An Unsupervised Domain Adaptation Model based on Dual-module Adversarial Training

Yiju Yang[a], Tianxiao Zhang[a], Guanyu Li[a], Taejoon Kim[a], Guanghui Wang[b,*]

[a] *Department of Electrical Engineering and Computer Science, University of Kansas, Lawrence, KS, USA, 66045*
[b] *Department of Computer Science, Ryerson University, Toronto, ON, Canada M5B 2K3*

Abstract

In this paper, we propose a dual-module network architecture that employs a domain discriminative feature module to encourage the domain invariant feature module to learn more domain invariant features. The proposed architecture can be applied to any model that utilizes domain invariant features for unsupervised domain adaptation to improve its ability to extract domain invariant features. We conduct experiments with the Domain-Adversarial Training of Neural Networks (DANN) model as a representative algorithm. In the training process, we supply the same input to the two modules and then extract their feature distribution and prediction results respectively. We propose a discrepancy loss to find the discrepancy of the prediction results and the feature distribution between the two modules. Through the adversarial training by maximizing the loss of their feature distribution and minimizing the discrepancy of their prediction results, the two modules are encouraged to learn more domain discriminative and domain invariant features respectively. Extensive comparative evaluations are conducted and the proposed approach outperforms the state-of-the-art in most unsupervised domain adaptation tasks.

Keywords: Domain adaptation, unsupervised learning, adversarial training, dual-module, classification.

*Corresponding author
Email address: wangcs@ryerson.ca (Guanghui Wang)

1. Introduction

In recent years, deep convolutional neural networks (CNNs) have achieved huge success in many computer vision applications, such as image classification [6, 28], object detection [27, 37], tracking [62], segmentation [20, 43], image generation [58], and crowd counting [48]. A lot of powerful network architectures have been proposed for efficient feature extraction, representation, and optimization [19, 59, 63]. However, most of these models need to be trained via supervised learning from a large collection of reliable training data.

In many practical application scenarios, we may not have or it is too expensive to obtain a lot of reliable labeled data, and labeling data correctly and accurately is usually more difficult than collecting data. For example, in medical image processing, labeling data correctly requires people to have sufficient professional training and domain knowledge. Therefore, how to make full use of unlabeled data becomes a more and more attractive topic in the computer vision field. An effective method to solve this kind of problem is to use unsupervised domain adaptation (UDA).

In unsupervised domain adaptation, we assume that there are two data sets. One is an unlabeled data set from the target task, called the target domain. The other data set is a labeled data set from the source task, called the source domain. There are some similarities between the two data sets, however, there are still some differences between them. For example, cartoon characters and real human beings have some commonalities, but we can also clearly distinguish the differences between them. When we train a model with the source domain data set, and then apply the trained model to the source task and the target task respectively, the performance for the target domain is usually significantly lower than that for the source domain. This is due to the domain shift between the source domain and the target domain. The extent of these reductions depends on the domain shift. If the domain shift is larger, the performance loss for the target task becomes greater. To solve this problem, we need to know how to align the source domain with the target domain and reduce the shift between

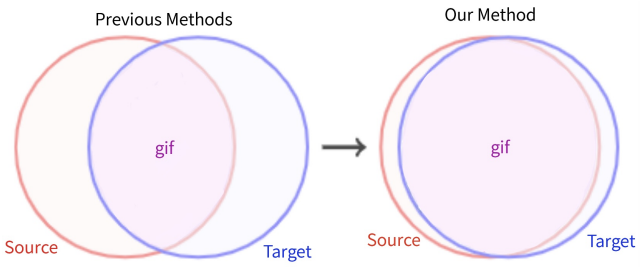


Figure 1: The red circle represents the features extracted from the source domain, and the blue circle denotes the features extracted from the target domain. The intersection area of the two circles is the gif (global invariant feature). Our method can further extract global invariant features on the basis of previous methods, reducing the distance between the source domain and the target domain.

the two domains.

With the advent of gradient reversal layers (GRL) from Domain-Adversarial Training of Neural Networks (DANN) [12], more and more people realize that adversarial training has a significant effect on aligning the feature of the source and target domains. DANN [12] extracts the global invariant features by training a domain discriminator to fool the feature extractor. The domain discriminator is a component composed of several fully connected layers. Its function is to distinguish the input data from the source domain or the target domain. If the discriminator could not recognize the extracted feature map, it means that the extracted feature map comes from the common space of these two domains. The global invariant features are from this space, and DANN utilizes a classifier for adversarial training to ensure the effectiveness of the features learned by the network. In addition to the adversarial training method based on GRL, there are many other adversarial training methods based on generative adversarial nets (GANs) [15, 32, 52, 22, 17]. Most of them have one thing in common, i.e., they adapt to the target domain by learning global invariant features.

Motivation. Taking DANN [12] as an example, although it has the advantage to use a discriminator to encourage the model to learn invariant features,

there is also a bottleneck. When we only rely on the discriminator to control the extraction ability of the domain invariant features, the conditions to extract domain invariant features will become very limited. When the extracted domain invariant features are strong enough to fool the discriminator, it will become difficult for the model to further improve its ability to extract the domain invariant features. In addition, when we only use the source domain labeled data to adjust the classifier, the extracted features will be more biased towards the source domain, which further limits its performance on the target task. This challenge also exists in other methods that employ the domain discriminator to extract the domain invariant features.

Contributions. In order to further improve the extraction ability of domain invariant features, we propose a dual-module architecture to solve this challenge. As shown in Figure 1, The proposed network is composed of a discriminative feature learning module and a domain invariant feature learning module, and the domain discriminative feature module is employed to encourage a domain invariant feature module to learn more domain invariant features. We take DANN [12] as an example to implement our model. We also employ the Maximum Classifier Discrepancy (MCD) [47] to learn the domain discriminative features from the target domain so that the classifier is not only affected by the domain discriminative features of the source domain.

In summary, the main contributions of this paper include:

1. We propose a novel dual-module network architecture to promote the learning of domain invariant features;
2. We propose a new adversarial loss to maximize the discrepancy of the feature distributions while minimizing the discrepancy of the prediction results;
3. We propose to maximize the classifier discrepancy to ameliorate the training imbalance issue of previous unsupervised domain adaptation methods.

