Journal of Computers Vol. 34 No. 5, October 2023, pp. 73-85
doi: 10.53106/199115992023103405006

Improving Unsupervised Domain Adaptation via Multiple Adversarial Learning

Yu-Dong Cao, Shuang-Jiang Hang*, Xu Jia

School of Electronics and Information Engineering, Liaoning University of Technology, Jinzhou city 121001, Liaoning Province, China

caoyd@lnut.edu.cn, 18653485406@163.com, jiaxu@lnut.edu.cn

Received 5 September 2022; Revised 23 January 2023; Accepted 13 March 2023

Abstract. Most machine learning methods assume the training and test sets to be independent and have identical distributions. However, this assumption does not always hold true in practical applications. Direct training usually induces poor performance if the training and test data have distribution shifts. To address this issue, a three-part model based on using a feature extractor, a classifier, and several domain discriminators is adopted herein. This unsupervised domain adaptation model is based on multiple adversarial learning with samples of different importance. A deep neural network is used for supervised classification learning of the source domain. Numerous adversarial networks are used to constitute the domain discriminators to align each category in the source and target domains and effectively transfer knowledge from the source domain to the target domain. Triplet loss functions—classification loss, label credibility loss, and discrimination loss—are presented to further optimize the model parameters. First, the label similarity metric is designed between the target and source domains data. Second, a credibility loss function is proposed to obtain an accurate label for the unlabeled data of the target domain under training iterations. Finally, a discrimination loss is designed for multiple adversarial domain discriminators to fully utilize the unlabeled data in the target domain during training. The discrimination loss function uses predicted label probabilities as dynamic weights for the train data. The proposed method is compared with mainstream domain adaptation approaches on four public datasets: Office-31, MNIST, USPS, and SVHN. Experimental results show that the proposed method can perform well in the target domain and improve generalization performance of the model.

Keywords: domain adaptation, transfer learning, deep neural network, adversarial learning, loss function

1 Introduction

When data distribution shifts between the training and test sets, many machine learning models cannot obtain good prediction results on the test set. Thus, it is challenging to recollect and label the training data with the same distribution as the test set. Domain adaptation, a special case of transfer learning technology, can solve the data distribution shift between the source (training set) and target (test set) domains. Further, deep learning technology is advancing; hence, knowledge transfer based on deep neural networks is attracting increasing research attention [1-2]. Domain adaptation, which learns a model by minimizing the difference between domain distributions, can apply the source domain knowledge to an unlabeled target domain. Therefore, further study on deep domain adaptation methods is significant [3]. These methods primarily include discrepancy-based domain adaptation [4] and adversarial-based domain adaptation [5]. Discrepancy-based domain adaptation adopts multiple adaptation layers and metric criteria to complete the knowledge transfer of the source and target domain data. Examples of this approach include deep domain confusion (DDC) [6], joint adaptation networks (JANs) [7], deep transfer networks [8], deep transfer low-rank coding (DTLC) for cross-domain learning [9], and weighted maximum mean discrepancy (MMD) for unsupervised domain adaptation [10]. The DDC network adopts a single linear kernel to one layer to minimize MMD [6]. However, the DDC network has shallow layers with only one adaptive layer, inducing a weak domain-invariant feature representation. The JAN model learned a transfer network by aligning the joint distributions of multiple domain-specific layers across domains with the proposed joint maximum mean discrepancy (JMDD) criterion. However, the JMMD estimate is too complex. The above deep domain adaptation methods add MMD to the cost function as a distance metric between the source domain and target domain features. The domain-invariant features between the source and target domains are learned by optimizing the source domain classification error and the MMD loss. The above methods require the artificial design across domains

* Corresponding Author

to map the data of source and target domains to the same feature space. Furthermore, adversarial-based transfer learning realizes knowledge transfer using the Generative Adversarial Nets framework.

Adversarial learning has been successfully applied to deep neural networks to learn transferable features by reducing distribution discrepancy across domains [11]. One is that the discriminator acts on the total data space, such as in coupled generative adversarial networks (CoGANs) [12] and pixel-level domain transfer with cross-domain consistency (CrDoCo) [13]. The other is that the discriminator acts on the feature space, such as in the domain-adversarial neural network (DANN) [14], domain separation networks (DSNs) [15], adversarial discriminative domain adaptation (ADDA) [16], and deep transfer metric learning [17]. Liu et al. [12] designed CoGANs that learned joint distribution with the data drawn from marginal distributions by enforcing a weight-sharing constraint. Chen et al. [13] introduced a cross-domain consistency loss function that strengthened consistent predictions for domain adaption DANN [14] augmented adaptation behavior using a new gradient reversal layer and a few standard layers in adversarial learning. Bousmalis et al. [15] presented a DSN for domain-invariant feature learning. ADDA [16] learned the discriminative mapping of target images to the target encoder by fooling the domain discriminator that distinguishes the encoded target images from the source examples. Hu et al. [17] presented a deep transfer metric learning method to learn a set of hierarchical nonlinear transformations by maximizing the interclass difference and minimizing the difference and the distribution divergence between the source and the target domains. The above adversarial-based methods can extract the domain-invariance feature with interclass discrimination and avoid designing the complex distance measurement. However, these methods primarily align the population distributions in the source and target domains without considering the complicated structures underlying the data distributions [18]. Therefore, the source and target domains could be combined with comprehensive data. Moreover, the discriminative structures are confusing, causing false alignment of the corresponding biased structures of different distributions (Fig. 1).

