e^{\boldsymbol{W}_{2j}\xi_i^{(s)}}}{\sum_l e^{\boldsymbol{W}_{2l}\xi_i^{(s)}}} \boldsymbol{\xi}_i^{(s)\top}\right) + 2\gamma \boldsymbol{W}_{2j},\end{aligned} \tag{15}$$

$$\begin{aligned}\frac{\partial \mathcal{J}}{\partial \boldsymbol{W}_2'} = &\sum_{i=1}^{n_s} 2\boldsymbol{W}_1'^{\top} A_i^{(s)} \circ B_i^{(s)} \boldsymbol{z}_i^{(s)\top} + 2\gamma \boldsymbol{W}_2' \\ &+ \sum_{i=1}^{n_t} 2\boldsymbol{W}_1'^{T} A_i^{(t)} \circ B_i^{(t)} \boldsymbol{z}_i^{(t)\top},\end{aligned} \tag{16}$$

$$\frac{\partial \mathcal{J}}{\partial \boldsymbol{W}_1'} = \sum_{i=1}^{n_s} 2A_i^{(s)} \hat{\boldsymbol{\xi}}_i^{(s)\top} + \sum_{i=1}^{n_t} 2A_i^{(t)} \hat{\boldsymbol{\xi}}_i^{(t)\top} + 2\gamma \boldsymbol{W}_1', \tag{17}$$

where $\boldsymbol{W}_{2j}$ is the jth row of $\boldsymbol{W}_2$, and n_{sj} is the number of instances with the label j in source domain. As the partial derivatives of the objective Equation (7) w.r.t. $\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_2', \boldsymbol{b}_1'$ are very similar to those of $\boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{W}_2', \boldsymbol{W}_1'$, respectively, we omit the details to avoid redundancy. Based on the preceding partial derivatives, we develop an alternatively iterating algorithm to derive the solutions by using the following rules:

$$\begin{aligned}&\boldsymbol{W}_1 \leftarrow \boldsymbol{W}_1 - \eta\frac{\partial \mathcal{J}}{\partial \boldsymbol{W}_1}, \quad \boldsymbol{b}_1 \leftarrow \boldsymbol{b}_1 - \eta\frac{\partial \mathcal{J}}{\partial \boldsymbol{b}_1}, \\ &\boldsymbol{W}_1' \leftarrow \boldsymbol{W}_1' - \eta\frac{\partial \mathcal{J}}{\partial \boldsymbol{W}_1'}, \quad \boldsymbol{b}_1' \leftarrow \boldsymbol{b}_1' - \eta\frac{\partial \mathcal{J}}{\partial \boldsymbol{b}_1'}, \\ &\boldsymbol{W}_2 \leftarrow \boldsymbol{W}_2 - \eta\frac{\partial \mathcal{J}}{\partial \boldsymbol{W}_2}, \quad \boldsymbol{b}_2 \leftarrow \boldsymbol{b}_2 - \eta\frac{\partial \mathcal{J}}{\partial \boldsymbol{b}_2}, \\ &\boldsymbol{W}_2' \leftarrow \boldsymbol{W}_2' - \eta\frac{\partial \mathcal{J}}{\partial \boldsymbol{W}_2'}, \quad \boldsymbol{b}_2' \leftarrow \boldsymbol{b}_2' - \eta\frac{\partial \mathcal{J}}{\partial \boldsymbol{b}_2'},\end{aligned} \tag{18}$$

where η is the step length, which determines the speed of convergence. The details of the proposed algorithm is summarized in Algorithm 1. Note that the proposed optimization problem is not convex, and thus there is no guarantee on obtaining an optimal global solution. To achieve a better local optimal solution of the proposed gradient descent approach, we first run SAE on all source and target domain data for pre-training, and then use the output of SAE to initialize the encoding and decoding weights.