We evaluated our proposed method on four popular benchmark datasets and compared its performance with previous work. The proposed structure outperforms the state-of-the-art by a large margin on unsupervised domain adaptation

of digit classification and object recognition. The source code of the proposed model can be accessed at the author's website[1].

2. Related Work

Domain adaptation could be taken as a subset of transfer learning [41], which is a commonly used technique in many computer vision tasks to improve the generalization ability of a model trained on a single domain. In this section, we describe some existing domain adaptation methods.

Learn domain invariant features. Recently, [8] explored what enables the contrastive prediction tasks to learn useful representations. [5] learns the semantic labels in a supervised fashion, and broadens its understanding of the data by learning from self-supervised signals how to solve a jigsaw puzzle on the same images. DICD [29] proposes to learn domain invariant features and class discriminative features simultaneously by a low-dimension latent feature space so that the samples with the same category are close to each other while the samples with different categories are away from each other.

Distance-based methods. Aligning the distribution between the source domain and the target domain is a very common method in solving unsupervised domain adaptation problems. Maximum Mean discrepancy (MMD) [16, 35, 54, 33] is a method of measuring the difference of two distributions. DAN [33] explored the multi-core version of MMD to define the distance between two distributions. JAN [35] learned a transfer network by aligning the joint distributions of multiple domain-specific layers across the domains based on a joint maximum mean discrepancy (JMMD) criterion. [34] enabled the classifier adaptation by plugging several layers into the deep network to explicitly learn the residual function with reference to the target classifier. CMD [61] is a metric on the set of probability distributions on a compact interval.

To solve the problem of unbalanced datasets, Deep Asymmetric Transfer

[1] https://github.com/rucv/DMDA

Network (DATN) [56] proposed to learn a transfer function from the target domain to the source domain and meanwhile adapting the source domain classifier with more discriminative power to the target domain. DeepCORAL [51] builds a specific deep neural network by aligning the distribution of second-order statistics to limit the invariant domain of the top layer. [7] proposed a Higher-order Moment Matching (HoMM) method to minimize the domain discrepancy.

Adversarial methods. Adversarial training is another very effective method to transfer domain information. Inspired by the work of gradient reversal layer [12], a group of domain adaptation methods has been proposed based on adversarial learning. RevGrad [12] proposed to learn the global invariant feature by using a discriminator that is used to reduce the discriminative features in the domain. Deep Reconstruction-Classification Networks (DRCN) [13] jointly learned a shared encoding representation for supervised classification of the labeled source data, and unsupervised reconstruction of the unlabeled target data. [3] extracted image representations that are partitioned into two subspaces. Adversarial Discriminative Domain Adaptation (ADDA) [53] trained two feature extractors for the source and target domains respectively, to generate embeddings to fool the discriminator.

Maximum Classifier Discrepancy (MCD) [47, 60] proposed to explore task-specific decision boundaries. CyCADA [21] introduced a cycle-consistency loss to match the pixel-level distribution. SimeNet [45] solved this problem by learning the domain invariant features and the categorical prototype representations. CAN [24] optimized the network by considering the discrepancy of the intra-class domain and the inter-class domain. Graph Convolutional Adversarial Network (GCAN) [38] realized the unsupervised domain adaptation by jointly modeling data structure, domain label, and class label in a unified deep model. [14] proposed a domain flow generation (DLOW) model to bridge two different domains by generating a continuous sequence of intermediate domains flowing from one domain to the other. [4] employed a variational auto-encoder to reconstruct the semantic latent variables and domain latent variables behind the data. Drop to Adapt (DTA) [26] leveraged adversarial dropout to learn strongly discrimina-

tive features by enforcing the cluster assumption. Instead of representing the classifier as a weight vector, [36] modeled it as a Gaussian distribution with its variance representing the inter-classifier discrepancy.

DMRL [57] utilizes dual mixup regularized learning to improve the class-level prediction in the target domain and boost domain invariant feature extracting. DMRL [57] focuses on data augmentation by generating extra mixed samples to improve the performance of adversarial domain adaptation. Dual modules have demonstrated great effectiveness with adversarial learning in unsupervised domain adaptation. DADA [11] is the first model which introduces a dual mechanism with adversarial learning in domain adaptation. It utilizes two discriminators whose outputs include classification predictions from both source and target domains so that domain-level and class-level adaptation could be considered at the same time in each discriminator. Although there are two discriminators in the model, they share the feature extractor. In the proposed model, the two discriminators have separate feature extractors that are focused on domain invariant features and domain discriminative features, respectively. Different from DMRL and DADA, we designed a new network structure for adversarial domain adaptation, and the proposed model executes the adversarial learning in the M1 module and maximizes the discrepancy of feature distributions between the two modules.

3. Proposed Method

In this paper, we consider the unsupervised domain adaptation problem. Suppose we have a source domain $D_s = \{(X_s, Y_s)\} = \{(x_s^i, y_s^i)\}_{i=1}^{n_s}$ with n_s labeled samples and a target domain $D_t = \{(X_t)\} = \{(x_t^i)\}_{i=1}^{n_t}$ with n_t unlabeled samples. The two domains share the same label space $Y = \{1, 2, 3, ..., K\}$, where K is the number of categories. We assume that the source sample x_s belongs to the source distribution P_s, and the target sample x_t belongs to the target distribution P_t, where $P_s \neq P_t$. Our goal is to train a classifier $f_\theta(x)$ that can minimize the target risk $\epsilon_t = E_{x \in D_t}[f_\theta(x) \neq y_t]$, where $f_\theta(x)$ represents the

output of the deep neural network, and θ represents the model parameters to be learned.

In the following subsections, we will present the main idea of the proposed strategy for unsupervised domain adaptation, the network architecture, and the training steps of the network.