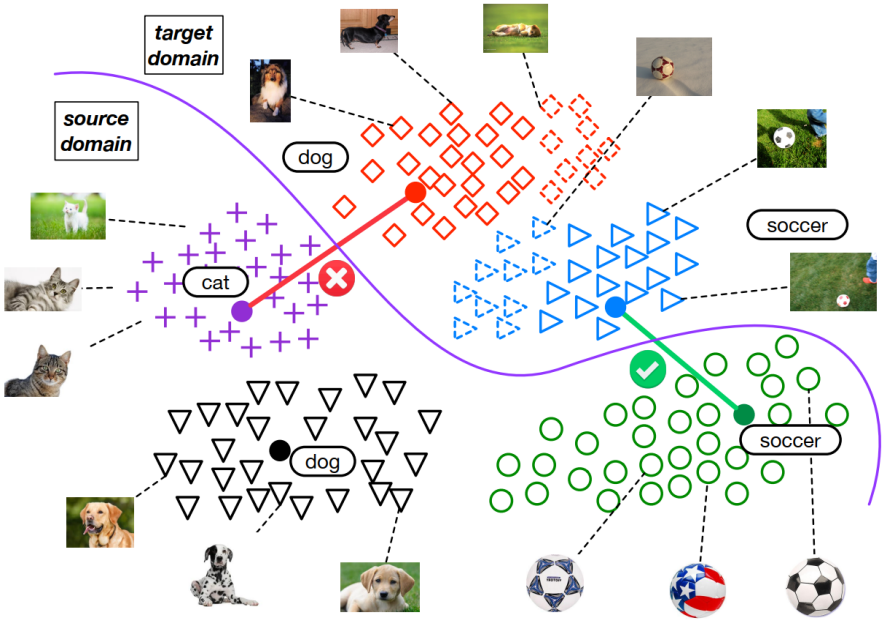


Fig. 1. The complexity of the domain adaptation

(The discriminative structures are confused or falsely aligned across domains. For example, the source category cat is falsely aligned with the target category dog, causing a wrong final classification (from [19])).

The classifier trained directly on the source domain cannot suitably distinguish the target domain data. Moreover, simply aligning the distribution of the overall data is insufficient and can induce negative migration. Based on Pei et al. [19], we present an unsupervised domain adaptation method with multiple adversarial learning using triplet loss functions for model parameter optimization. Compared with traditional domain adaptation algorithms, our method considers each category distribution between the source and target domains, compensat-

ing for the limitation of the traditional domain adaptation approaches in only focusing on the general alignment of the source and target domains.

This study and its features are summarized as follows: (1) a label-prediction method for the target domain data is proposed to align the distribution of data from the same categories across domains. We designed the label similarity metric between domains to obtain accurate labels of the target domain data. (2) Triplet loss functions with enhanced adversarial learning are designed to optimize model parameters. Notably, the label credibility loss function improves the reliability of trained data in the target domain. (3) Numerous adversarial networks are adopted with weighted loss functions of the discriminators to align the same categories across domains.

2 Proposed Unsupervised Domain Adaptation Scheme

Here, we present a synopsis of our method and describe the designed loss functions to optimize the parameters of the model performing cross-domain consistency. Typically, there is a difference in the feature distribution between the source and target domains. However, the domain discriminator of a single adversarial network can only reduce the disparity in the overall feature distribution between the source and target domains and cannot ensure that the feature discrepancy between categories is sufficiently limited. Therefore, accurately aligning the features of each category between the source and target domains is challenging.

2.1 Method Overview

Domain adaptation solutions are widely used in machine learning problems that lack labeled data. Majority of the current adversarial-based domain adaptation methods reduce the disparity in the feature distribution between the source and target domains. However, reducing the distribution discrepancy between the same categories across different domains has increased significance. Therefore, we adopted multiple adversarial networks as domain discriminators to align the feature distribution of the same categories across domains.

The target domain data lacks label information in the unsupervised domain adaptation; therefore, each discriminator is accountable for matching the target and source domain. data related to the category k. Hence, aligning corresponding categories between the source and target domains is important. Our solution is to first learn the deep domain-invariant feature and then gradually generate the accurate label for target domain data following many training epochs. The data distribution of each category is aligned using adversarial learning. Fig. 2 shows the structure of our model, which includes the transfer feature extractor $\boldsymbol{f} = G_f(\boldsymbol{x})$ with the parameter θ_f , the adaptive classifier $\hat{\boldsymbol{y}} = G_c(\boldsymbol{f})$ with the parameter θ_c , and multiple discriminators $\boldsymbol{d}_j = G_{d_j}(\boldsymbol{f})$ with the parameter θ_{d_j}, where $j \in \{1, 2, ..., m\}$. $\hat{\boldsymbol{y}}$ is the predicted label using the probability distribution over the label space. Our model makes the source and target domains indistinguishable using various domain discriminators, each responsible for matching the source and target data associated with the same category, reducing the shift between the data distributions of the distinctive objects.

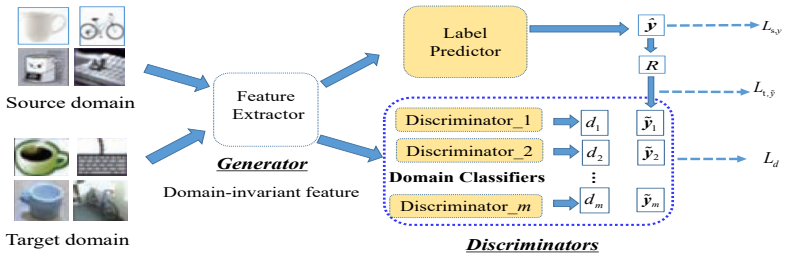


Fig. 2. Model architecture

Let $\boldsymbol{D}_s = \{(\boldsymbol{x}_i^s, \boldsymbol{y}_i^s) \mid i = 1, ..., n_s\}$ be samples in the source domain, where, $\boldsymbol{x}_i^s$ represents the i^{th} sample, $\boldsymbol{y}_i^s$ is one hot code of the category label in the source domain, and n_s is the total number of samples in the source domain. We denote $\boldsymbol{D}_t = \{\boldsymbol{x}_i^t \mid i = 1, ..., n_t\}$ as samples in the target domain, where $\boldsymbol{x}_i^t$ represents the unlabeled samples in the target domain, and n_t represents the total number of samples in the target domain. The source and target domains data are assumed to come from different joint distributions $P(\boldsymbol{X}^s, \boldsymbol{Y}^s)$ and $Q(\boldsymbol{X}^t, \boldsymbol{Y}^t)$, where $P \neq Q$ $\boldsymbol{X}^s$ and $\boldsymbol{X}^t$ are the source and target domain sample sets, respectively, and $\boldsymbol{Y}^s$ and $\boldsymbol{Y}^t$ are the label sets. Moreover, the loss function needs to be designed to optimize the model parameters.

