FixBi: Bridging Domain Spaces for Unsupervised Domain Adaptation

Jaemin Na[1], Heechul Jung[2], Hyung Jin Chang[3], and Wonjun Hwang[1]
[1]Ajou University, Korea, [2]Kyungpook National University, Korea, [3]University of Birmingham, UK
osial46@ajou.ac.kr, heechul@knu.ac.kr, h.j.chang@bham.ac.uk, wjhwang@ajou.ac.kr

Abstract

Unsupervised domain adaptation (UDA) methods for learning domain invariant representations have achieved remarkable progress. However, most of the studies were based on direct adaptation from the source domain to the target domain and have suffered from large domain discrepancies. In this paper, we propose a UDA method that effectively handles such large domain discrepancies. We introduce a fixed ratio-based mixup to augment multiple intermediate domains between the source and target domain. From the augmented-domains, we train the source-dominant model and the target-dominant model that have complementary characteristics. Using our confidence-based learning methodologies, e.g., bidirectional matching with high-confidence predictions and self-penalization using low-confidence predictions, the models can learn from each other or from its own results. Through our proposed methods, the models gradually transfer domain knowledge from the source to the target domain. Extensive experiments demonstrate the superiority of our proposed method on three public benchmarks: Office-31, Office-Home, and VisDA-2017. [1]

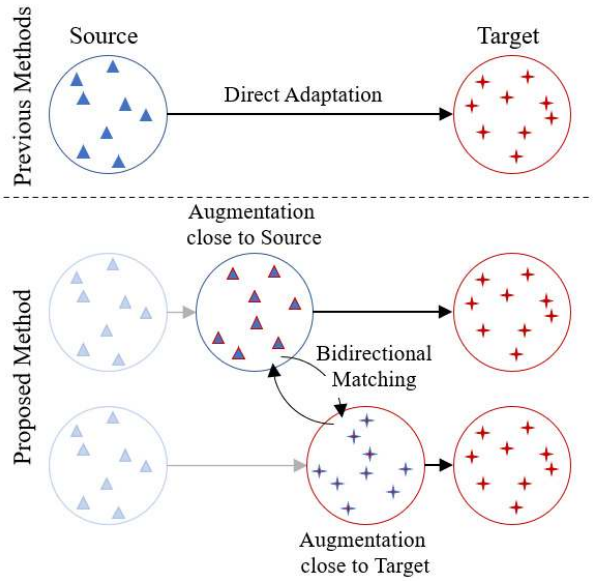


Figure 1. Comparison of previous domain adaptation methods and our proposed method. **Top:** Previous methods try to adapt directly without any consideration of large domain discrepancies. **Bottom:** Our proposed method utilize augmented domains between the source and target domain for efficient domain adaptation.

1. Introduction

Recently, we have seen considerable improvements in several computer vision applications using deep learning; however, this success has been limited to supervised learning methods with abundant labeled data. Collecting and labeling data from various domains is an expensive and time-consuming task. To address this problem, semi-supervised learning [45, 3, 34] and unsupervised learning [9] have been studied; however, in most cases, it was assumed that learning of the model occurred in a similar domain.

UDA refers to a set of transfer learning methods for transferring knowledge learned from the source domain to the target domain under the assumption of domain discrepancy. Moreover, it is useful when the source domain contains enough labeled data to learn, but not much labeled data are present in the target domain. Domain adaptation (DA) generally assumes that the two domains have the same conditional distribution, but different marginal distributions. Under these assumptions, effective knowledge transfer is difficult when the two domains have large marginal distribution gaps. This becomes much more challenging in a scenario where the target domain has no labeled data at all.

In previous UDA methods, a domain discriminator [8, 37] was introduced to encourage domain confusion through domain-adversarial objectives and minimize the gap between the source and target distributions. In [21, 24], domain discrepancy based approaches used metrics such as maximum mean discrepancy (MMD) and joint MMD (JMMD) to reduce the difference between two feature

[1]Our code is available at https://github.com/NaJaeMin92/FixBi.

spaces. Moreover, inspired by the generative adversarial network (GAN), GAN-based DA methods [15, 7] have attempted to generate transferable representations to minimize domain discrepancy. Most of these studies have directly adapted the knowledge learned from the source domain to the target domain. However, fundamentally, this does not take into account the case where the distance between the source and target domain is large, as shown in Figure 1.