3.1. Main Idea

In the method that employs the domain discriminator to help the feature extractor learn domain invariant features, the space represented by the domain invariant features is a fusion of the source domain and the target domain. We believe that the ideal state of this type of method is to first focus on learning the domain invariant features, and then learn some domain discriminative features from the target domain until the process reaches its upper limit to further improve the performance. When using the domain discriminator as one of the conditions of adversarial training, a big issue is that, with the increase of the model's ability to learn domain invariant features, the ability of the discriminator to give correct judgment to the data will be greatly reduced. When it decays beyond a certain threshold, the model will not be able to further improve the learning ability of domain invariant features. Thus, the domain discriminator has become a bottleneck of learning domain invariant features, restricting the model to reach its upper limit of learning domain invariant features.

To address this challenge, we propose a dual-module structure to further improve the domain invariant feature extraction capabilities of the original algorithm. As shown in Figure 2, one module is designed to learn the domain-invariant features in the data, and the other module is used to learn the domain discriminative features in the data. Then we encourage the two modules to their extremity by constructing a new adversarial loss function. As a result, one module learns more domain-invariant features, and the other module learns more domain discriminative features. We implement the proposed strategy on top of the most representative algorithm DANN [12] that employs a domain discriminator to learn domain invariant features. In the adversarial training

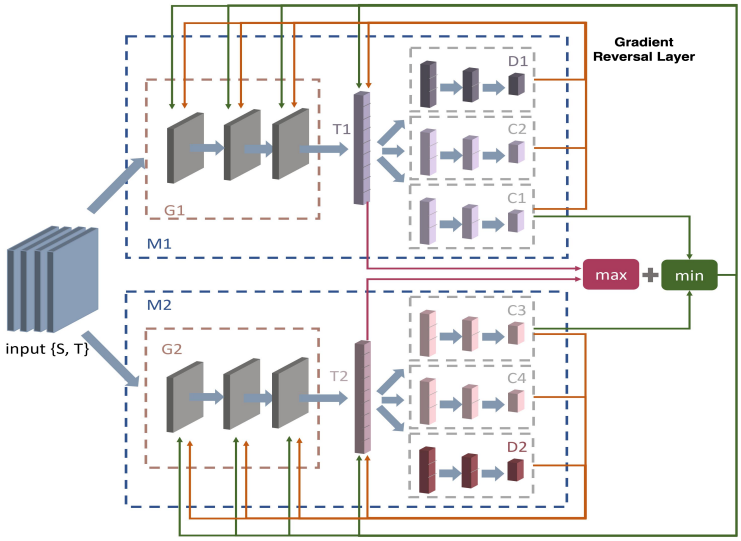


Figure 2: The proposed network architecture has two modules. M1 learns the domain invariant features, and M2 learns the domain discriminative features. G1 is the feature extractor of M1, and G2 is the feature extractor of M2. T1 and T2 are learner transformation layers. D1 and D2 are domain discriminators. C1, C2, C3, and C4 are classifiers. Orange lines denote the backward process of training steps 1 and 2, and Green lines denote the backward process of training step 3. The gradient reversal layer in M1 helps the feature extractor to learn domain invariant features, while M2 does not have the gradient reversal layer so that the feature extractor can learn the domain discriminative features.

of DANN, only the source data is used to adjust the classifier. Therefore, we adopt Maximum Classifier Distance (MCD) [47] to learn the domain discriminative features from the target domain after the model reaches its upper limit to further improve its performance.

Dual-module architecture. In the dual-module architecture, we need to build two modules with completely different functions. As shown in Figure 2, We call the module that can learn the domain invariant features as M1, and the module that can learn domain discriminative features as M2. Different from [10] which employs coupled neural networks for the source domain and target

domain respectively, both modules in our model receive the data from the source domain and the target domain since each module has a discriminator. For the module M1, we can use any algorithm that uses a discriminator to learn domain invariant features. In our experiment, we use DANN [12] as the module M1, since it has a clear and concise structure. For the module M2, the easiest way is to employ the source-only method. As the number of training increases, the ability of the feature extractor to extract domain discriminative features will be greatly improved. In other words, among the features extracted by M2, the domain discriminative features will have a larger ratio than the domain invariant features.

In our experiments, our M2 also follows the same DANN structure. The only difference is that we remove the gradient reversal layer for adversarial training, so the discriminator will make the learned features more discriminative. As a result, when we input the same data into the two modules, the feature distribution from M1 will be based on domain invariant features, and the feature distribution from M2 will be based on domain discriminative features. We designed an adversarial loss for training this dual-module architecture. In the adversarial loss, we expand the discrepancy in feature distribution between the two modules. At the same time, we narrow the discrepancy in the prediction between the two modules. We will provide more details about this in Section 3.4.

Target domain discriminative feature. Maximum Classifier Discrepancy (MCD) [47] is to make the model learn the decision boundary of a specific task by maximizing the prediction difference of the two classifiers. Since the two classifiers are iterated under the same conditions, their prediction results are very similar. However, for some data that is easy to be confused, the results given by the two classifiers may be different. Therefore, if the information captured by the feature extractor is more discriminative, the features that determine the decision boundary will also be larger. We follow this idea and use MCD to optimize our classifier, since the classifier of DANN is only adjusted by the source domain. We add a classifier to each of the two modules, which

means that there are two classifiers with the same structure in each module. We assign these two classifiers to random parameters and train them together. We make our module learn the domain discriminative features from the target domain by learning the decision boundary. This can make up for the lack of the domain discriminative features only from the source domain.

3.2. Network Structure

In this section, we will elaborate on the proposed network structure in detail. As shown in Figure 2, our network employs a dual-module structure. M1 is the domain invariant feature module, and M2 is the domain discriminative feature module. M1 is composed of a feature extractor G1, a discriminator D1, two classifiers C1 and C2, and a Linear transformation layer T1. M2 consists of a feature extractor G2, a discriminator D2, two classifiers C3 and C4, and a Linear transformation layer T2. Since the loss of our dual-module architecture needs to calculate the discrepancy of the two feature distributions, we need an independent linear transformation layer to convert the feature map into a feature distribution. For each module, We embed a linear transformation layer after the feature extractor. For the linear transformation layer, we set its input and output sizes as that of the feature extractor, so as to ensure that there is no information loss in this process.