2.2 Loss Function Design

The loss function plays a crucial role when optimizing the model parameters. Triple loss functions are designed for our model on the source and target domain data.

Classification Loss. The feature extractor $G_f(\cdot)$ and the classifier $G_c(\cdot)$ are trained on the source domain through supervised learning. The designed loss function of the classifier is as follows:

$$L_{s,y} = \frac{1}{n_{\boldsymbol{D}_{s,batch}}} \sum_{\boldsymbol{x}_i \in \boldsymbol{D}_{s,batch}} L_{\text{EC}}\left(G_{\text{c}}\left(G_{\text{f}}\left(\boldsymbol{x}_i\right)\right), \boldsymbol{y}_i\right) \quad , \tag{1}$$

where L_{EC} is the cross-entropy loss function, $\boldsymbol{D}_{s,batch}$ is the sample set of the source domain per epoch during training, and $n_{\boldsymbol{D}_{s,batch}}$ is the number of samples. Following each epoch, the classifier $G_c(\cdot)$ is used to classify the samples in the source and target domains and subsequently obtain the probability distribution of the samples of all categories.

Label Credibility Loss. Unlabeled samples exist in the target domain, and the predicted label $\hat{\boldsymbol{y}}_i^s$ in the source domain is not exactly consistent with the actual label $\boldsymbol{y}_i^s$ in the original training epochs. Here, we propose a similarity metric between the labels to obtain the similarity between target domain samples and the samples of each category in the source domain. This metric expression is as follows:

$$\begin{aligned} R_k\left(\boldsymbol{x}_j^{\text{t}}\right) &= max \frac{cov\left(\hat{\boldsymbol{y}}_i^s, \boldsymbol{y}_j^{\text{t}}\right)}{\sqrt{D\left(\hat{\boldsymbol{y}}_i^s\right)}\sqrt{D\left(\boldsymbol{y}_j^{\text{t}}\right)}} \\ &\text{s.t.} \quad y_{i,k}^s = 1 \end{aligned} \tag{2}$$

Here, $\hat{\boldsymbol{y}}_i^s$ and $\hat{\boldsymbol{y}}_j^t$ are the labels of the source domain sample and the target domain sample predicted by the classifier, respectively. $R_k(\boldsymbol{x}_j^t)$ is the probability value that the sample $\boldsymbol{x}_j^t$ belongs to category c in the source domain, cov represents the covariance, and D is the variance. The constraint condition $\boldsymbol{y}_{i,k}^s = 1$ indicates that the predicted label of the source domain sample is close to its real label to ensure good classification. Furthermore, the closer $R_k(\boldsymbol{x}_j^t)$ is to 1, the higher the probability that $\boldsymbol{x}_j^t$ is in category k; the closer $R_k(\boldsymbol{x}_j^t)$ is to 0, the lower the likelihood that $\boldsymbol{x}_j^t$ belongs to category k. The similarity between $\boldsymbol{x}_j^t$ and samples of each category is calculated, and the accurately predicted label of $\boldsymbol{x}_j^t$ in the target domain is obtained as follows:

$$\tilde{\boldsymbol{y}}_j^{\text{t}} = \frac{1}{\sum_{k=1}^{m} R_k\left(\boldsymbol{x}_j^{\text{t}}\right)} \left[R_1\left(\boldsymbol{x}_j^{\text{t}}\right) \quad R_2\left(\boldsymbol{x}_j^{\text{t}}\right) \quad \ldots \quad R_m\left(\boldsymbol{x}_j^{\text{t}}\right) \right], \tag{3}$$

where m represents the number of categories.

If an element of the predicted label in the target domain is close to 1, and other elements are close to 0 after many training epochs, this means that this sample is similar to a particular sample in the source domain. Therefore, the target domain sample can be accurately classified, which is considered creditable during training. The credibility loss function in the target domain is as follows:

$$L_{\mathrm{t},\tilde{y}}=\frac{1}{n_{\boldsymbol{D}_{\mathrm{t},batch}}}\sum_{\boldsymbol{x}_i\in\boldsymbol{D}_{batch}}L_{\mathrm{E}}\left(\tilde{\boldsymbol{y}}_i^{\mathrm{t}}\right), \tag{4}$$

where L_E is the entropy function, $\boldsymbol{D}_{t,batch}$ is the sample set of the target domain per epoch, and $n_{\boldsymbol{D}_{t,batch}}$ is the number of training samples.

Discrimination Loss. The domain discriminator of each category corresponds to a loss function. The labeling of the sample affects the optimization performance of the model. For example, the k^{th} category samples positively affect the k^{th} domain discriminator and negatively impact other domain discriminators during learning. The loss function weighted by the predicted label of the sample is as follows:

$$L_{\mathrm{d}}=\frac{1}{n_{\boldsymbol{D}_{batch}}}\sum_{\boldsymbol{x}_i\in\boldsymbol{D}_{batch}}\sum_{j=1}^{m}\tilde{y}_{i,j}L_{\mathrm{EC}}\left(G_{\mathrm{d}_j}\left(G_{\mathrm{c}}\left(\boldsymbol{x}_i\right)\right),\boldsymbol{d}_i\right). \tag{5}$$

