Adaptable Balanced Distribution for Domain Adaptation With Strong Alignment

ZHENGSHAN WANG†1, XIANGJUN WANG†1, FENG LIU†1, PEIPEI GAO‡2, AND YUBO NI†1

†State Key Laboratory Precision Measuring Technology and Instruments, Tianjin University, Tianjin 300072, China
‡College of Computer Science, Nankai University, Tianjin 300071, China

Corresponding author: Xiangjun Wang (tjuxjw@126.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 51575388.

ABSTRACT Aligning and balancing the marginal and conditional feature distributions are two critical procedures for unsupervised domain adaptation (UDA) problems. However, existing methods usually consider the former while ignoring the latter. To improve the problems of instability and imbalance, we propose the Adaptative Joint Distribution Adaptation Network (AJDAN) by analyzing the multi-modal interactions between the two types of distributions and adding a self-learning network to simultaneously balance them. Furthermore, we give higher weights to samples that are far away from the domain boundary (easy-to-classify samples) using Strong Binary Cross-Entropy (SBCE). The strong alignment strategy is adjustable and allows the network to better train easy-to-classify samples than traditional Binary Cross-Entropy (BCE) in various scenarios. The experiment shows that AJDAN with SBCE (AJDAN+S) provides an average of 68.3% accuracy on the Office-Home dataset, and 89.1% accuracy on the Office31 dataset, showing its superiority by 2~3 percent above the existing state-of-the-art methods.

INDEX TERMS Domain adaptation, deep learning, joint distribution alignment, strong alignment.

I. INTRODUCTION

In recent years, deep learning models have achieved great success in the field of computer vision such as object recognition [1]–[4], object detection [5], [6], and object tracking [7], [8] due to abundant training datasets. However, due to the phenomenon known as dataset bias or domain shift [9]–[14], the performance of existing models trained on source datasets tends to be reduced heavily on different but related target datasets [15]. Under these circumstances, transfer learning which aims at alleviating this phenomenon, has become popular in recent years. Transfer learning usually assumes that the source and target data are from similar but different distributions [13]. For instance, the same zebra would have different distributions between a realistic photograph and an oil painting. Therefore, the critical procedure for transfer learning is to minimize the distribution divergence among different domains [17]–[21].

Domain adaptation approaches can effectively minimize distribution divergence. There are mainly three ways to realize domain adaptation in existing methods: adapt marginal distribution [22] alone, conditional distribution [23] alone, or both [17], [20]. Experiments have shown that aligning two distributions together is far more efficient than aligning either alone [24], and mainstream solutions have adopted the idea of aligning two distributions together currently. In this context, aligning two distributions together and maintaining their balance simultaneously have become more significant.

Existing methods such as Joint Distribution Adaptation (JDA) [17] or Joint Adaptation Networks (JAN) [25] align multiple convolutional network layers simultaneously. However, different distributions are often considered equally in existing methods, which is not consistent with the actual situation. As shown in Fig.1, in real applications, the marginal and conditional distributions often contribute together to the overall distribution, and their effects are not always the same. For example, under the situation of similarity between the two domains being fairly low (source → target I in Fig.1), the marginal distribution becomes more critical. When the marginal distribution is close (source → target II in Fig.1), the conditional distribution should be taken more seriously [24]. Unfortunately, in practice, it is usually challenging to determine which distribution is more critical (the unknown Target in Fig.1). To solve this imbalanced problem, existing methods such as Balanced Distribution Adaptation (BDA) [26], Dynamic Adversarial Adaptation Network
FIGURE 1. The influence of marginal distribution and conditional distribution.

(DAAN) [24], and Manifold Embedded Distribution Alignment (MEDA [27]) give these two distributions different weights by calculating the MMD [28] distance or Proxy $A$-distance [9]. However, the two distributions are independent in these methods and thus these methods fail to fully capture their underlying multi-modal interactions. In addition to this, the parameters in these methods need to be given manually and cause instability among multiple datasets. Under these circumstances, we combined the two distributions and let the network maintain their balance through self-learning. Compared with the existing methods, the main characteristic of this paper is replacing the previous manual parameters with self-learning parameters. In addition, this paper is also the first to propose this idea.