In addition, the domain discriminators in these two modules have completely different functions. The discriminator D1 plays the same role as the discriminator in DANN to fuse the two domains. However, the discriminator D2 is employed to separate the two domains. Therefore, M1 learns domain invariant features, while M2 learns the domain discriminative features. Please note that the sub-components used in our two modules have exactly the same structure. For example, G1 and G2 have the same structure, D1 and D2 are the same, and C1, C2, C3, and C4 are the same.

3.3. Discrepancy Loss

In this study, we follow the discrepancy loss in [47]. We use the absolute value of the difference between the probability outputs of the two classifiers as

the discrepancy loss:

$$dis(p^1, p^2) = \frac{1}{K} \sum_{k=1}^{K} |p_k^1 - p_k^2| \tag{1}$$

where p^1 and p^2 are the probability outputs of the two classifiers respectively, which are the prediction scores for all the categories, and K is the number of categories, and p_k^1 and p_k^2 are the specific values of their k_{th} category.

3.4. Training Steps

Step 1: Our model learns the decision boundary by using MCD [47]. Learning the decision boundary is essential to learning the discriminative feature. The main reason we put this as the first step is to avoid conflict between the learning process of the decision boundary and the domain invariant feature learning. In order to bring our model closer to the ideal state of domain fusion, our subsequent steps can effectively reduce the redundant domain discriminative features obtained from the decision boundary, thereby reducing the negative impact of the conflict. Following the setting of [47], we fix the number of iterations to 4 for learning the boundary in target domain in all our experiments. Since the function of the linear transformation layer T is to convert the feature map into a feature distribution, our linear transformation layer only updates the parameters when the feature extractor updates the parameters.

Step 2: In this step, we continue to train the two modules separately. Taking the M1 module as an example, we use the adversarial loss of DANN for training. In other words, we train the model according to the training method from the original algorithm. After this step, the two modules begin to have some differences. This step is necessary for the proposed dual-module structure. For the algorithms that do not use MCD for pre-processing, this will be the first step in the entire training process.

For M1, we conduct adversarial training through gradient reversal layer (grl) to learn the domain invariant features.

$$L_C = L_{C1}(f_\theta(X_s), Y_s) + L_{C2}(f_\theta(X_s), Y_s) \tag{2}$$

$$L_{M1} = L_C + grl(L_{D1}(f_\theta(X_s))) + grl(L_{D1}(f_\theta(X_t))) \tag{3}$$

where L_{M1} is the total loss of the whole M1 module, L_C is the total loss of two classifiers, L_{C1}, L_{C2}, and L_{D1} are the cross-entropy loss for the classifiers and discriminator.

M2 does not have the gradient reversal layer, so the discriminator will prompt the feature extractor to learn the domain discriminative features.

$$L_C = L_{C3}(f_\theta(X_s), Y_s) + L_{C4}(f_\theta(X_s), Y_s) \tag{4}$$

$$L_{M2} = L_C + L_{D2}(f_\theta(X_s)) + L_{D2}(f_\theta(X_t)) \tag{5}$$

where L_{M2} is the total loss of the whole M2 module, L_C is the total loss of the two classifiers, L_{C3}, L_{C4}, and L_{D2} are the cross-entropy loss for the classifiers and discriminator.

Step 3: In this step, we conduct an adversarial loss function L for our two modules. The two modules have separate parameters and they do not share parameters with each other. Thus, we intend to maximize the discrepancy of the features extracted by the feature extractors while minimizing the difference of the predicted classification results so that the feature extractors could concentrate on invariant features and discriminative features respectively, while the predicted results would not have much difference. We input the same set of data into the two modules and extract the output from the transformation layer T and the classifier C. We use the linear transformation layer to convert the feature maps generated by the feature extractors into feature distributions, and calculate the discrepancy between the two modules, as illustrated in Figure 2. We use a gradient reversal layer (grl) to maximize the discrepancy of the feature distributions of the two modules.

Although there are two classifiers in each module, we only utilize C1 (from M1) and C3 (from M2) to calculate the discrepancy between the predictions of the two modules. At this time, the discrepancy of the predicted results between

the two modules is minimized so that, although the feature extractors in the two modules focus on invariant features and discriminative features respectively, the predicted results would not have too much discrepancy.

Our adversarial loss function is to play a Min-Max game with these two discrepancies.

$$grl(dis(t)) = \max(dis(t_1^s, t_2^s) + dis(t_1^t, t_2^t)) \tag{6}$$

$$dis(c) = \min(dis(c_1^s, c_3^s) + dis(c_1^t, c_3^t)) \tag{7}$$

$$L = grl(dis(t)) + dis(c) \tag{8}$$

where L is the total loss, $dis()$ is the discrepancy loss. t_1^s means the output is from T1 and the input is from the source domain, and t_2^s means the output from T2 and input from the source domain. t_1^t means the output from T1 and input from the target domain, and t_2^t means the output from T2 and input from the target domain. c_1^s means the probability output from C1 and takes input from the source domain, and c_1^t means the probability output from C1 and takes input from the target domain. c_3^s means the probability output from C3 and takes input from the source domain, and c_3^t means the probability output from C3 and takes input from the target domain.

4. Experiments

We conducted extensive experiments to evaluate the proposed architecture and the effect of different components in the architecture. We conduct our experiments on three digits datasets, two traffic sign datasets, and one object classification dataset. In the following, **OURS** mentioned in the results refers to the case where only the dual-module structure is used. **MCD+DANN** refers to the case where MCD is directly employed to solve the imbalance problem in DANN. **OURS+1M** refers to the case where MCD is only used by the M1

module. **OURS+2M** refers to the case where MCD is used by both M1 and M2 modules. We also compare the performance with DANN and MCD as the baselines.

4.1. Experiments on Digits and Traffic Signs Datasets

In this section, we evaluate our model using the following five datasets: MNIST [25], Street View House Numbers (SVHN) [40], USPS [23], Synthetic Traffic Signs (SYN SIGNS) [39], and the German Traffic Signs Recognition Benchmark (GTSRB) [50].