ALGORITHM 1: Transfer Learning with Double Encoding-Layer Autoencoder (TLDA)

Input: Given one source domain $\boldsymbol{D}_s = \{\boldsymbol{x}_i^{(s)}, y_i^{(s)}\}|_{i=1}^{n_s}$, and one target domain $\boldsymbol{D}_t = \{\boldsymbol{x}_i^{(t)}\}|_{i=1}^{n_t}$, trade-off parameters α, β, γ, the number of nodes in embedding layer and label layer, k and c.
Output: Results of label layer $\boldsymbol{z}$ and embedded layer $\boldsymbol{\xi}$.

(1) Initialize $\boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{W}'_2, \boldsymbol{W}'_1$ and $\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}'_2, \boldsymbol{b}'_1$ by Stacked Autoencoders performed on both source and target domains;
(2) Compute the partial derivatives of all variables according to Equations (14), (15), (16), and (17);
(3) Iteratively update the variables using Equations (18);
(4) Continue Step2 and Step3 until the algorithm converges;
(5) Computing the embedding layer $\boldsymbol{\xi}$ and label layer $\boldsymbol{z}$ using Equation (9), and then construct target classifiers as described in Section 4.3.

4.3 Classifier Construction

After all the parameters are learned, we can construct classifiers for the target domain in two ways. The first way is directly to use the output of the second encoding layer. That is, for any instance $\boldsymbol{x}^{(t)}$ in the target domain, the output of the label layer $\boldsymbol{z}^{(t)} = f(\boldsymbol{W}_2\boldsymbol{\xi}^{(t)} + \boldsymbol{b}_2)$ can indicate the probabilities of $\boldsymbol{x}^{(t)}$, which class it belongs to. We choose the maximum probability and the corresponding label as the prediction. The second way is to apply standard classification algorithms, for example, logistic regression(LR) (Snyman 2005; Friedman and Rob 2010) to train a classifier on embedded source domain data. Then the classifier is applied to predict class labels for embedded target domain data. These two methods are denoted as TLDA_1 and TLDA_2, respectively.

4.4 Distance Measure for Distribution Difference

It is well known that measuring the distribution discrepancy between different domains is still a challenging problem. Here, we propose a new distance measure based on autoencoder and try to explain why the proposed double encoding-layer autoencoder framework can work well for transfer learning. Given the source domain data with $\boldsymbol{D}_s = \{\boldsymbol{x}_i^{(s)}\}|_{i=1}^{n_s}$, and target domain data $\boldsymbol{D}_t = \{\boldsymbol{x}_i^{(t)}\}|_{i=1}^{n_t}$, we first run the autoencoder code over the source domain data to derive the encoding and decoding weights, $\boldsymbol{W}_1^{(s)}, \boldsymbol{b}_1^{(s)}, \boldsymbol{W}_1'^{(s)}, \boldsymbol{b}_1'^{(s)}$ by the following optimization problem:

$$\min_{\boldsymbol{W}_1^{(s)}, \boldsymbol{b}_1^{(s)}, \boldsymbol{W}_1'^{(s)}, \boldsymbol{b}_1'^{(s)}} \sum_{i=1}^{n_s} \|\hat{\boldsymbol{x}}_i^{(s)} - \boldsymbol{x}_i^{(s)}\|^2. \tag{19}$$

Then the distance measure based on autoencoder is formally defined as

$$\text{AE} = \frac{1}{n_t} \sum_{i=1}^{n_t} \|\hat{\boldsymbol{x}}_i^{(t)} - \boldsymbol{x}_i^{(t)}\|^2, \tag{20}$$

where

$$\begin{aligned} \boldsymbol{\xi}^{(t)} &= f(\boldsymbol{W}_1^{(s)}\boldsymbol{x}^{(t)} + \boldsymbol{b}_1^{(s)}), \\ \hat{\boldsymbol{x}}^{(t)} &= f(\boldsymbol{W}_1'^{(s)}\boldsymbol{\xi}^{(t)} + \boldsymbol{b}_1'^{(s)}). \end{aligned} \tag{21}$$