In this paper, our goal is to compensate efficiently for the large domain discrepancies. To address this challenge, we construct multiple intermediate augmented domains, whose characteristics are different and complementary to each other. To achieve this, we propose a fixed ratio-based mixup. Our proposed mixup approach minimizes the domain randomness of [41, 43] between the source and target samples and generates multiple intermediate domains, as shown in Figure 1. For example, an augmented domain close to the source domain has more reliable label information, but it has a lower correlation with the target domain. By contrast, label information in an augmented domain close to the target domain is relatively inaccurate, but the similarity to the target domain is much higher.

In these augmented domains, we train the complementary models that teach each other to bridge between the source and target domain. Specifically, we introduce a bidirectional matching based on the high-confidence predictions of each model for the target samples, moving the intermediate domains to the target domain. We also apply self-penalization, which penalizes its own model to improve performance through self-training. Moreover, to properly impose the characteristics of models that change with each iteration, we use an adaptive threshold by the confidence distribution of each mini-batch, not a predefined one [12, 44]. Finally, to prevent divergence of the augmented models generated in different domains, we propose a consistency regularization using an augmented domain with the same ratio of source and target samples.

We conduct extensive ablation studies for a detailed analysis of our proposed method and achieve comparable performance to state-of-the-art methods in standard DA benchmarks such as Office-31 [29], Office-Home [40], and VisDA-2017 [27]. The main contributions of this paper are summarized as follows.

- We propose a fixed ratio-based mixup to efficiently bridges the source and target domains utilizing the intermediate domains.
- We propose confidence-based learning methodologies: a bidirectional matching and a self-penalization using positive and negative pseudo-labels, respectively.
- We empirically validate the superiority of our method to UDA with extensive ablation studies and evaluations on three standard benchmarks.

2. Related Work

Semi-supervised Learning. Semi-supervised learning (SSL) [2, 3, 18, 33, 45, 34, 6] leverages unlabeled data to improve a model's performance when limited labeled data is provided, which alleviates the expensive labeling process efficiently. Many recently proposed semi-supervised learning methods, such as MixMatch [3], FixMatch [34], and ReMixMatch [2], based on augmentation viewpoints. MixMatch [3] used low-entropy labels for data-augmented unlabeled instances and mixed labeled and unlabeled data for semi-supervised learning. On the basis of consistency regularization and pseudo-labeling, FixMatch [34] generates pseudo-labels using the model's predictions on weakly augmented unlabeled images. Then, when the examples have high-confidence predictions, they train the model using strong-augmented images. Note that in general, they assumed that labeled and unlabeled data have similar domains or feature distributions.

Basically, semi-supervised domain adaptation has more information about some target labels compared with UDA, and some related works [1, 30, 28, 19, 44] have been proposed leveraging semi-supervised signals. Specifically, in [30], a minimax entropy approach was proposed that adversarially optimizes an adaptive few-shot model. In [28], the learning of opposite structures was unified whereby it consists of a generator and two classifiers trained with opposite forms of losses for a unified object.

Meanwhile, [44] addresses semi-supervised domain adaptation by breaking it down into SSL and UDA problems. Two models are in charge of each sub-problem and are trained based on co-teaching. One model is trained with labeled source samples and labeled target samples, and the other model is trained with unlabeled target samples and labeled target samples. In this way, by using different combinations of data, it provides two different perspectives. By contrast, in this paper, we guarantee two different perspectives with the two types of our fixed ratio-based mixup. In addition, [44] used co-teaching [12] concepts to train the mixup objectives between the source and target domain, whereas we use this concept to train the pseudo-labels in the target domain with bidirectional matching. Furthermore, when applying the mixup operation, [44] uses only selected target samples, whereas we use all target samples.

Unsupervised Domain Adaptation. Recent works [10, 35, 38, 24, 8, 37, 31, 43] have focused on UDA based on domain alignment and discriminative domain-invariant feature learning methods. For example, a deep adaptation network (DAN) [21] minimized MMD over domain-specific layers, and joint adaptation networks [24] aligned the joint distributions of domain-specific layers across different domains