MNIST: The dataset contains images of digits 0 to 9 in different styles. It is composed of $60,000$ training and $10,000$ testing images.

USPS: This is also a digit dataset with $7,291$ training and $2,007$ testing images.

SVHN: Another digit dataset with $73,257$ training, $26,032$ testing, and $53,1131$ extra training images.

SYN SIGNS: This is a synthetic traffic sign dataset, which contains $100,000$ labeled images, and 43 classes.

GTSRB: A dataset for German traffic signs recognition benchmark. The training set contains $39,209$ labeled images and the test set contains $12,630$ images. It also contains 43 classes.

We evaluate the unsupervised domain adaptation model on the following four transfer scenarios:

- SVHN $\rightarrow$ MNIST
- USPS $\rightarrow$ MNIST
- MNIST $\rightarrow$USPS
- SYNSIG $\rightarrow$ GTSRB

During the experiments, we employ the CNN architecture and the input size used in [47]. We used mini-batch stochastic gradient descent (SGD) to optimize our model and set the learning rate at 0.002 in all experiments. We

Method	**SVHN** to **MNIST**	**MNIST** to **USPS**	**USPS** to **MNIST**	**SYNSIG** to **GTSRB**
Source only	67.1	79.4	63.4	85.1
DANN [12]	71.1	85.1	73.0 ± 0.2	88.7
ADDA [53]	76.0 ± 1.8	-	90.1 ± 0.8	-
CoGAN [32]	-	-	89.1 ± 0.8	-
PixelDA [2]	-	95.9	-	-
ASSC [18]	95.7 ± 1.5	-	-	82.8 ± 1.3
UNIT [31]	90.5	96.0	93.6	-
CyCADA [21]	90.4 ± 0.4	95.6 ± 0.2	96.5 ± 0.1	-
GTA [49]	92.4 ± 0.9	95.3 ± 0.7	90.8 ± 1.3	-
DeepJDOT [1]	96.7	95.7	96.4	-
SimNet [45]	-	96.4	95.6	-
GICT [46]	98.7	96.2	96.6	-
STAR [36]	98.8 ± 0.05	97.8 ± 0.1	97.7 ± 0.05	95.8 ± 0.2
MCD [47]	96.2 ± 0.4	96.5 ± 0.3	94.1 ± 0.3	94.4 ± 0.3
MCD+DANN	91.4 ± 0.2	97.3 ± 0.3	96.8 ± 0.1	90.7 ± 0.2
ours	98.9 ± 0.1	95.1 ± 0.4	96.1 ± 0.2	91.1 ± 0.2
ours+1M	98.3 ± 0.1	97.1 ± 0.2	97.0 ± 0.1	90.8 ± 0.2
ours+2M	$\mathbf{99.3 \pm 0.1}$	$\mathbf{98.0 \pm 0.4}$	$\mathbf{97.7 \pm 0.1}$	$\mathbf{97.0 \pm 0.2}$

Table 1: The performance on digit classification and sign classification. We report the mean and the standard deviation of the accuracy obtained over 5 trials.

follow DANN [12] and employ the SGD training schedule for the part of learning domain invariant feature: the learning rate adjusted by $\eta_p = \frac{\eta_0}{(1+\alpha p)^\beta}$, where p denotes the process of training iterations that is normalized in [0, 1], and we set $\eta_0 = 0.002$, $\alpha = 10$, and $\beta = 0.75$; the hyper-parameter λ is initialized at 0 and is gradually increased to 1 by $\lambda_p = \frac{2}{1+exp(-\gamma p)} - 1$, where we set $\gamma = 10$. For the maximum classifier discrepancy, we set the hyper-parameter $k = 4$ (the number of iterations for learning the boundary in the target domain) in all experiments. We set the batch size to 128 in all experiments. We follow the protocol of unsupervised domain adaptation and do not use validation samples to tune the hyper-parameters.

We present the digit classification and sign classification performance in Table 1. From the table, it is clear that the proposed method outperforms previous models in all settings, where **OURS+2M** is the top-performing variant. In order to explore the direct impact of MCD on DANN, we combined them and compared the experimental results with the state-of-the-art. We can find that the performance of **MCD+DANN** is lower than MCD in both SVHN→MNIST and SYNSIG→GTSRB tasks. The result demonstrates that when MCD acts directly on DANN, it sometimes may cause conflict with DANN, while the structure of **OURS+2M** can effectively resolve this conflict. Compared with MCD (baseline), we obtain an improvement of 3.1% in SVHN→MNIST, 1.5% in MNIST→USPS, 3.6% in USPS→MNIST, and 2.6% in SYNSIG→GTSRB. In addition, our model outperforms the state-of-the-art methods on all tasks.

4.2. Experiments on VisDA Classification Dataset

We further evaluate our model on the large VisDA-2017 dataset [44]. The VisDA-2017 image classification is a 12-class domain adaptation dataset used to evaluate the adaptation from synthetic-object to real-object images. The source domain consists of 152,397 synthetic images, where 3D CAD models are rendered from various conditions. The target domain consists of 55,388 real images taken from the MS-COCO dataset [30].

In this experiment, we employ Resnet-18 [19] as our feature extractor, and

the parameters are adopted from the ImageNet pre-trained model. The pre-trained model of our Resnet-18 comes from Pytorch [42] and all experimental implementations are based on Pytorch.

The input images are of the size 224×224. First, we resize the input image to 256, and then crop the image to 224×224 in the center. When we train the model using only the source domain, we just modify the output size of the original last fully connected layer to a size that conforms to VisDA-2017 [44]. In other tasks, we utilize a three-layer fully connected layer structure to replace the one-layer fully connected layer structure of the original classifier. For algorithms that require a discriminator, we employ a discriminator with a three-layer fully connected layer structure. In order to eliminate the interference factors, except for the source only, all other algorithms use the same classifier and the same discriminator. We uniformly use SGD as the optimizer for training, and use 1×10^{-3} for the learning rate of all methods. We use 64 as the batch size for training.