Here, L_{EC} is the cross-entropy loss function. If the training data $\boldsymbol{x}_i$ is the source domain sample, the real label vector y_i is used as its weight vector. If the training data $\boldsymbol{x}_i$ is the target domain sample, the predicted label vector is used as its weight vector, which is calculated from Equation (3). If the training samples are in the source domain, their label $\hat{\boldsymbol{y}}$ is taken as its actual label $\boldsymbol{y}$, which acts as the weight of the loss function. $\boldsymbol{D}_{batch}$ is the set of samples from the source and target domains, and $n_{\boldsymbol{D}_{batch}}$ is the number of samples in this set. $\boldsymbol{d}_i$ is the domain label, $\boldsymbol{d}_i$ = $[0\ 1]^T$ is the label of the source domain, and is the label of the target domain. The above strategy shares a similar idea with the attention mechanism [19]. In summary, the overall objective function is constructed as follows:

$$L\left(\boldsymbol{\theta}_{\mathrm{f}},\boldsymbol{\theta}_{\mathrm{y}},\boldsymbol{\theta}_{\mathrm{d}}\right)=L_{\mathrm{s},y}+\lambda_1 L_{\mathrm{d}}+\lambda_2 L_{\mathrm{t},\tilde{y}}, \tag{6}$$

where, λ_1 and λ_2 are the trade-off parameters (see Section 4.2 for how to determine their values). θ_f, θ_c, and θ_d are the parameters of the feature extractor $G_f(\cdot)$, classifier $G_c(\cdot)$, and domain discriminator $G_d(\cdot)$, respectively. An alternating optimization strategy is adopted here. We fix the parameter θ_d, and then update the parameters θ_f and θ_c:

$$\boldsymbol{\theta}_{\mathrm{f}}^{*},\boldsymbol{\theta}_{\mathrm{c}}^{*}=\arg\min_{\boldsymbol{\theta}_{\mathrm{f}},\boldsymbol{\theta}_{\mathrm{c}}}L\left(\boldsymbol{\theta}_{\mathrm{f}},\boldsymbol{\theta}_{\mathrm{c}},\boldsymbol{\theta}_{\mathrm{d}}\right). \tag{7}$$

Similarly, we fix the parameters θ_f and θ_c, and then sequentially update the parameter θ_d for m discriminators:

$$\boldsymbol{\theta}_{\mathrm{d}}^{*}=\arg\max_{\boldsymbol{\theta}_{\mathrm{d}}}L\left(\boldsymbol{\theta}_{\mathrm{f}},\boldsymbol{\theta}_{\mathrm{c}},\boldsymbol{\theta}_{\mathrm{d}}\right). \tag{8}$$

The minibatch gradient descent method optimizes the model parameters through multiple iterations. The training process is summarized as follows.

Input: $\boldsymbol{D}_s = \{(x_i^s, y_i^s)\}_{i=1}^{ns}$, $\boldsymbol{D}_t = \{(x_i^t)\}_{i=1}^{nt}$, and balance the parameters λ_1 and λ_2, the number of repeating n_{epoch}, batch size n_{epoch}, k = 0, and initial learning rate $\mu^{(0)}$.

Step 1. Initialize $\theta_f^{(0)}$, $\theta_c^{(0)}$, and $\theta_d^{(0)}$.

Step 2. ***for*** i = 1, 2, ..., n_{epoch}, ***do***

Step 3. ***for*** $j = 1, 2, ..., \left\lceil \frac{n_s + n_t}{n_{batch}} \right\rceil$, ***do***

Step 4. Update $\boldsymbol{\theta}_f$ by calculating $\frac{L^{(k)}}{\theta_f}$ and $\mu^{(k)}$ with the random batch sampling using Equations (6) and (11):

$$\theta_f^{(k+1)} \leftarrow \theta_f^{(k)} - \mu^{(k)} \frac{\partial L^{(k)}}{\partial \theta_f}$$

Step 5. Update $\boldsymbol{\theta}_c$ by calculating $\frac{\partial L^{(c)}}{\partial \theta_c}$ and $\mu^{(k)}$ with the random batch sampling using Equations (6) and (11):

$$\theta_c^{(k+1)} \leftarrow \theta_c^{(k)} - \mu^{(k)} \frac{\partial L^{(k)}}{\partial \theta_c}$$

Step 6. Update $\boldsymbol{\theta}_d$ by calculating $\frac{\partial L^{(k)}}{\partial \theta_d}$ and $\mu^{(k)}$ with the random batch sampling using Equations (6) and (11):

$$\theta_d^{(k+1)} \leftarrow \theta_d^{(k)} - \mu^{(k)} \frac{\partial L^{(k)}}{\partial \theta_d}$$

Step 7. $k = k + 1$

Step 8. ***end for***

Step 9. ***end for***

Output: The optimized parameters $\hat{\boldsymbol{\theta}}_c$, $\hat{\boldsymbol{\theta}}_f$, and $\hat{\boldsymbol{\theta}}_d$.

3 Experimental Results and Analysis

Here, we evaluated the proposed method using state-of-the-art deep domain adaptation baselines on four datasets. The following elaborates on the detailed experiments and results.

3.1 Dataset

We adopted four widely used datasets to construct a standard benchmark for domain adaptation: Office-31 [20], Modified National Institute of Standards and Technology (MNIST) dataset [21], the US Postal (USPS) handwritten digit dataset [22], and the Street View House Numbers (SVHN) dataset [23].

Office-31 dataset: This dataset contains 31 categories and 4,652 images from three distinct domains: Amazon (A), Webcam (W), and DSLR (D). A contains 2,817 images downloaded from Amazon.com with 31 categories. W includes 795 images taken by the web in office settings. D contains 498 images taken by digital SLR cameras in office settings. We used all domain combinations and established six transfer tasks: W → A, A → W, D → W, W → D, A → D, and D → A. Some samples in Office-31 are shown in Fig. 3.