In addition to the distribution alignment strategy mentioned above, a suitable loss function can also improve the performance of final network. Existing methods adopt Cross-Entropy (CE) or Binary Cross-Entropy (BCE) [29] loss functions for the discriminator [30], which will lead to the problem of vanishing gradients. As shown in Fig.2, if the input samples are far from the domain boundary (easy-to-classify samples), the training curves of these samples will fall to the dotted line position. These easy-to-classify samples cause slight errors because they are on the right side. However, these samples are still far from the domain boundary, which means the features extracted from them are not transferable. Finally, the domain classifier network will “ignore” these samples, and the convolutional network part cannot learn the actual transferable features due to not obtaining enough loss gradient. Existing methods, such as Least Squares Generative Adversarial Networks [31] (LSGAN), adopt the least squares loss function for the discriminator. However, the loss function of LSGAN is constant, while the characteristics of datasets are varied and LSGAN fail to apply to multiple datasets. Therefore, this paper designed a novel loss function using Strong Binary Cross-Entropy (SBCE) which can give higher weights to samples far away from the domain boundary. Furthermore, SBCE can be adjusted flexibly according to different problems. As far as we know, SBCE is the first loss function in the field of transfer learning aiming to tackle easy-to-classify samples and is adjustable in various scenarios.

This paper proposes two novel methods to tackle the above two issues. For distribution adaptation, we present the Adaptative Joint Domain Adaptation Network (AJDAN). Compared with the existing techniques, AJDAN combines the marginal and conditional distributions and balances them through self-learning, which significantly enhances the robustness of adaptation ability. For the shortcoming of the traditional loss function, we propose the Strong Binary Cross-Entropy (SBCE), which can dynamically increase the weight to these easy-to-classifier samples. To sum up, our main contributions are as follows:

- We propose AJDAN, which combines marginal and conditional feature distributions and at the same time maintains their balance through self-learning.
- We design a new loss function called SBCE. By dynamically increasing the weight of samples far away from the domain boundary, SBCE can realize strong alignment between the source and target domains.

II. RELATED RESEARCH

A. UNSUPERVISED DOMAIN ADAPTATION (UDA)

Unsupervised domain adaptation (UDA) belongs to a sub-direction of transfer learning, which is an open theoretical and practical problem. Defining datasets with different distribution as different domains, the purpose of UDA is to use the source labeled domain dataset to train the network,
and the trained network can achieve good performance in the unlabeled target domain dataset. The key of UDA is to project the transformation of different domains to the same subspace, such as Reproducing Kernel Hilbert Space (RKHS) and reduce the distribution discrepancy.

Compared to traditional methods, deep networks can learn more transferable features for domain adaptation [32], [33]. Therefore, domain adaptation methods based on deep learning have become popular these years. For instance, Maximum Mean Discrepancy (MMD) [25], [34], Correlation Alignment (CORAL) [35], and Central Moment Discrepancy (CMD) [36] aim to reduce the distribution divergence among domains. The distribution divergence previously was limited to a single layer and gradually expanded to multi-layers [25]. For example, Pan et al. proposed Transfer Component Analysis (TCA) [22] to align the marginal distributions between domains. Based on TCA, Joint Distribution Adaptation (JDA) [17] proposes matching both marginal and conditional distributions. Subsequent researchers [20], [37] extended JDA by adding structural consistency and domain invariant clustering. More layers mean more structural information, and how to manage multiple information become a new problem. Existing methods such as Balanced Distribution Adaptation (BDA) [26], Dynamic Adversarial Adaptation Network (DAAN) [24], and Manifold Embedded Distribution Alignment (MEDA) [27] all give their solutions. They provide different distributions with different weight by calculating the MMD [28] distance or Proxy A-distance [9]. However, these methods do not perform consistently on multiple datasets because of artificial parameters. Therefore, reducing artificial parameters to make the network realize self-learning is a direction to improve the network performance.

B. DOMAIN-ADVERSARIAL LEARNING

As a particular case of deep domain adaptation, domain-adversarial learning means these methods that learn transferable features and minimize distribution discrepancy using adversarial learning methods [38]. They integrate adversarial learning and domain adaptation into a two-player game such as Generative Adversarial Networks (GANs) [30]. Specifically, they exhibit an architecture whose feature extraction layers are simultaneously shared by the task classifier and domain classifier. The first classifier aims to correctly predict task-specific class labels on the source data, while the second aims to predict the domain labels of input samples. The feature extraction layers can learn domain invariant features by deceiving domain classifiers and eventually achieve the goal of domain adaptation.

Recently, we have witnessed a considerable amount of research [39]–[41] on domain-adversarial learning. For example, DANN [42] proposed a domain adversarial training method to generate domain invariant features by deceiving a domain discriminator. As a pioneering work, their domain-adversarial learning idea was adopted by many subsequent researchers [43], [44].