The results on VisDA-2017 are shown in Table 2. We can see that our model achieves an accuracy much higher than previous methods. In addition, our method performs better than the source only model in all classes, whereas MCD and DANN perform worse than the source only model in some classes such as train, motorcycle, and car. Same as the last experiment, **OURS+2M** is the top-performing variant. On average, we obtain an improvement of 6.1% compared with MCD, and 13.3% compared with DANN. Our model also achieves the best performance in eight out of the twelve categories in this experiment.

4.3. Ablation Study

We conducted ablation studies using the digital and traffic sign datasets with the same unsupervised domain adaptation setting as **Section 4.1**. The proposed algorithm has two modules, and each module has two partial components, namely **dif-module**, **ddf-module**, **dif-MCD** and **ddf-MCD**. The **dif-module** refers to the component used to learn domain invariant features in the model. This is a necessary module in the algorithm and also in our baseline.

Method	plane	bcycl	bus	car	horse	knife	mcycl	person	plant	sktbrd	train	truck	mean
Source Only	30.2	4.3	27.9	60.6	31.0	2.1	82.7	7.4	67.7	12.9	79.4	2.3	40.3
DANN [12]	72.3	53.1	64.7	31.8	58.2	14.3	80.7	**60.0**	70.0	41.4	**89.7**	20.7	55.9
MCD [47]	82.2	18.7	86.6	**62.1**	73.6	**41.0**	89.1	58.9	80.7	64.3	74.2	12.5	63.1
OURS	83.6	60.0	64.1	56.4	65.8	12.5	91.4	39.3	66.7	55.0	78.3	31.2	60.1
OURS+1M	85.2	58.5	76.5	47.1	73.5	24.7	89.3	58.9	75.0	62.2	80.1	**32.0**	63.4
OURS+2M	**86.0**	**61.5**	**88.3**	61.6	**83.8**	6.7	**92.9**	56.8	**89.9**	**68.8**	87.3	23.0	**69.2**

Table 2: Results of unsupervised domain adaptation on VisDA2017 [44] image classification track. The accuracy is obtained by fine-tuning ResNet-18 [19] model pre-trained on ImageNet [9]. This task evaluates the adaptation capability from synthetic CAD model images to real-world MS COCO [30] images. Our model achieves the best performance in most categories.

dif-Module	ddf-Module	dif-MCD	ddf-MCD	S→M	M→U	U→M	SIG→GTS	Avg
√				71.1	85.1	73.0 ± 0.2	88.7	79.5
√		√		91.4 ± 0.2	97.3 ± 0.3	96.8 ± 0.1	90.7 ± 0.2	94.1
√	√			98.9 ± 0.1	95.1 ± 0.4	96.1 ± 0.2	91.1 ± 0.2	95.3
√	√	√		98.3 ± 0.1	97.1 ± 0.2	97.0 ± 0.1	90.8 ± 0.2	95.8
√	√	√	√	$\mathbf{99.3 \pm 0.1}$	$\mathbf{98.0 \pm 0.4}$	$\mathbf{97.7 \pm 0.1}$	$\mathbf{97.0 \pm 0.2}$	**98.0**

Table 3: Ablation study of our method for unsupervised domain adaptation on digital and traffic sign datasets.

The **ddf-module** refers to the component used to learn domain discriminative features in the model. The function of this module is to expand the discrepancy in feature distribution during training. The **dif-MCD** refers to the use of MCD to align the class distribution of domain invariant feature module classifiers, and the **ddf-MCD** refers to the use of MCD to align the class distribution of domain discriminative feature module classifiers. Therefore, we design the ablation study to test the influence of each component on the overall algorithm performance.

As shown in Table 3, the performance of dual-module adversarial training has a significant improvement over using a single domain adaptation method. There are two most intuitive examples. (i) The performance of $DANN$ with dual-module adversarial training is 18.5% higher than the one without dual-

module adversarial training. (ii) The performance of $DANN + MCD$ with dual-module adversarial training is 3.9% higher than the original single module.

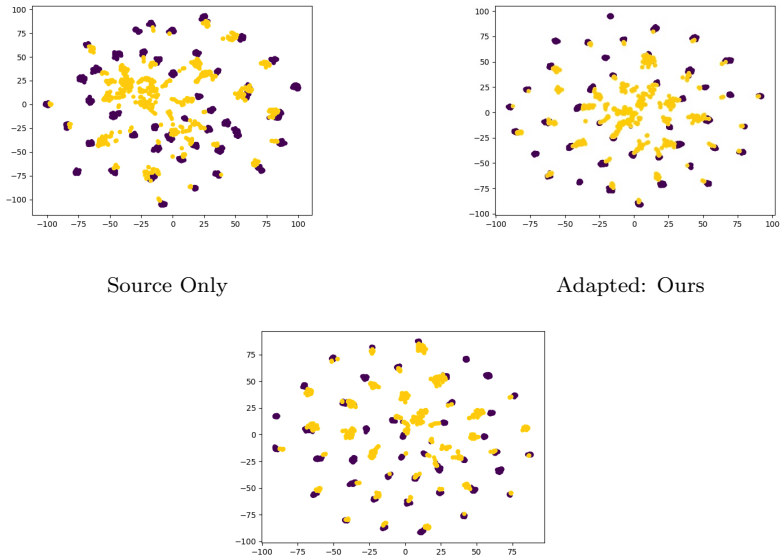
Although the dual-module architecture costs more time in training, only one module is utilized for inference. More specifically, only G1, T1, and C1 are employed in our dual-module structure module and its variants. The introduced transformation layer T1 is a lightweight module and introduces almost negligible overhead. Thus, no matter the architecture is one module or dual-module, the inference time is almost the same. The second module in the dual-module system assists the first module to better learn the domain invariant features and lift the upper limit of the quality of the learned features.

4.4. Visualization

In Figure 3 , we use T-SNE [55] to visualize the models we trained. We choose the task that transfer SYN SIGNS [39] to GTSRB [50]. SYN SIGNS is the source domain, and GTSRB is the target domain. After training, we select 2000 data for visualization, where 1000 images from the source domain and 1000 images from the target domain. It can be clearly seen from the result that, compared to the Source Only method, our proposed model has a significant effect on reducing the domain shift, especially for the **OURS+M2** variant.