(a) Samples in the Amazon domain

(b) Samples in the Webcam domain

(c) Samples in the DSLR domain

Fig. 3. Samples in the Office-31 dataset

MNIST dataset: This dataset contains training and test sets. Each set includes 10 numeric categories with a total of 70,000 images. The samples are grayscale images with 28×28 pixels. Examples are shown in Fig. 4.

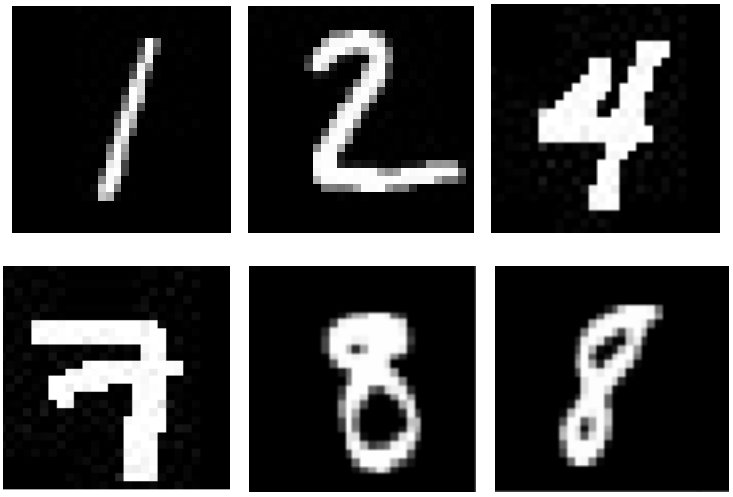


Fig. 4. Samples in the MNIST dataset

USPS dataset: This dataset contains training and test sets. Each set includes 10 categories with a total of 9,298 samples. The samples are grayscale images with pixels. Examples are shown in Fig. 5.

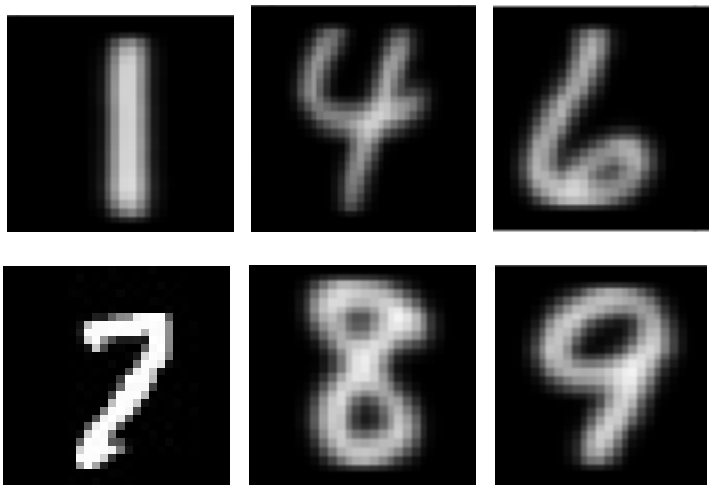


Fig. 5. Samples in the USPS dataset

SVHN dataset: This dataset contains training and test sets. Each set includes 10 numeric categories with a total of 90,000 samples. The samples are grayscale images with 32×32 pixels, and some examples are depicted in Fig. 6.

Fig. 6. Samples in the SVHN dataset

3.2 Model Parameter Setting

In our experiment, the balance parameters of the loss function are determined by the following:

$$\lambda_1 = \frac{2}{1+exp\left(-10\times pro\right)} - 1, \tag{9}$$

and

$$\lambda_2 = \frac{0.2}{1+exp\left(-10\times pro\right)} - 1, \tag{10}$$

where *pro* represents the training process linearly changing from 0 to 1 [15]. Other parameter settings are presented in Table 1, where $\mu^{(0)}$ represents the initial learning rate. The equation of learning rate update is as follows:

$$\mu^{(k)} = \frac{\mu^{(0)}}{(1+\alpha\cdot pro)\beta}, \tag{11}$$

where $\alpha = 10$, and $\beta = 0.75$, as shown in [14]. The various initial learning rates are on the MNIST → USPS task, whose results are shown in Fig. 7. Fig. 7 shows that our method achieves the best results with an initial learning rate of 0.01. Therefore, the initial learning rate was set to 0.01.

Table 1. Model parameter setting

Source → Target.	n_{epoch}	n_{batch}	$\mu^{(0)}$
Office-31	160	128	0.001
SVHN → MNIST	160	64	0.01
MNIST → USPS	160	64	0.01
USPS → MNIST	160	64	0.01

We adopt AlexNet [24] as base architectures for learning deep representations. The dropout operation is used in the fully connected layers fc6 and fc7; the neurons are ignored with a probability of 50%; and the fully connected layer fc8 is adopted as the input of the classifier and domain discriminators. According to the difficulty of the transfer learning task, we designed the differentiated architectures of the classifier and domain discriminators. For the transfer learning tasks USPS → MNIST and MNIST → USPS, the classifier architecture is fc8 → 100 → 100 → 10, and the architecture of each domain discriminator is fc8 → 100 → 1. For the transfer learning task SVHN → MNIST, the architecture of the classifier is fc8 → 3072 → 2048 → 10, and the architecture of each domain discriminator is fc8 → 1024 → 1024 → 1. Subsequently, the same architecture fc8 → 1024 → 1024 → 1 as RevGrad [15] is used in the Office-31 dataset. Finally, the normalization operation is applied to the convolutional layer to ensure a fair comparison.