However, when data distributions embody complex multi-modal structures, existing methods based on one domain classifier may fail to capture such multi-modal structures. Therefore, they cannot realize discriminative distribution alignment without mode mismatch. Some researchers have given their solutions. JDA [17] attempted to tackle this by aligning distributions via separate domain discriminators. MADA [40] enabled fine-grained alignment of different data distributions based on multiple domain discriminators. Co-DA [41] constructed multiple diverse feature space and aligned distributions individually. Unfortunately, these methods do not work very well since separating label-prediction and feature distributions make it hard to capture the multi-modal information between them. In CDAN [45], the author attempted to capture the multi-modal structures underlying the data by a multi-linear conditioning scheme. However, experiments show that the multi-linear conditioning scheme cannot seize complete joint feature alignment because it ignores the marginal distribution information. We try to solve these problems by creating a new joint distribution feature and therefore improved the accuracy by 2 ~ 3%.

III. METHOD

This section elaborates on our proposed algorithm. First, we introduce the problem definition. Then, we cover the basics and then present the Adaptive Joint Domain Adaptation network (AJDAN). Finally, we recommend the Strong AJDAN (AJDAN+S) method.

A. PROBLEM DEFINITION

Transfer learning applies to many scenarios, such as object recognition, object detection, and object tracking. This paper mainly studies object recognition problems. Assume a source domain $D_s = \{(x_i^s, y_i^s)\}_{i=1}^{n_s}$ and a target domain $D_t = \{(x_i^t)\}_{i=1}^{n_t}$, where $n_s$ represents the number of labeled samples, $n_t$ represents the number of unlabeled samples. $x_i^s$ and $x_i^t$ represent the source and target input images respectively and $y_i^s$ represents the corresponding class labels. The $D_s$ and $D_t$ here are sampled from two joint distributions $P(x^s, y^s)$ and $Q(x^t, y^t)$ respectively, which distributes independently and identically ($P \neq Q$).

Suppose an image classification network is composed of a feature extractor part $G$ and a task label classifier part $C$. The objective function of the network can be summarized as:

$$L_{\text{task}}(\theta_C, \theta_G) = \frac{1}{n_s} \sum_{i=1}^{n_s} \ell(C(G(x_i^s)), y_i^s)$$

(1)

where $\theta_C$ and $\theta_G$ mean the parameters of $C$ and $G$. $\ell$ means the loss function. To simplify the following analysis, we define $G(x)$ with $f_m$ and define $C(G(x))$ with $\hat{y}$. The goal of transfer learning is using the feature extraction network $G$ and task classifier $C$ trained in $D_s$ to obtain $y^t$ of $D_t$. 

VOLUME 9, 2021

100667
B. ADVERSARIAL LEARNING FOR DOMAIN ADAPTATION

Inspired by GAN [30], adversarial learning has been effectively used to minimize the discrepancy between domains. It uses a domain classifier $D$ to distinguish the domain label of the input sample, whose objective function is:

$$L(\theta_D) = \frac{1}{n_s + n_t} \sum_{x_i \in D_s \cup D_t} \ell(D(P), z_i)$$

where $z_i$ means the domain label of the input samples. $P$ means the feature distribution of the input sample. $\theta_D$ means the parameter of the domain classifier $D$. Motivated by GAN, the feature extractor $G$ acts like the generator, aiming to produce the transferable feature representation for confusing the domain classifier $D$. The domain classifier $D$ plays the role of the discriminator, attempting to detect whether the extracted features come from the source domain or the target domain. Under the circumstances, $G$ and $D$ form an adversarial relationship.

After the training converges, $D$ fails to distinguish the input samples from the two different domains, meaning the discrepancy between the two domains is small, and they are fused totally. This training process can be formalized as (3).

$$\begin{align*}
(\hat{\theta}_D) &= \arg\min_{\theta_D} L(\theta_D) \\
(\hat{\theta}_G) &= \arg\max_{\theta_G} L(\theta_D) \approx Distance(P^s, P^t) \rightarrow 0
\end{align*}$$

where $\hat{\theta}_D$ and $\hat{\theta}_G$ mean the parameters of network $D$ and $G$ when they fully converge. Here, $\theta_D$ aims to reduce $L(\theta_D)$, and $\theta_G$ aims to increase $L(\theta_D)$. After convergence, the two kinds of distributions are completely fused and the distance between them closes to zero.

C. ADAPTATIVE JOINT DOMAIN ADAPTATION NETWORK (AJDAN)

The practical procedure of transfer learning is to minimize both the marginal and conditional feature distribution [17], [20] between domains. Specifically, this refers to minimizing the distance as shown in (4).

$$D(D_s, D_t) \approx (1 - \mu)Distance(P(x^s), P(x^t)) + \mu Distance(P(y^s|x^s), P(y^t|x^t)) \mu \in [0, 1]$$