5. Conclusion and Future Work

In this paper, we have proposed a dual-module network architecture that can strongly encourage domain invariant feature learning. The network architecture is composed of a discriminative feature learning module and a domain invariant feature learning module. We have also proposed an adversarial loss function using the difference between the feature distributions of the two modules and the similarity of their predicted results. The two modules will compete with each other to maximize the difference in feature distribution. The proposed model employs the maximum classifier discrepancy to solve the imbalance problem of domain discriminative feature extraction in the target domain for the two modules. Extensive experiments demonstrate that the proposed method achieves



Source Only

Adapted: Ours

Adapted: Ours+2M

Figure 3: Visualization by using T-SNE [55]. We take 2000 images from the task SYN SIGNS → GTSRB. Through visualization, we can easily find that our proposed method can fuse the source domain with the target domain well.

state-of-the-art performance on the standard unsupervised domain adaptation benchmarks and significantly improves its performance.

The proposed dual-module strategy can be extended to other UDA models as long as the UDA models adopt adversarial learning. For other UDA models, we can design dual modules like this with one module employing adversarial training for invariant features and another module exploiting regular training for discriminative features. Finally, we can adopt adversarial training between the two modules to further improve the learning ability for invariant features. We are currently working on the applications for other UDA tasks like object detection and segmentation.

Acknowledgement

This work was partly supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) under grant no. RGPIN-2021-04244, and the National Aeronautics and Space Administration (NASA) under grant no. 80NSSC20M0160.

References

[1] B. Bhushan Damodaran, B. Kellenberger, R. Flamary, D. Tuia, and N. Courty, "Deepjdot: Deep joint distribution optimal transport for unsupervised domain adaptation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 447–463.

[2] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan, "Unsupervised pixel-level domain adaptation with generative adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 3722–3731.

[3] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan, "Domain separation networks," in *Advances in neural information processing systems*, 2016, pp. 343–351.

[4] R. Cai, Z. Li, P. Wei, J. Qiao, K. Zhang, and Z. Hao, "Learning disentangled semantic representation for domain adaptation," in *IJCAI: proceedings of the conference*, vol. 2019. NIH Public Access, 2019, p. 2060.

[5] F. M. Carlucci, A. D'Innocente, S. Bucci, B. Caputo, and T. Tommasi, "Domain generalization by solving jigsaw puzzles," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[6] F. Cen, X. Zhao, W. Li, and G. Wang, "Deep feature augmentation for occluded image classification," *Pattern Recognition*, vol. 111, p. 107737, 2021.

[7] C. Chen, Z. Fu, Z. Chen, S. Jin, Z. Cheng, X. Jin, and X.-S. Hua, "Homm: Higher-order moment matching for unsupervised domain adaptation," *order*, vol. 1, no. 10, p. 20, 2020.

[8] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," *arXiv preprint arXiv:2002.05709*, 2020.

[9] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *CVPR09*, 2009.

[10] Z. Ding, N. M. Nasrabadi, and Y. Fu, "Semi-supervised deep domain adaptation via coupled neural networks," *IEEE Transactions on Image Processing*, vol. 27, no. 11, pp. 5214–5224, 2018.

[11] Y. Du, Z. Tan, Q. Chen, X. Zhang, Y. Yao, and C. Wang, "Dual adversarial domain adaptation," *arXiv preprint arXiv:2001.00153*, 2020.

[12] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *International conference on machine learning*, 2015, pp. 1180–1189.

[13] M. Ghifary, W. B. Kleijn, M. Zhang, D. Balduzzi, and W. Li, "Deep reconstruction-classification networks for unsupervised domain adaptation," in *European Conference on Computer Vision*. Springer, 2016, pp. 597–613.

[14] R. Gong, W. Li, Y. Chen, and L. V. Gool, "Dlow: Domain flow for adaptation and generalization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2477–2486.

[15] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger,

Eds. Curran Associates, Inc., 2014, pp. 2672–2680. [Online]. Available: http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf

[16] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola, "A kernel two-sample test," *The Journal of Machine Learning Research*, vol. 13, no. 1, pp. 723–773, 2012.

[17] J. Gui, Z. Sun, Y. Wen, D. Tao, and J. Ye, "A review on generative adversarial networks: Algorithms, theory, and applications," *arXiv preprint arXiv:2001.06937*, 2020.

[18] P. Haeusser, T. Frerix, A. Mordvintsev, and D. Cremers, "Associative domain adaptation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2765–2773.

[19] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[20] L. He, J. Lu, G. Wang, S. Song, and J. Zhou, "Sosd-net: Joint semantic object segmentation and depth estimation from monocular images," *Neurocomputing*, vol. 440, pp. 251–263, 2021.

[21] J. Hoffman, E. Tzeng, T. Park, J.-Y. Zhu, P. Isola, K. Saenko, A. Efros, and T. Darrell, "Cycada: Cycle-consistent adversarial domain adaptation," in *International conference on machine learning*, 2018, pp. 1989–1998.

[22] L. Hu, M. Kan, S. Shan, and X. Chen, "Duplex generative adversarial network for unsupervised domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[23] J. J. Hull, "A database for handwritten text recognition research," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 5, pp. 550–554, 1994.

[24] G. Kang, L. Jiang, Y. Yang, and A. G. Hauptmann, "Contrastive adaptation network for unsupervised domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4893–4902.

[25] Y. LeCun and C. Cortes, "MNIST handwritten digit database," 2010. [Online]. Available: http://yann.lecun.com/exdb/mnist/

[26] S. Lee, D. Kim, N. Kim, and S.-G. Jeong, "Drop to adapt: Learning discriminative features for unsupervised domain adaptation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 91–100.

[27] K. Li, M. I. Fathan, K. Patel, T. Zhang, C. Zhong, A. Bansal, A. Rastogi, J. S. Wang, and G. Wang, "Colonoscopy polyp detection and classification: Dataset creation and comparative evaluations," *arXiv preprint arXiv:2104.10824*, 2021.