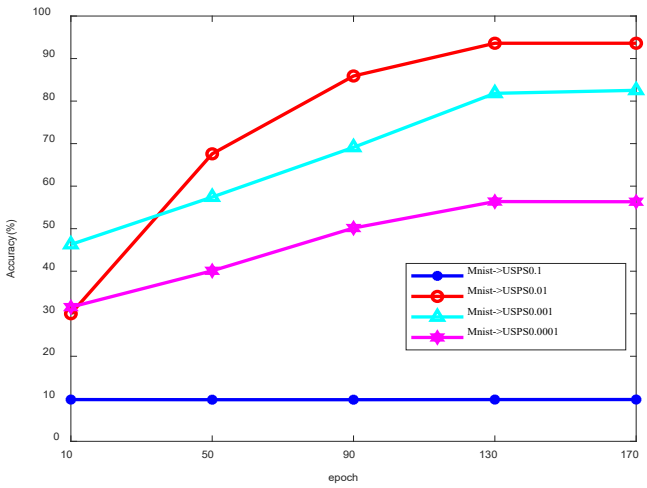


Fig. 7. Comparison of the learning rate curve

Table 2. Classification accuracy on the Office-31 dataset (%)

Methods	D → W	W → D	A → D	D → A	A → W	W → A	Avg
DDC [6]	95.0	98.5	64.4	52.1	61.8	52.2	70.6
JAN [7]	96.6	99.5	71.8	58.3	74.9	55.0	76.0
DTLC [9]	97.1	99.2	68.6	55.5	70.78	54.1	74.2
RevGrad [15]	96.4	99.2	72.3	53.4	73.0	51.2	74.3
AlexNet [24]	95.4	99.0	63.8	51.1	61.6	49.8	70.1
DRCN [25]	96.4	99.0	66.8	56.0	68.7	54.9	73.6
B-JMMD [30]	96.2	99.4	72.3	57.2	76.7	**58.4**	76.7
D-CORAL [31]	95.7	99.2	66.8	52.8	66.4	51.5	72.1
DAN [32]	96.0	99.0	67.0	54.0	68.5	53.1	72.9
Ours	**97.5**	**99.5**	**72.9**	**59.5**	**76.7**	57.1	**77.2**

We evaluated the performance of the proposed method against state-of-the-art domain adaptation methods based on AlexNet, including DDC [6], JAN [7], DTLC [9], CoGAN [13], DANN [14], RevGrad [15], ADDA [16], DRCN [25], ATDA [26], UNIT [27], GTA [28], MSTN [29], B-JMMD [30], D-CORAL [31], and the deep adaptation network (DAN) [32]. The comparison results on the Office-31 dataset are presented in Table 2, and the unsupervised domain adaptation results are taken directly from published studies to ensure a fair comparison with identical evaluation settings. DDC [6] introduces an adaptation layer and the domain confusion loss function to AlexNet and fine-tunes it in the source and target domains. Based on JMDD [7], B-JMMD [30] uses a novel backpropagation algorithm to further reduce domain discrepancy. Deep CORAL [31] learns a nonlinear transformation to align correlations of layer activations in deep neural networks and measures the differences between the source and target domains using CORAL loss. DAN [32] adopts an optimal multikernel MMD in

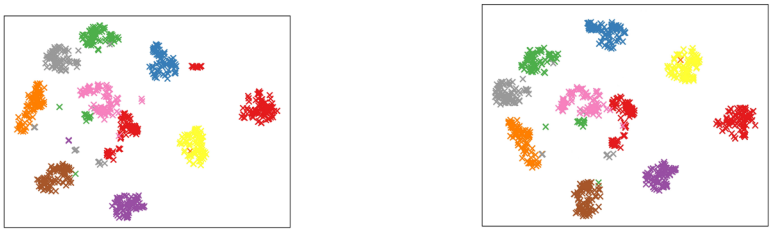
multilayers to match the mean embeddings of marginal distributions. We can draw the following observations from Table 2. Our method outperforms almost all compared approaches on most of the transfer tasks (five out of six tasks). Specifically, Our method significantly exceeds the performance of both DDC [6] and AlexNet [24], and successfully avoids the negative transfer trap. The average classification accuracy (Avg.) of our approach is 77.1%, which is an average improvement over the B-JMMD of 0.5%.

From the comparison results in Table 3, the accuracy of our method is shown to be improved on multiple transfer tasks, especially for some problematic transfer tasks, such as the SVHN → MNIST transfer task. Compared with MSTN, the Avg. of our approach is improved by 1.6% in the target domain. Compared with GTA, the Avg. of the proposed model is improved by 1.8% in the target domain. Compared with the ATDA algorithm, the average accuracy of the proposed model is improved by 1.5% in the target domain. Second, the *t*-distributed stochastic neighbor embedding (t-SNE) visualization approach is adopted to compare the feature distribution of our method using multiple adversarial networks with that of DANN [14] using a single adversarial network on the SVHN → MNIST task.

Table 3. Classification accuracy on the USPS–MNIST–SVHN datasets (%)

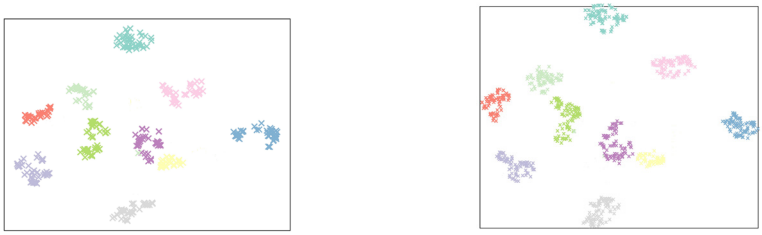
Method	USPS → MNIST	MNIST → USPS	SVHN → MNIST	Avg
CoGAN [13]	89.1	91.2	--	--
DANN [14]	73.0	85.1	73.9	77.3
RevGrad [15]	73.0	77.1	73.9	74.6
ADDA [16]	90.1	89.4	76.0	85.1
DRCN [25]	73.7	91.8	82.0	82.5
ATDA [26]	84.1	93.1	85.8	92.3
UNIT [27]	93.5	**95.9**	90.5	93.3
GTA [28]	90.8	92.8	92.4	92.0
MSTN [29]	--	92.9	91.7	--
Ours	**94.6**	93.7	**93.3**	**93.8**

Fig. 8(a) and Fig. 8(b) show the data distribution of the categories in the source and target domains for the DANN model. It can be observed that the classification performance is poor due to outliers. Fig. 8(c) and Fig. 8(d) show the data distributions of the categories for our model. It can be observed that the outliers have disappeared, and the alignment of each category is realized with increased accuracy.