Based on the theories in DANN [42], $f_m$ contains the global feature information and thus we use $f_m$ to represent $P(x)$. Because the target domain $D_t$ has no labels, it is not feasible to calculate the conditional feature distribution $P(y^t|x^t)$. According to a quantitative study [33], [45], the output of the task classifier $\hat{y}$ potentially reveals the multi-modal structure. Therefore, many existing methods combine $\hat{y}$ and $f_m$ to simulate conditional feature representation. Research [46] has shown that a multilinear map of infinite-dimensional nonlinear feature maps has been successfully applied to embed a conditional distribution and the multilinear map can model multiplicative interactions between different variables. Inspired by this, different from most existing methods, that concatenates $\hat{y}$ and $f_m$ roughly, we adopt the tensor product between $\hat{y}$ and $f_m$ to perform lossless multilinear fusion. This operation can be represented by the formula:

$$P(y|x) = f_m \otimes \hat{y}$$

To facilitate the following analysis, we use $f_c$ to refer to $P(y|x)$. Equation (5) can be simplified as (6).

$$f_c = f_m \otimes \hat{y}$$

The equilibrium parameter $\mu$ in (4) control the weight of the marginal distribution and the conditional distribution. When $\mu \rightarrow 0$, it means that the datasets are dissimilar. Thus, the marginal distribution $P(x)$ is more dominant. When $\mu \rightarrow 1$, it reveals that the datasets are similar, so the conditional distribution $P(y|x)$ is more important. Existing methods such as MEDA [26] evaluate the weight using the proxy $A - distance$ by learning a domain invariant classifier, and DAAAN [24] obtains $\mu$ using the value of loss function from different domain classifiers. However, MEDA requires 1 + C binary classifiers for the calculation of its adaptive factor, and DAAAN [42] contains too many manual parameters and lacks stability over multiple datasets. Apart from reducing the existence of manual parameters, experiments have also shown that it is necessary to analyze the status of feature distributions and their relationship to enhance the performance. Existing methods usually consider the former and ignore the latter, which leads to a decline in their performance. Therefore, the paper tries to reduce the number of artificial parameters and capture the relationship between distributions.

To avoid using the artificial parameter $\mu$, we plan to form the marginal feature representation $f_m$ and conditional feature representation $f_c$ into a new joint variable $f_j$ and then equation (4) can be transformed into (5).

$$D(D_s, D_t) \approx Distance(f_j(P(x_s), P(y_s|x_s))), f_j(P(x_t), P(y_t|x_t))$$

Determining the ways to obtain $j$ requires a specific analysis of the status of the two distributions. However, the marginal and conditional feature distribute in RKHS, which makes it arduous to analyze them manually. A recent advance named Conditional Generative Adversarial Networks (CGANs) [47] discovers that different distributions can be matched better by conditioning the generator and discriminator on relevant information, such as associated labels and affiliated modality. This discovery shows that GAN can be extended to a conditional model using some specific auxiliary information, such as class labels or data from other modalities.

Motivated by conditional GANs, we observe that different distributions can be combined as auxiliary information to each other. By the way shown in (6), domain variances in both feature representations $f_m$ and $f_c$ can be modeled simultaneously.

$$f_j = f_m \oplus f_c$$

$$f_j = f_m \oplus f_c$$ here means concatenating the marginal feature representation $f_m$ and conditional feature representation $f_c$.
into a new joint variable \( f_j \). We name this process as joint combination. The new variable \( f_j \) here contains the joint information of the original two feature distributions. This is a lossless combination because the original feature information is unchanged, so we do not have to worry about information loss during the transformation process. Furthermore, the mutual auxiliary of the two kinds of feature information also enables the network to extract deeper information between them [47]. We set a joint domain discriminator \( D_j \) to analyze the joint information and predict the domain labels of input samples. The following way to align the joint variable \( f_j \) is the same as that to align a single distribution. The entire network architecture is shown in Fig.3.

We named this method as Adaptative Joint Domain Adaptation (AJDAN). Experiments show that AJDAN can achieve good results with a simplified network. The objective function of AJDAN is as follows.

\[
L_{total}(\theta_G, \theta_C, \theta_D) = \frac{1}{n_c} \sum_{i=1}^{n_c} l_{task}(C(G(x_i^j)), y_i^j) - \frac{\lambda}{n_s + n_t} \sum_{x_i \in D_s \cup D_t} [\ell(D_j((G(x_i^j) \otimes C(G(x_i^j)))) \oplus G(x_i^j)), z_i)]
\]

(9)

After training convergence, the parameters \( \theta_G, \theta_C, \theta_D \) will deliver a saddle point.

\[
(\hat{\theta}_G, \hat{\theta}_C) = \arg \min_{\theta_G, \theta_C} L_{total}(\theta_G, \theta_C, \theta_D)
\]

(10)

\[
(\hat{\theta}_D) = \arg \max_{\theta_G, \theta_C} L_{total}(\theta_G, \theta_C, \theta_D)
\]

(11)

AJDAN can also be viewed as an updated version of existing methods. The difference between them lies in AJDAN replacing manually calculated weight parameters with network self-training weight parameters. Experimental results show that adaptive learning makes it possible to achieve better results over multiple datasets without adding another large network.