A smaller value of AE indicates that target domain has a more similar distribution with source domain, whereas a bigger value of AE shows the larger gap of distribution mismath. If the target domain is equivalent to the source domain, then we have the smallest value of AE, in other word, the value of AE can be equivalent to 0 if the encoding and decoding weights are finely learnt. Thus, the proposed distance measure AE can be used to measure how close between the source

Table 2. Description of the ImageNet Dataset

	D_1	D_2	D_3	D_4	D_5
#positive instance	1,510	1,326	1,415	1,555	986
#negitive instance	1,427	1,427	1,427	1,427	1,427
#features	1,000	1,000	1,000	1,000	1,000

and target domains. It is acknowledged that if the distribution difference between source and target domains is small, the traditional supervised learning can perform well on the target domain, whereas the worse performance is obtained. The results in the experimental section coincide with this analysis. In our framework TLDA, the encoding and decoding weights are shared across different domains for knowledge transfer, and the source and target domain data are embedded to the common representations, which implies that autoencoder is also used to draw the distributions to be more similar.

5 EXPERIMENTAL EVALUATION

In this section, we first conduct systemic experiments on three real-world image datasets and one text dataset to show the effectiveness of the proposed framework. Three of the these four datasets are on binary classification, and the last one is on multi-class classification. Then, we empirically investigate why our framework can work well for transfer learning.

5.1 Datasets and Preprocessing

ImageNet Dataset[1] contains five domains, that is, D1 (*ambulance+scooter*), D2 (*taxi+scooter*), D3 (*jeep+scooter*), D4 (*minivan+scooter*), and D5 (*passenger car+scooter*). Data from different domains come from different categories, for example, *taxi* from D2 and *jeep* from D3; therefore, this dataset is proper for a transfer-learning study. To construct classification problems, we randomly choose two from the five domains, where one is considered as the source domain and the other is considered as the target domain. Therefore, we construct 20 (P_5^2) transfer-learning classification problems. Statistics of this dataset is shown in Table 2.

Corel Dataset[2] Zhuang et al. (2010) include two different top categories, *flower* and *traffic*. Each top category further consists of four subcategories. We use *flower* as positive instances and *traffic* as negative ones. To construct the transfer-learning classification problems, we randomly select one subcategory from *flower* and one from *traffic* as the source domain, and then choose another subcategory of *flower* and another one of *traffic* from the remaining subcategories to construct the target domain. In this way, we can construct 144 ($P_4^2 \cdot P_4^2$) transfer-learning classification problems.

Leaves Dataset Mallah and Orwell (2013) includes 100 plant species that are divided into 32 different genera, and each specie has 16 instances. We choose four genera with more than four plant species to construct four-class classification problems, and use 64 shape descriptor features to represent an instance. Each genus is regarded as a domain. Similar to the construction of ImageNet dataset, we can construct 12 (P_4^2) four-class classification problems.

Health Dataset is used for web content classification in the previous work (Banerjee and Scholz 2008). The web pages from the publicly available web resources are categorized under many categories, for example, health, shopping, science, programming, and music. In this article, we want to

[1] http://www.image-net.org/download-features.
[2] http://archive.ics.uci.edu/ml/datasets/Corel+Image+Features.

classify the web pages to belong to health or not for each web resource, and the data are crawled from four public web sites, including Wikipedia,[3] DMOZ,[4] Google,[5] and Del.icio.us.[6] If each web resource is regarded as a domain, then the data from different domains may have different distributions. Therefore, the constructed problems are suitable for transfer learning. Similarly with the construction method of ImageNet dataset, we can finally construct 12 (P_4^2) classification problems for four domains. In this dataset, each domain contains about 1,800 web pages, and the number of features is 6,045.

5.2 Baseline Methods

We compare our methods with the following baselines:

- —Logistic Regression (LR) (Friedman and Rob 2010): traditional supervised learning algorithm without transfer learning.
- —Transfer component analysis (TCA) (Pan et al. 2011): it aims at learning a low-dimensional representation for transfer learning. Here, we also use Logistic Regression as the basic classifier.
- —Marginalized Stacked Denoising Autoencoders (mSDA) (Chen et al. 2012): this is a transfer-learning algorithms based on stacked autoencoders.