(a) Source = SVHN, DANN model

(b) Target = MNIST, DANN model



(c) Source = SVHN, our model

(d) Target = MNIST, our model

Fig. 8. Visualized results with t-SNE

The proxy A-distance [14] is used as the evaluation criterion to measure the similarity between the source and target domains in our experiment. The proxy A-distance is expressed as follows:

$$d_{\mathrm{A}} = 2\left(1 - 2\varepsilon\right) , \tag{12}$$

where d_{A} is called the H-divergence. ε represents the generalization error, which can be calculated from the average value of the test error or the expectation. The comparison results are demonstrated in Fig. 9.

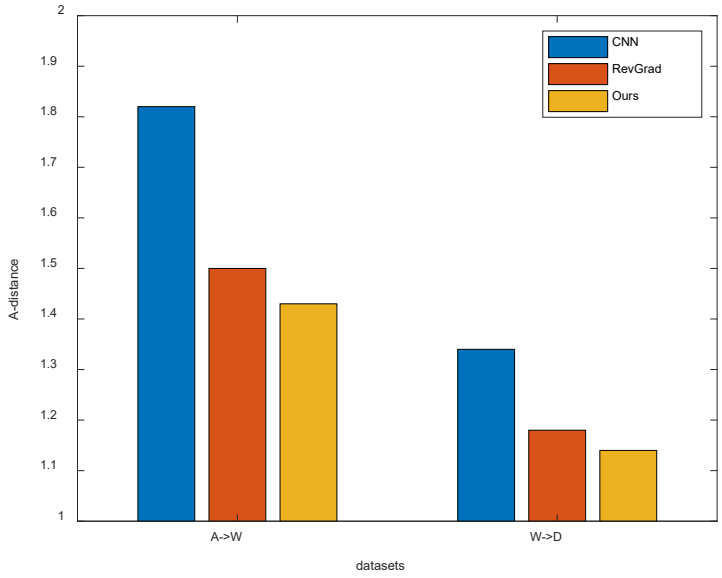


Fig. 9. Comparison of the distribution difference

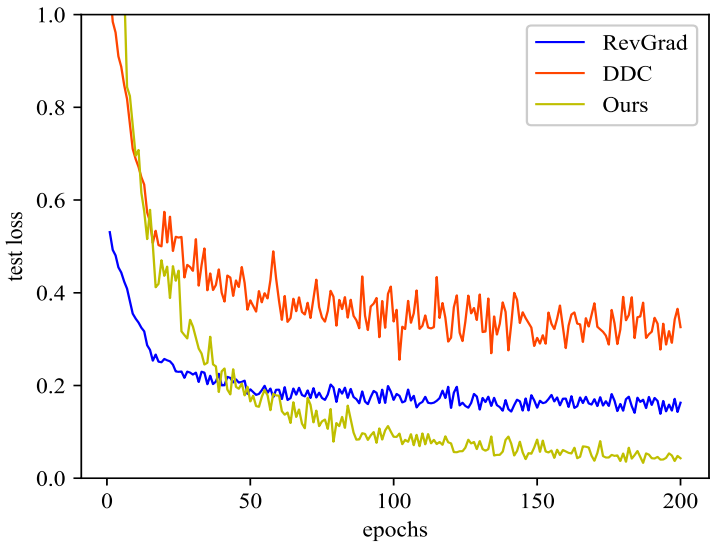


Fig. 10. Convergence comparison on the transfer task A → W from the Office-31 dataset

From Fig. 9, the proxy A-distance value obtained by the CNN model without domain adaptation is observed to be significant because the categories in the source and target domains are not effectively aligned. The difference between domains is significantly reduced in the adversarial-based RevGrad [15]. Furthermore, compared with RevGrad [15], the A-distance value in our method is reduced by 0.05 for the A → W transfer task and 0.04 for the W → D task. In our method, the sample distribution of the same category is observed to be increasingly consistent after domain adaptation when various adversarial networks are used. The convergence of DDC [6], RevGrad [15], and our method is compared (Fig. 10. Furthermore, among the models, the test loss is observed to be the largest in the DDC model after the 200th epoch. RevGrad and our method with domain adaptation have good convergence performance. Our method performs better than RevGrad because multiple adversarial networks are adopted and accurate alignments of each category between the source and target domains are realized.

4 Conclusions

In this study, we present a dynamically weighted multi-adversarial domain adaptation method under unsupervised learning to align each category data distribution in the source and target domains. Our method uses the label similarity measurement function to accurately predict the label of the target domain data. Triplet loss functions are jointly constructed to further optimize model parameters. By minimizing the loss functions, the model gradually converges and reduces the prediction error. The credibility of the predicted label is further improved by minimizing the entropy value of the predicted label in the target domain during training such that the weight coefficients of the loss functions of the domain discriminators are reasonable. Furthermore, our method does not use the directly predicted label in the target domain, reducing the risk of using the wrong data in the target domain during training. The experimental results show that our method can perform better than other approaches in the target domain.

Our next research goals are to further reduce the computational complexity of predicting the label in the target domain to improve training efficiency and design a more effective loss function with a large interclass difference to promote model performance. Additionally, we aim to extend our model to advance multiple source and target domains.

5 Acknowledgment

We would like to express our gratitude for the partial support from the Applied Basic Research Project of Liaoning Province (2022JH2/101300279).

References

[1] M. Long, Y. Cao, Z. Cao, J. Wang, M.I. Jordan, Transferable representation learning with deep adaptation networks, IEEE transactions on pattern analysis and machine intelligence 41(12)(2019) 3071-3085.

[2] Y. Zhu, F. Zhuang, J. Wang, J. Chen, Z. Shi, W. Wu, Q. He, Multi-representation adaptation network for cross-domain image classification, Neural networks 119(2019) 214-221.