We can theoretically demonstrate the conclusion that AJDAN is an upgraded version of existing methods. Taking DANN [34] as an example, which denotes the distance of the feature distribution as

\[
\text{Distance}(P(x_1), P(x_2)) = 2(1 - 2L(\theta_{D_m})))
\]

\[
\text{Distance}(P(y^{|x^s|}), P(y^{|x^t|})) = 2(1 - 2L(\theta_{D_t})))
\]

(12)

Set two domain discriminators \( D_m \) and \( D_t \), which are used for marginal features and conditional features respectively. Then, equation (4) can be converted to (13).

\[
\min(\text{Distance}(D_s, D_t)) \approx \min((1 - \mu)\text{Distance}(P(x^s), P(x^t)) + \mu \text{Distance}(P(y^{|x^s|}), P(y^{|x^t|})))
\]

\[
= \min(2 - (1 - \mu)(1 - 2L(\theta_{D_m}))) + 2 \mu
\]

\[
= 2 - 4 \max((1 - \mu)L(\theta_{D_m}) + (\mu)L(\theta_{D_t}))
\]

(13)

where \( L(\theta_{D_m}) \) and \( L(\theta_{D_t}) \) are loss values of \( D_m \) and \( D_t \) respectively. To make the network not overly incline to the marginal or conditional feature distributions alone, their feature distributions need to satisfy (11).

\[
\text{Distance}(P(x^s), P(x^t)) \approx \text{Distance}(P(y^{|x^s|}), P(y^{|x^t|}))
\]

(14)

Based on (12), equation (14) can be simplified to (15).

\[
L(\theta_{D_m}) = L(\theta_{D_t})
\]

(15)

The existing methods usually manually adjust the calculation formula of \( \mu \) to achieve (13), whereas AJDAN hides the balancing process in the neural network training. We can transform their formulas to the same matrix form to prove the point. For existing methods, equation (13) can be expressed in matrix form (16).

\[
\min(\text{Distance}(D_s, D_t)) \quad 0 \leq \mu \leq 1
\]

\[
\rightarrow \min(\text{Distance}(f_j^s, f_j^t))
\]

\[
\rightarrow \max(Z^T M_{D_m} F_M + (\mu) Z^T M_{D_t} F_C)
\]

(16)

For AJDAN, the objective function can be formalized as (14).

\[
\min(\text{Distance}(D_s, D_t))
\]

\[
\rightarrow \min(\text{Distance}(f_j^s, f_j^t))
\]

\[
\rightarrow \max(Z^T M_{D_j} F_j)
\]

(17)

where \( F_M, F_C \) and \( F_j \) and \( M_{D_m}, M_{D_t}, \) and \( M_{D_j} \) are matrix forms of \( f_m, f_c, \) and \( f_j \) and \( \theta_{D_m}, \theta_{D_t}, \) and \( \theta_{D_j} \) respectively. \( Z \) is the one-hot encoding format of domain label \( z \), which is \([0, 1]^2 \) or \([1, 0]^2 \). Moreover, equation (16) and equation (17) can be converted into (18).

\[
(16) = \max(1 - \mu) Z^T M_{D_m} F_M + (\mu) Z^T M_{D_t} F_C
\]

\[
= \max(Z^T[(1 - \mu) M_{D_m} \mu * M_{D_t}][F_M, F_C]^T)
\]

\[
\approx \max(Z^T M_{D_j}[F_M, F_C]^T) = \max(Z^T M_{D_j} F_j) = (17)
\]

(18)

As shown in (18), equation (16) and (17) turn out to be equal finally. The difference is that the \( D_j \) in AJDAN replaces the functions of \( D_m, D_t, \) and \( \mu \). Therefore, compared with the existing methods, AJDAN reduces the parameters that need to be trained and reduces artificial variables.

To make the above demonstration more intuitive, as shown in Fig.4, the green lines are parameters related to marginal feature distribution, and the red lines are parameters associated with conditional feature distribution. These neuron nodes adjust the weight of the two distributions, which is same as the role of \( \mu \). Because these parameters are self-learned by the network, they are more effective and robust over multiple datasets.
FIGURE 3. The Network structure of AJDAN. $X_s$ and $X_t$ represent source domain and target domain input data. $G$ is the convolution part of the network. $C$ is the classification part of the network. The right half of the image shows the conditional and marginal distributions and their processing steps respectively.

FIGURE 4. Parameters in domain classifier $D_j$.