Since TLDA considers two distance measures, that is, KL and AE, so we further compare the one without considering KL divergence, that is, $\alpha = 0$. This method is indeed a special case of TLDA, denoted as DA.

Implementation Details: After some preliminary experiments, we set $\alpha = 0.5$, $\beta = 0.5$, $\gamma = 0.00001$, and $k = 10$ for the ImageNet and Corel datasets, $\alpha = 0.5$, $\beta = 0.05$, $k = 5$, and $\gamma = 0.0001$ for the Leaves dataset, while $\alpha = 10$, $\beta = 10$, $k = 20$, and $\gamma = 0.005$ for the Health dataset. For mSDA, we use the authors' source code[7] and adopt the default parameters as reported in Chen et al. (2012). For TCA, the number of latent dimensions is carefully tuned, for example, for the Corel dataset, the number is sampled from [10, 80] with interval 10, and its best results are reported. Note that, the sigmoid function is used as the nonlinear activation function in autoencoder, and the range of output values are between 0 and 1. Therefore, the samples of four datasets are normalized in this way $x = \frac{x}{\sqrt{x^\top x}}$.

5.3 Experimental Results

All the results of these four datasets are shown in Figure 2 and Table 3. Figure 2 shows the detailed results over the 20 classification problems on the ImageNet dataset, in which x axis represents the index of the problems, and y axis represents the corresponding accuracy. All the problems are sorted by the increasing order of the accuracy from LR for clear comparison. From the figure, we have the following insightful observations:

- —TLDA is significantly better than LR on all datasets, which indicates the efficiency of our proposed transfer-learning framework.
- —TLDA performs better than TCA, which shows the superiority of applying double encoding-layer autoencoder to learn a good representation for transfer learning. TLDA also

[3] http://www.wikipedia.org/.
[4] http://www.dmoz.org/.
[5] http://www.google.com.
[6] http://del.icio.us/.
[7] http://www.cse.wustl.edu/mchen/.

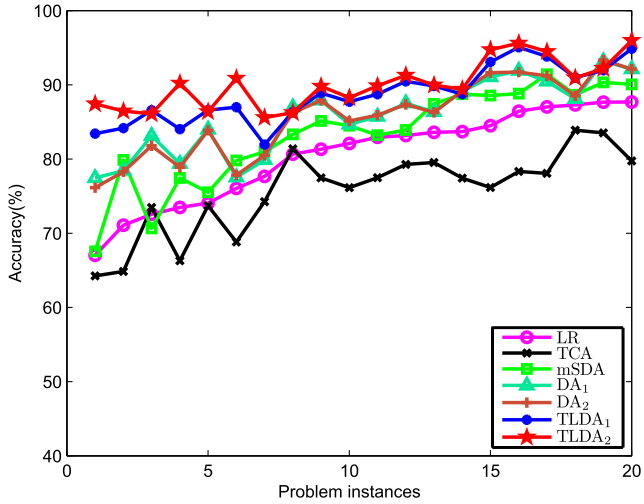


Fig. 2. Classification accuracy on the ImageNet dataset.

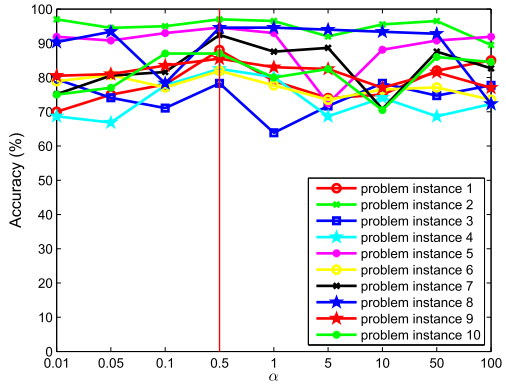
Table 3. Average Results (%) on Four Datasets