[3] J. Thompson, M. Schonwiesener, Y. Bengio, D. Willett, How transferable are features in convolutional neural network acoustic models across languages? in: Proc. 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2019.

[4] B. Sun, J. Feng, K. Saenko, Return of frustratingly easy domain adaptation, in: Proc. 2016 the 30th AAAI Conference on Artificial Intelligence (AAAI), 2016.

[5] Y. Zhang, B. Deng, H. Tang, L. Zhang, K. Jia, Unsupervised Multi-Class Domain Adaptation: Theory, Algorithms, and Practice, IEEE transactions on pattern analysis and machine intelligence 44(5)(2022) 2775-2792.

[6] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, T. Darrell, Deep domain confusion: Maximizing for domain invariance, arXiv:1412.3474. <https://arxiv.org/abs/1412.3474v1>, 2014 (accessed: 08.09.22).

[7] M.S. Long, H. Zhu, J.M. Wang, M.I. Jordan, Deep transfer learning with joint adaptation networks, in: Proc. 2017 the 34th international conference on machine learning (ICML), 2017.

[8] X. Zhang, F.X. Yu, S.F. Chang, S. Wang, Deep transfer network: Unsupervised domain adaptation, arXiv:1503.00591. <https://arxiv.org/abs/1503.00591>, 2015 (accessed: 08.09.22).

[9] Z. Ding, Y. Fu, Deep transfer low-rank coding for cross-domain learning, IEEE transactions on neural networks and

learning systems 30(6)(2019) 1768-1779.
[10] H. Yan, Y. Ding, P. Li, Q. Wang, Y. Xu, W. Zuo, Mind the class weight bias: Weighted maximum mean discrepancy for unsupervised domain adaptation, in: Proc. 2017 IEEE conference on computer vision and pattern recognition (CVPR), 2017.
[11] I.J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, in: Proc. 2014 Advances in Neural Information Processing Systems (NIPS), 2014.
[12] M.Y. Liu, O. Tuzel, Coupled generative adversarial networks, in: Proc. 2016 in Advances in neural information processing systems (NIPS), 2016.
[13] Y.C. Chen, Y.Y. Lin, M.H. Yang, J.B. Huang, Crdoco: Pixel-level domain transfer with cross-domain consistency, in: Proc. 2019 IEEE conference on computer vision and pattern recognition (CVPR), 2019.
[14] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, V. Lempitsky, Domain-adversarial training of neural networks, Journal of machine learning research 17(1)(2016) 1-35.
[15] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, D. Erhan, Domain separation networks, in: Proc. 2016 Advances in Neural Information Processing Systems (NIPS), 2016.
[16] E. Tzeng, J. Hoffman, K. Saenko, T. Darrell, Adversarial discriminative domain adaptation, in: Proc. 2017 IEEE conference on computer vision and pattern recognition (CVPR), 2017.
[17] J. Hu, J. Lu, Y.P. Tan, Deep transfer metric learning, in: Proc. 2015 IEEE conference on computer vision and pattern recognition (CVPR), 2015.
[18] S. Roy, A. Siarohin, E. Sangineto, S.R. Bulo, N. Sebe, E. Ricci, Unsupervised domain adaptation using feature-whitening and consensus loss, in: Proc. 2019 IEEE conference on computer vision and pattern recognition (CVPR), 2019.
[19] Z. Pei, Z. Cao, M. Long, J. Wang, Multi-adversarial domain adaptation, in: Proc. 2018 AAAI conference on artificial intelligence (AAAI), 2018.
[20] K. Saenko, B. Kulis, M. Fritz, T. Darrell, Adapting visual category models to new domains, in: Proc. 2010 the 11th European conference on computer vision (ECCV), 2010.
[21] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE 86(11)(1998) 2278-2324.
[22] J.J. Hull, A database for handwritten text recognition research, IEEE transactions on pattern analysis and machine intelligence 16(5)(1994) 550-554.
[23] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, A. Ng, Reading digits in natural images with unsupervised feature learning, in: Proc. 2011 NIPS workshop on deep learning and unsupervised feature learning (NIPS), 2011.
[24] A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet classification with deep convolutional neural networks, Communications of the ACM 60(6)(2017) 84-90.
[25] M. Ghifary, W.B. Kleijn, M. Zhang, D. Balduzzi, W. Li, Deep reconstruction-classification networks for unsupervised domain adaptation, in: Proc. 2016 European Conference on Computer Vision (ECCV), 2016.
[26] K. Saito, Y. Ushiku, T. Harada, Asymmetric tri-training for unsupervised domain adaptation, in: Proc. 2017 International Conference on Machine Learning (ICML), 2017.
[27] M.Y. Liu, T. Breuel, J. Kautz, Unsupervised image-to-image translation networks, in: Proc. 2017 Advances in neural information processing systems (NIPS), 2017.
[28] S. Sankaranarayana, Y. Balaji, C.D. Castillo, R. Chellappa, Generate to adapt: Aligning domains using generative adversarial networks, in: Proc. 2018 IEEE conference on Computer Vision and Pattern Recognition (CVPR), 2018.
[29] S. Xie, Z. Zheng, L. Chen, C. Chen, Learning semantic representations for unsupervised domain adaptation, in: Proc. 2018 International Conference on Machine Learning (ICML), 2018.
[30] C. Meng, C. Xu, Q. Lei, W. Su, J. Wu, Balanced joint maximum mean discrepancy for deep transfer learning, Analysis and applications 19(3)(2021) 491-508.
[31] B. Sun, K. Saenko, Deep CORAL: correlation alignment for deep domain adaptation, in: Proc. 2016 European Conference on Computer Vision (ECCV), 2016.
[32] M. Long, Y. Cao, J. Wang, M.I. Jordan, Learning transferable features with deep adaptation networks, in: Proc. 32nd International Conference on Machine Learning (ICML), 2015.