D. STRONG ALIGNMENT FOR EASY-TO-CLASSIFY SAMPLES

To achieve adversarial learning, existing methods [42], [45] utilize one or more domain classifiers to distinguish the target features from the source features. The loss function will affect the performance of the domain classifiers. Existing methods have widely use traditional Cross-Entropy (CE) in various deep learning tasks due to its outstanding performance. The formula of Binary Cross-Entropy (BCE), which is a particular case of CE in the binary problem is shown in (19):

$$\text{BCE} : l(p, z) = -[z \log(p) + (1 - z) \log(1 - p)]$$

(19)

where $p \in [0, 1]$ is the output of the domain classifier, $z$ is the domain label of the input sample, and $l(p, z)$ is the value of the BCE loss function. Fig.2(b) shows the function curve of BCE when $z$ equals 0 or 1. As shown in Fig.5, easy-to-classify samples are those that are far from the domain boundary while hard-to-classify target samples are those that are near the domain boundary. To put it another way, easy-to-classify samples are those whose predicted domain labels are close to real domain labels. However, the gradient of BCE function for the easy-to-classify samples (the dotted line position in Fig.2(b)) is gentle, and therefore the training weight is inadequate. This problem makes it challenging for the network to train easy-to-classify samples effectively. However, in real life, easy-to-classify samples occupy a considerable proportion and need to be emphasized. Existing methods that adopt traditional BCE ignore these samples, and thus their performances are reduced. Therefore, focusing on easy-to-classify samples can achieve a strong alignment and obtain more ideal results on multi datasets.

Some researchers have used the least squares loss function to relieve this problem [31], but the least squares loss function is constant and cannot be adjusted according to different datasets. To tackle this, we propose a new loss function named Strong Binary Cross-Entropy (SBCE) by adding a modulating factor $f(p_t)$ to the traditional BCE loss function. We can select the most suitable SBCE for current datasets by adjusting $f(p_t)$, so SBCE is more practical than the least squares loss function. The formula of SBCE is shown in (17).

$$\text{SBCE} : l(p_t, z) = -f(p_t) \log(p_t)$$

(20)
A(Artistic), C(Clipart), P(Product), and R(Real-World) and each domain contains 65 object classes.

VisDA2017 [50] is a challenging testbed for UDA with the domain shift from synthetic data to real imagery. In total there are ∼280k images from 12 categories.

ImageCLEF-DA [51] is a dataset organized by selecting 12 common classes shared by three public datasets (domains): Caltech-256 (C), ImageNet ILSVRC 2012 (I), and Pascal VOC 2012 (P). The dataset organizers selected 50 images per class and 600 images in total for each domain.

B. TECHNICAL DETAILS

The methods compared in this paper are Domain Adversarial Neural Network (DANN) [42], Conditional Domain Adversarial Network (CDAN) [45], Joint Adaptation Network (JAN) [25], Deep Adaptation Network (DAN) [34], and Dynamic Adversarial Adaptation Network (DAAN) [24]. Among them, DANN is aimed at marginal feature and CDAN is aimed at conditional feature. Furthermore, JAN, DAN, and DAAN consider both, but they combine two features crudely without using network self-learning. The AJDAN proposed in this paper not only combines these two, but also uses the self-learning weight. Table 4 shows the difference among these methods.

We follow the standard protocols for unsupervised domain adaptation [42], [25]. We use all labeled source examples and all unlabeled target examples and compare the average classification accuracy based on three random experiments. We conduct importance-weighted cross-validation (IWCV) [54] to select hyper-parameters for all methods. For some key parameters in existing methods, we set λ = 1 for all experiments in CDAN. For MMD-based methods such as JAN, we use Gaussian kernel with the bandwidth set to median pairwise distances on training data [34].

We use ResNet50 [52] pre-trained on the ImageNet [53] dataset as our backbone network and replace the last convolutional layer of the network for the current tasks. The labeled source domain data and unlabeled target domain data are used to finetune the model, and they share all network parameters. Because the value of λ can affect the result on the target domain, we will analyze its specific influence in a later section. Here, we use a mini-batch stochastic gradient descent (SGD) with the momentum equal 0.9 and set λ and γ to 1. The learning rate η was set up η = η0/(1+aρ)β, where ρ goes from zero to one linearly with the number of iterations. To ensure data’s authenticity, we take the average outcome of three repeat experiments as the final result.

C. EXPERIMENTAL RESULT

In this section, we compare the results of various methods in standard datasets and the full results are shown in Table. 1, Table 2, and Table 3.