	LR	TCA	mSDA	DA_1	DA_2	$TLDA_1$	$TLDA_2$
	ImageNet Dataset						
Left	67.0	64.3	67.6	77.4	76.1	83.4	**87.4**
Right	81.2	76.3	84.1	86.2	86.2	89.0	**90.2**
Total	80.5	75.7	83.3	85.7	85.7	88.7	**90.1**
	Corel Dataset						
Left	61.7	65.4	70.5	64.6	65.8	71.1	**74.0**
Right	80.1	82.0	75.4	80.1	80.3	**83.2**	83.0
Total	74.8	76.5	74.0	75.6	76.1	79.6	**80.4**
	Leaves Dataset						
Left	51.9	**65.9**	47.2	63.4	59.1	64.1	57.8
Right	75.0	89.8	59.4	58.6	57.8	**91.4**	89.8
Total	55.7	**69.9**	49.2	62.6	58.9	68.6	63.2
	Health Dataset						
Left	62.6	68.7	61.5	60.4	60.4	**70.4**	**70.4**
Right	80.1	74.5	**85.0**	81.7	81.6	84.7	84.7
Total	74.3	72.3	77.2	74.6	74.6	**79.9**	**79.9**

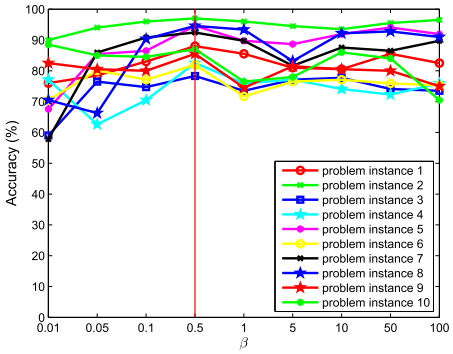
outperforms mSDA, which indicates the effectiveness of encoding label information from source domain.

—TLDA is better than DA, which indicates that TLDA can benefit from taking advantage of both distance measures, that is, KL and AE. DA is also better than LR, which shows the success of using deep learning for transfer learning.

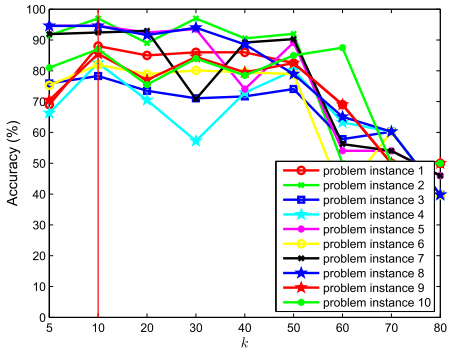
—LR performs slightly worse than mSDA, even better than TCA sometimes. This may be because on the constructed cross-domain classification problems, it is not easy to make knowledge transfer successfully. This observation again validates the effectiveness of our method.



(a) The Parameter Influence of α



(b) The Parameter Influence of β



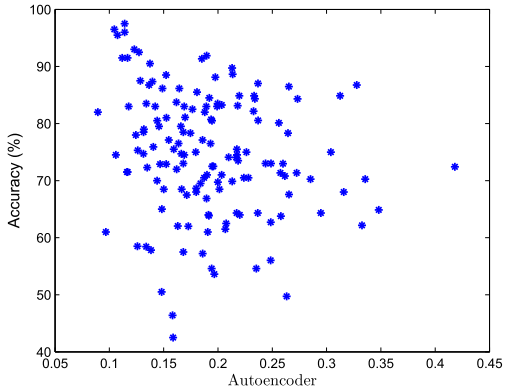
(c) The Parameter Influence of k

Fig. 3. The study of parameter influence on TLRA_1.