[28] K. Li, N. Y. Wang, Y. Yang, and G. Wang, "Sgnet: A super-class guided network for image classification and object detection," *arXiv preprint arXiv:2104.12898*, 2021.

[29] S. Li, S. Song, G. Huang, Z. Ding, and C. Wu, "Domain invariant and class discriminative feature learning for visual domain adaptation," *IEEE Transactions on Image Processing*, vol. 27, no. 9, pp. 4260–4273, 2018.

[30] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.

[31] M.-Y. Liu, T. Breuel, and J. Kautz, "Unsupervised image-to-image translation networks," in *Advances in neural information processing systems*, 2017, pp. 700–708.

[32] M.-Y. Liu and O. Tuzel, "Coupled generative adversarial networks," in *Advances in neural information processing systems*, 2016, pp. 469–477.

[33] M. Long, Y. Cao, J. Wang, and M. Jordan, "Learning transferable features with deep adaptation networks," in *International conference on machine learning*, 2015, pp. 97–105.

[34] M. Long, H. Zhu, J. Wang, and M. I. Jordan, "Unsupervised domain adaptation with residual transfer networks," in *Advances in neural information processing systems*, 2016, pp. 136–144.

[35] ——, "Deep transfer learning with joint adaptation networks," in *International conference on machine learning*. PMLR, 2017, pp. 2208–2217.

[36] Z. Lu, Y. Yang, X. Zhu, C. Liu, Y.-Z. Song, and T. Xiang, "Stochastic classifiers for unsupervised domain adaptation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 9111–9120.

[37] W. Ma, Y. Wu, Z. Wang, and G. Wang, "Mdcn: Multi-scale, deep inception convolutional neural networks for efficient object detection," in *2018 24th International Conference on Pattern Recognition (ICPR)*. IEEE, 2018, pp. 2510–2515.

[38] X. Ma, T. Zhang, and C. Xu, "Gcan: Graph convolutional adversarial network for unsupervised domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8266–8276.

[39] B. Moiseev, A. Konev, A. Chigorin, and A. Konushin, "Evaluation of traffic sign recognition methods trained on synthetically generated data," in *Advanced Concepts for Intelligent Vision Systems*, J. Blanc-Talon, A. Kasinski, W. Philips, D. Popescu, and P. Scheunders, Eds. Cham: Springer International Publishing, 2013, pp. 576–583.

[40] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," 2011.

[41] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, pp. 1345–1359, 2009.

[42] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.

[43] K. Patel, A. M. Bur, and G. Wang, "Enhanced u-net: A feature enhancement network for polyp segmentation," *arXiv preprint arXiv:2105.00999*, 2021.

[44] X. Peng, B. Usman, N. Kaushik, J. Hoffman, D. Wang, and K. Saenko, "Visda: The visual domain adaptation challenge," 2017.

[45] P. O. Pinheiro, "Unsupervised domain adaptation with similarity learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8004–8013.

[46] C. Qin, L. Wang, Y. Zhang, and Y. Fu, "Generatively inferential co-training for unsupervised domain adaptation," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2019, pp. 0–0.

[47] K. Saito, K. Watanabe, Y. Ushiku, and T. Harada, "Maximum classifier discrepancy for unsupervised domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3723–3732.

[48] U. Sajid, W. Ma, and G. Wang, "Multi-resolution fusion and multi-scale input priors based crowd counting," in *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE, 2021, pp. 5790–5797.

[49] S. Sankaranarayanan, Y. Balaji, C. D. Castillo, and R. Chellappa, "Generate to adapt: Aligning domains using generative adversarial networks,"

in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8503–8512.

[50] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "The German Traffic Sign Recognition Benchmark: A multi-class classification competition," in *IEEE International Joint Conference on Neural Networks*, 2011, pp. 1453–1460.

[51] B. Sun and K. Saenko, "Deep coral: Correlation alignment for deep domain adaptation," in *European conference on computer vision*. Springer, 2016, pp. 443–450.

[52] L. Tran, X. Yin, and X. Liu, "Disentangled representation learning gan for pose-invariant face recognition," in *In Proceeding of IEEE Computer Vision and Pattern Recognition*, Honolulu, HI, July 2017.

[53] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7167–7176.

[54] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell, "Deep domain confusion: Maximizing for domain invariance," *arXiv preprint arXiv:1412.3474*, 2014.

[55] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008. [Online]. Available: http://www.jmlr.org/papers/v9/vandermaaten08a.html

[56] D. Wang, P. Cui, and W. Zhu, "Deep asymmetric transfer network for unbalanced domain adaptation." in *AAAI*, 2018, pp. 443–450.

[57] Y. Wu, D. Inkpen, and A. El-Roby, "Dual mixup regularized learning for adversarial domain adaptation," in *European Conference on Computer Vision*. Springer, 2020, pp. 540–555.

[58] W. Xu, C. Long, R. Wang, and G. Wang, "Drb-gan: A dynamic resblock generative adversarial network for artistic style transfer," *arXiv preprint arXiv:2108.07379*, 2021.

[59] W. Xu, G. Wang, A. Sullivan, and Z. Zhang, "Towards learning affine-invariant representations via data-efficient cnns," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2020, pp. 904–913.

[60] Y. Yang, T. Kim, and G. Wang, "Multiple classifiers based maximum classifier discrepancy for unsupervised domain adaptation," *arXiv preprint arXiv:2108.00610*, 2021.

[61] W. Zellinger, T. Grubinger, E. Lughofer, T. Natschläger, and S. Saminger-Platz, "Central moment discrepancy (cmd) for domain-invariant representation learning," *arXiv preprint arXiv:1702.08811*, 2017.

[62] X. Zhang, T. Zhang, Y. Yang, Z. Wang, and G. Wang, "Real-time golf ball detection and tracking based on convolutional neural networks," in *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2020, pp. 2808–2813.

[63] Z. Zhang, Y. Wu, and G. Wang, "Bpgrad: Towards global optimality in deep learning via branch and pruning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3301–3309.