Table 1 shows the classification accuracy of 12 tasks on the office-home dataset. Compared with DANN aligning the marginal distribution alone and CDAN+E aligning the conditional distribution alone, the accuracy of the
| METHOD       | A→C | A→P | A→R | C→A | C→P | C→R | P→A | P→C | P→R | R→A | R→C | R→P | AVERAGE |
|--------------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|--------|
| ResNet50 [52]| 34.9| 50.0| 58.0| 37.4| 41.9| 46.2| 38.5| 31.2| 60.4| 53.9| 41.2| 59.9| 46.1   |
| DAN [34]    | 43.6| 57.0| 67.9| 45.8| 56.5| 60.4| 44.0| 43.6| 67.7| 63.1| 51.5| 74.3| 56.3   |
| DANN [42]   | 45.6| 59.3| 70.1| 47.0| 58.5| 60.9| 46.1| 43.7| 68.5| 63.2| 51.8| 76.8| 57.6   |
| JAN [25]    | 45.9| 61.2| 68.9| 50.4| 59.7| 61.0| 45.8| 43.4| 70.3| 63.9| 52.4| 76.8| 58.3   |
| CDAN+E [45] | 50.7| 70.6| 76.0| 57.6| 70.0| 70.0| 57.4| 50.9| 77.3| 70.9| 56.7| 81.6| 65.8   |
| DAAN [24]   | 50.5| 68.9| 73.7| 53.7| 62.7| 64.6| 53.5| 45.0| 74.0| 66.3| 54.0| 78.8| 61.8   |
| AJDAN        | 51.2| 71.1| 76.0| 60.1| 70.1| 71.5| 59.2| 52.2| 78.5| 72.2| 58.2| 82.2| 66.9   |
| AJDAN+S      | 53.0| 71.5| 77.7| 62.5| 71.9| 72.7| 62.8| 53.6| 79.2| 72.5| 57.5| 84.4| 68.3   |

When the source domain and the target domain are similar, such as the D→W and W→D tasks in Office31 or I→C task in ImageCLEF-DA. AJDAN+S shows little advancement (less than 0.1%) compared with CDAN+E. The reason is because their marginal distributions are close by nature, so increasing the alignment of marginal feature distribution does not work effectively.

### D. EMPIRICAL ANALYSIS

#### 1) ABLATION STUDY

To compare the diverse influence of aligning marginal, conditional, and joint distributions, we test and visualize the performance of DANN, CDAN+E, and AJDAN. To compare the impact of strong alignment, we test the performance of AJDAN and AJDAN+S. The backbone network is ResNet50, and λ is set to 1. Fig. 7(a) shows the result.
As shown in Fig. 7(a), the joint alignment strategy performs better in most transfer tasks on Office-Home datasets. The average accuracy of AJDAN increases by 9% compared to that of DANN and 1% compared to that of CDAN+E. After adoption of the strong alignment strategy, the average accuracy of AJDAN+S increases by 11% relative to that of DANN, 3% relative to that of CDAN+E, and 1.5% relative to that of AJDAN.

It is worth noting that the strong alignment strategy is not effective in all cases. For example, we test CDAN, AJDAN, and AJDAN+S on VisDA-2017 and Digit datasets, and the results are shown in Table 5.

For the SVHN→MNIST task in the Digit dataset, AJDAN+S performs better than CDAN+E and AJDAN. However, in the VisDA-2017 dataset (Synthetic→Real), AJDAN performs better than AJDAN+S. The use of the strong alignment strategy does not give the anticipated improvement. Instead, it decreases the accuracy by 3%.

The reason is that, aligning features distribution means, not only object categories but also backgrounds and scene layouts must be similar across domains. When using strong alignment, feature alignment is done both at the global image scale and at the instance (object) scale, so strictly matching might work well for small domain shifts that only affect the appearance/texture of objects (e.g. weather related shifts), but it is likely to hurt the performance for larger shifts that affect the layout of the scene. Fortunately, the formula we give is flexible. In the above experiments, γ is fixed to 1 to make sure the experiment is consistent, but we can adjust it and make AJDAN+S adaptive for different tasks in practical application.

2) BALANCE FACTOR \( \lambda \)

Factor \( \lambda \) controls the balance between classification task and domain alignment task. To test the influence of the \( \lambda \) factor, we adjusted its value and tested the accuracy of the corresponding AJDAN+S in the A→W task of Office31. As shown in Fig. 7(b), the overall accuracy will increase first and then decrease. For example, in the A→W task, the accuracy reached a peak at 0.6, while in the D→A task, the accuracy reached a peak at 1.0. In general, factor \( \lambda \) is suitable between 0.6 and 1.

3) CONVERGENCE AND COMPUTATION COST

Fig. 7(c) is a graph showing the changes in accuracy over iteration of the A→W task on the Office31 dataset. When training, ResNet50 converges rapidly within 1000 iterations, but its accuracy in the target domain stops increasing after that and decreases instead. The reason is that as the training iteration proceeds, the network overfits the source domain, so the performance in the target domain drops instead of rising. After using the transfer learning strategy, this overfitting phenomenon is avoided in DANN, CDAN, and AJDAN. In terms of convergence speed, the convergence speed of CDAN and AJDAN is similar, and are slightly better than that of DANN. In terms of accuracy, AJDAN is slightly more accurate than CDAN and more accurate than DANN and ResNet50.