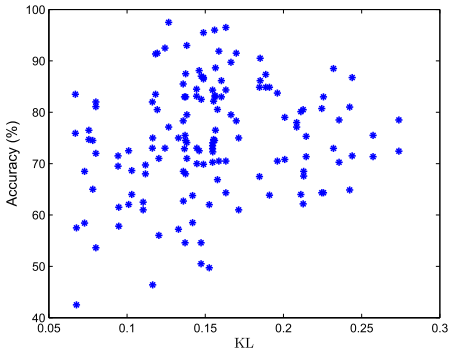
We also divide the constructed problems into two groups: the first group consists of problems on which the classification accuracy of LR is lower than 70%, and the rest problems are considered as a second group. The lower of classification accuracy of LR in some certain indicates the higher degree of the difficulty in knowledge transfer. The averaged accuracy of these two group as well as the averaged accuracy over all problems on these four datasets are reported in Table 3, denoted as *Left*, *Right*, and *Total*, respectively. We can find that the proposed methods perform better than all the compared algorithms on both groups of problems, except for that on the Leaves dataset, the performance of TLDA_1 is comparable with that of TCA. Also, in general, we observe the much larger margin of accuracy improvement of TLDA on all datasets when the accuracy from LR is lower than 70%, which indicates the stronger transfer ability of our model.

5.4 Why TLDA Can Work for Transfer Learning

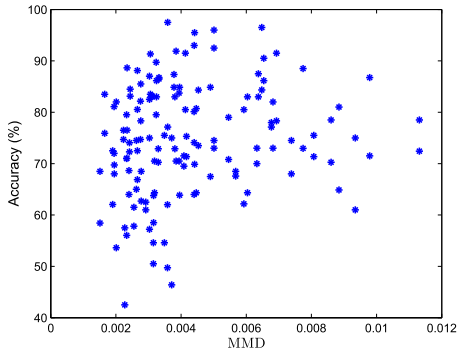
As mentioned earlier, the classification accuracy of LR can indicate the degree of transfer difficulty. In other words, higher (or lower) accuracy of LR indicates easier (or harder) to make transfer. Here, we empirically investigate the relationship between the accuracy of LR on the Corel dataset and AE, and we compare AE with the other two measures KL and MMD. Note that here the values of KL and MMD are computed based on the original feature space. The detailed results are shown in Figure 4. It is obviously observed that AE can better reflect the degree of transfer difficulty and



(a) Auc of LR vs. AE



(b) Auc of LR vs. KL



(c) Auc of LR vs. MMD

Fig. 4. The Relationship between the Accuracy (%) of LR and three distance measures on the Corel dataset.

Table 4. The Correlation Coefficients between the Accuracy of LR and Three Distance Measures on Four Datasets

	AE	KL	MMD
ImageNet Dataset	**−0.1365**	0.1312	0.2041
Corel Dataset	**−0.2009**	0.1796	0.1616
Leaves Dataset	**−0.5074**	−0.3339	−0.3751
Health Dataset	**−0.1665**	−0.1600	−0.0521

have stronger correlation with the performance of LR, and AE significantly outperforms both KL and MMD.

To quantitatively show the effectiveness of AE, the correlation coefficients between the accuracy of LR and three distance measures on four datasets are recorded in Table 4. (The value range of correlation coefficient is $[-1, 1]$, and minus value means negative correlation.) Since lower values of three measures indicate the transfer-learning problems easier to make transfer, and higher accuracies of LR can be obtained. Therefore, lower value of correlation coefficient is better. These results in Table 4 again validate the effectiveness of AE. In our model TLDA, both source and target domains share the same encoding and decoding weights, which means that AE is also adopted as

well as KL to enforce their distributions more similar. We conjecture this consideration of both AE and KL lead to the success of our model.

5.5 Parameter Sensitivity

In this section, we investigate the influence of the parameters α, β, and k in the objective Equation (7). In this experiment, when tuning one parameter, the values of the rest two are fixed. Specifically, α and β are sampled from $\{0.01, 0.05, 0.1, 0.5, 1, 5, 10, 50, 100\}$, and k is selected from $10, 20, \ldots, 80$. We select 10 of the 144 problems on the Corel dataset for experiment and report the results in Figure 3. From the figure, we can observe that the performance of TLDA_1 is relatively stable to the selection of α and β, while it decreases dramatically when the value of k is large. Thus, we set $\alpha = 0.5$, $\beta = 0.5$, and $k = 10$ to achieve good and stable results for the ImageNet and Corel datasets.