Table 6 shows the computation costs of various methods and their corresponding accuracy in Office-Home dataset.
Since the transfer network is a small part compared with the overall network, there is little difference in overall computation cost among different methods. As shown in Table 6, DAAN has the largest computation cost for setting a transfer network separately for each category. In general, AJDAN achieves the best results without additional computation.

### Visualization

To compare the existing methods and AJDAN, we use t-SNE [48] to visualize the feature distribution of ResNet50, DANN, CDAN, and AJDAN. Fig. 8 shows the feature distribution of A→W tasks in the Office31 dataset and A→C tasks in the Office-Home dataset. The two tasks correspond to remote and near domain adaptation problems respectively. The red points represent source domain and the green points represent target domain.

As shown in Fig. 8, in ResNet50, the target domain features are scattered around the source domain features, and the overall distribution is far away from the source domain, clustered in small areas. In DANN, the two types of characteristics are mixed to a certain extent, which is better than ResNet50. In CDAN, the features of the source and target domains are further fused, but there were still some unsatisfactory phenomena, such as some target domain features failing to find the corresponding source domain features. Finally, AJDAN combines the advantages of DANN and CDAN and achieves the best results compared with the former.

### V. Conclusion

Within the field of UDA, the paper proposes a new domain adaptation method named Adaptative Joint Distribution Adaptation Network (AJDAN). Experimental results show that, compared with DANN based on marginal feature alignment, CDAN based on the conditional feature alignment, and JAN based on joint alignment using manual parameters, AJDAN based on joint alignment using self-learning parameters offer a result that is 2% above the existing methods at their best. Moreover, we also replace traditional BCE with SBCE. Experimental results show that the use of AJDAN+S can further improve the accuracy by approximately 3%. Regarding the direction for improvement of AJDAN in the future, we note that the value of γ and λ needs to be determined empirically, which brings inconvenience to practical use. In future work, a better approach is needed to obtain the values of γ and λ, and the performance is expected to improve further.

### REFERENCES

[1] Y. Guo, Z. Zhang, and F. Tang, “Feature selection with kernelized multiclass support vector machine,” Pattern Recognit., vol. 117, Sep. 2021, Art. no. 107988.

[2] Y. Guo, H. Yang, M. Chen, J. Cheng, and D. Gong, “Ensemble prediction-based dynamic robust multi-objective optimization methods,” Swarm Evol. Comput., vol. 48, pp. 156–171, Aug. 2019.
[51] B. Caputo, H. Müller, J. Martinez-Gomez, M. Villegas, B. Acar, N. Patricia, N. Marvasti, S. Uşkudarlı, R. Paredes, M. Cazorla, I. García-Varea, and V. Morell, “ImageCLEF 2014: Overview and analysis of the results,” in Proc. CLEF, 2014, pp. 192–211.

[52] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 770–778.

[53] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “ImageNet: A large-scale hierarchical image database,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2009, pp. 248–255.

[54] M. Sugiyama, M. Krauledat, and K.-R. Müller, “Covariate shift adaptation by importance weighted cross validation,” J. Mach. Learn. Res., vol. 8, pp. 985–1005, May 2007.

ZHENGSHAN WANG was born in Jiangsu, China. He is currently pursuing the master’s degree with the College of Precision Instrument and Optoelectronics Engineering, Tianjin University. His research interest includes computer vision.

XIANGJUN WANG received the B.S., M.S., and Ph.D. degrees in precision measurement technology and instruments from Tianjin University, Tianjin, China, in 1980, 1985, and 1990, respectively. He is currently a Professor and the Director of the Precision Measurement System Research Group, Tianjin University. His research interests include photoelectric sensors and testing, computer vision, image analysis, MOEMS, and MEMS.

FENG LIU was born in 1978. He received the Ph.D. degree from the State Key Laboratory Precision Measuring Technology and Instruments. The main research interest includes visual detection technology.

PEIPEI GAO was born in 1978. She received the Ph.D. degree from the College of Computer Science, Nankai University, China. Her research interests include emotional computing, speech synthesis, and recognition.

YUBO NI received the B.S. and M.S. degrees from the Sino European Institute of Aviation Engineering, Civil Aviation University of China, Tianjin, China, in 2013 and 2016, respectively. He is currently pursuing the Ph.D. degree in instrumentation science and technology with Tianjin University. His research interests include precision measurement technology and instruments, vision measurement technology, and smart sensors.