6 CONCLUSION

In this article, we adapt the double encoding-layer autoencoder to transfer learning and propose a supervised representation learning framework. In this framework, the well-known representation learning model autoencoder is considered, and we extend it to a deeper architecture. Indeed, there are two layers for encoding, one is for embedding, where we impose the KL divergence constraints to draw the two distributions of embedded source and target domains similar. The other is label layer, by which we can easily encode the label information from source domain. A series of experiments conducted on three real-world image datasets and one text dataset demonstrate the effectiveness of the proposed methods. Furthermore, to empirically analyze why TLDA can work well for transfer learning, we propose a new distance measure based on autoencoder, which is validated to better characterize the degree of transfer difficulty. We conjecture the success of our model may be owed to the consideration of both AE and KL.

REFERENCES

Somnath Banerjee and Martin Scholz. 2008. Leveraging web 2.0 sources for web content classification. In *Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, Volume 1*. IEEE Computer Society, 300–306.

Yoshua Bengio. 2009. Learning deep architectures for AI. *Found. Trends Mach. Learn.* 2, 1 (2009), 1–127.

John Blitzer, Ryan McDonald, and Fernando Pereira. 2006. Domain adaptation with structural correspondence learning. In *Proceedings of the 2006 Conference on Empirical Methods on Natural Language Processing*. 120–128.

Minmin Chen, Zhixiang Eddie Xu, Kilian Q. Weinberger, and Fei Sha. 2012. Marginalized denoising autoencoders for domain adaptation. In *Proceedings of the 29th International Conference on Machine Learning*.

Koby Crammer, Mark Dredze, and Fernando Pereira. 2012. Confidence-weighted linear classification for text categorization. *J. Mach. Learn. Res.* 13, 1 (2012), 1891–1926.

W. Y. Dai, G. R. Xue, Q. Yang, and Y. Yu. 2007a. Co-clustering based classification for out-of-domain documents. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

W. Y. Dai, Q. Yang, G. R. Xue, and Y. Yu. 2007b. Boosting for transfer learning. In *Proceedings of the 24th International Conference on Machine Learning*.

Trevor Hastie, Friedman Jerome, and Rob Tibshirani. 2010. Regularization paths for generalized linear models via coordinate descent. *J. Stat. Softw.* 33, 1 (2010), 1.

Yaroslav Ganin and Victor Lempitsky. 2015. Unsupervised domain adaptation by backpropagation. In *Proceedings of the 32nd International Conference on Machine Learning (ICML'15)*. 1180–1189.

J. Gao, W. Fan, J. Jiang, and J. W. Han. 2008. Knowledge transfer via multiple model local structure mapping. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Mingming Gong, Kun Zhang, Tongliang Liu, Dacheng Tao, Clark Glymour, and Bernhard Schölkopf. 2016. Domain adaptation with conditional transferable components. In *Proceedings of the 33rd International Conference on Machine Learning*. 2839–2848.

Jing Jiang and Chengxiang Zhai. 2007. Instance weighting for domain adaptation in NLP. In *Proceedings of the 2007 Conference of the Association for Computational Linguistics*. 264–271.

Ivor Tsang, Joey Tianyi Zhou, Sinno Jialin Pan, and Yan Yan. 2014. Hybrid heterogeneous transfer learning through deep learning. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence*. 2213–2220.

Solomon Kullback. 1987. Letter to the editor: The Kullback-Leibler distance (1987).

Andrew R. Liddle, Pia Mukherjee, and David Parkinson. 2010. Model selection and multi-model inference. *Bayes. Meth. Cosmol.* 1 (2010), 79.

Tongliang Liu, Dacheng Tao, Mingli Song, and Stephen J. Maybank. 2017. Algorithm-dependent generalization bounds for multi-task learning. *IEEE Trans. Pattern Anal. Mach. Intell.* 39, 2 (2017), 227–241.

Mingsheng Long, Yue Cao, Jianmin Wang, and Michael I. Jordan. 2015. Learning transferable features with deep adaptation networks. In *Proceedings of the International Machine Learning Society (ICML'15)*. 97–105.

Mingsheng Long, Jianmin Wang, Yue Cao, Jianguang Sun, and Philip S. Yu. 2016. Deep learning of transferable representation for scalable domain adaptation. *IEEE Trans. Knowl. Data Min.* 28, 8 (2016), 2027–2040.

Yong Luo, Tongliang Liu, Dacheng Tao, and Chao Xu. 2014. Decomposition-based transfer distance metric learning for image classification. *IEEE Trans. Image Process.* 23, 9 (2014), 3789–3801.

Cope Mallah and Orwell. 2013. Plant leaf classification using probabilistic integration of shape, texture and margin features. *Signal Processing, Pattern Recognition and Applications* (2013).

S. J. Pan, J. T. Kwok, and Q. Yang. 2008. Transfer learning via dimensionality reduction. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence*.

Sinno Jialin Pan, Ivor W. Tsang, James T. Kwok, Qiang Yang, and others. 2011. Domain adaptation via transfer component analysis. *IEEE Trans. Neural Networks* 22, 2 (2011), 199–210.

Sinno Jialin Pan and Qiang Yang. 2010. A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.* 22, 10 (2010), 1345–1359.

Christopher Poultney, Sumit Chopra, Yann L. Cun, and others. 2006. Efficient learning of sparse representations with an energy-based model. In *Advances in Neural Information Processing Systems*. 1137–1144.

Si Si, Dacheng Tao, and Bo Geng. 2010. Bregman divergence-based regularization for transfer subspace learning. *IEEE Trans. Knowl. Data Eng.* 22, 7 (2010), 929–942.

Jan Snyman. 2005. *Practical Mathematical Optimization: An Introduction to Basic Optimization Theory and Classical and New Gradient-based Algorithms*. Vol. 97. Springer Science & Business Media.

Eric Tzeng, Judy Hoffman, Trevor Darrell, and Kate Saenko. 2015. Simultaneous deep transfer across domains and tasks. In *Proceedings of the IEEE International Conference on Computer Vision*. 4068–4076.

Bengio Vincent Larochelle and Manzagol. 2008. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th International Conference on Machine Learning*. ACM, 1096–1103.

Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. 2010. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *J. Mach. Learn. Res.* 11 (2010), 3371–3408.

Antoine Xavier and Bengio. 2011. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of the 28th International Conference on Machine Learning*. 513–520.

D. K. Xing, W. Y. Dai, G. R. Xue, and Y. Yu. 2007. Bridged refinement for transfer learning. In *Proceedings of the 10th Pacific Asia Knowledge Discovery and Data Mining*.

Fuzhen Zhuang, Xiaohu Cheng, Ping Luo, Sinno Jialin Pan, and Qing He. 2015. Supervised representation learning: Transfer learning with deep autoencoders. In *Proceedings of 24th International Joint Conference on Artificial Intelligence*. 4119–4125.

Fuzhen Zhuang, Xiaohu Cheng, Sinno Jialin Pan, Wenchao Yu, Qing He, and Zhongzhi Shi. 2014. Transfer learning with multiple sources via consensus regularized autoencoders. In *Machine Learning and Knowledge Discovery in Databases*. Springer, 417–431.

Fuzhen Zhuang, Ping Luo, Hui Xiong, Yuhong Xiong, Qing He, and Zhongzhi Shi. 2010. Cross-domain learning from multiple sources: A consensus regularization perspective. *IEEE Trans. Knowl. Data Eng.* 22, 12 (2010), 1664–1678.

Received January 2017; revised June 2017; accepted June 2017