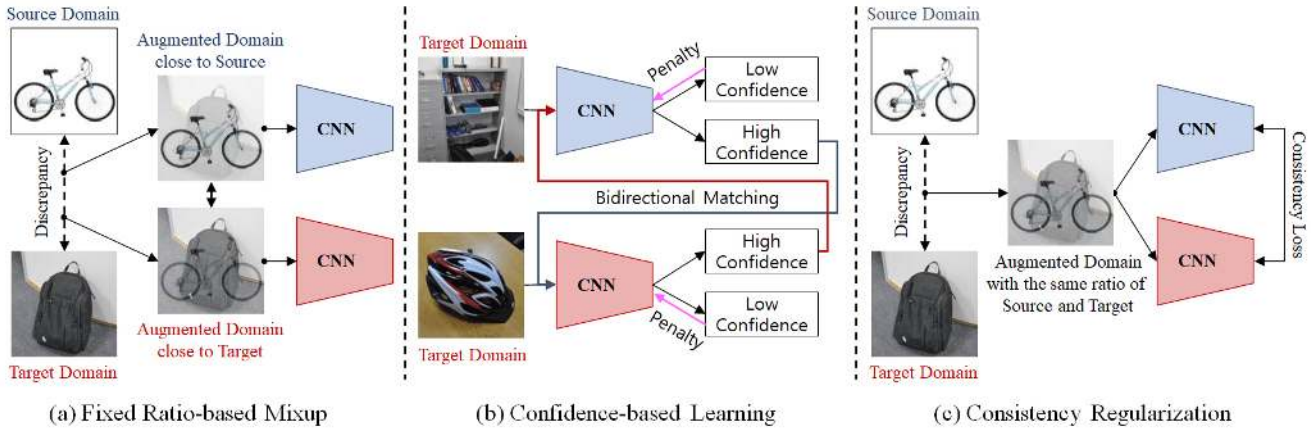


Figure 2. **An overview of the proposed method.** The proposed method consists of (a) fixed ratio-based mixup, (b) confidence-based learning, e.g., bidirectional matching with the positive pseudo-labels and self-penalization with the negative pseudo-labels, and (c) consistency regularization. Best viewed in color.

based on a JMMD. Deep domain confusion (DCC) [38] made use of MMD metrics in the fully connected layer for learning both discriminative and transferable domains. A domain adversarial neural network (DANN) [8] learned a domain invariant representation by back-propagating the reverse gradients of the domain classifier. Adversarial discriminative domain adaptation (ADDA) [37] learned a discriminative representation using the source labels, and then, a separate encoding that maps the target data to the same space based on a domain-adversarial loss is used. Maximum classification discrepancy (MCD) [32] tried to align the distribution of a target domain by considering task-specific decision boundaries by maximizing the discrepancy on the target samples and then generating features that minimize this discrepancy. A contrastive adaptation network (CAN) [17] optimized the metric for minimizing the domain discrepancy, which explicitly models the intra-class domain discrepancy and the inter-class domain discrepancy.

Robust spherical domain adaptation (RSDA) [11] used a spherical classifier for label prediction and a spherical domain discriminator for discriminating domain labels and utilized robust pseudo-label loss in the spherical feature space. Structurally regularized deep clustering (SRDC) [36] enhanced target discrimination by clustering intermediate network features and structural regularization with soft selection of less divergent source examples. Dual mixup regularized learning (DMRL) [41] guided the classifier to enhance consistent predictions between samples and enriched the intrinsic structures of the latent space. For mixing the source and target domains, they proposed two mixup regularizations based on randomness.

Note that in this study, we create bridges between the target and source domain by augmenting multiple intermediate domains. For this purpose, unlike [41, 43], the scope of the augmented domain was not expanded simply by relying on randomness. However, two fixed ratio-based mixups are used to create a source-closed augmented domain, which has a clear label but is at a distance from the target domain, and a target-closed augmented domain, which has the opposite properties. Then, they teach each other in order to transfer domain knowledge to the target side.

3. Proposed Method

In UDA, we are given labeled data $\mathcal{X}^s = \{(x_i^s, y_i^s)\}_{i=1}^{N_s}$ from the source domain and unlabeled data $\mathcal{X}^t = \{(x_i^t)\}_{i=1}^{N_t}$ from the target domain where the N_s and N_t denote the sizes of $\mathcal{X}^s$ and $\mathcal{X}^t$, respectively. The large distribution gap between $P(\mathcal{X}^s)$ and $P(\mathcal{X}^t)$ is one of the major obstacles for the UDA problem. Our goal is to ensure that the knowledge learned from the source domain is well generalized in the target domain. In this section, we introduce our FixBi algorithm and the detailed ideas it builds on, as shown in Figure 2.

3.1. Fixed Ratio-based Mixup

In general, the mixup [46] is a kind of data augmentation method to increase the robustness of neural networks when learning from corrupt labels. Recent studies [3, 34] have utilized the mixup to construct virtual samples with convex combinations between labeled and unlabeled data. In this context, most domain adaptation methods [26, 43, 44, 41] based on the mixup use mixup-ratio λ with randomly sampled values from the beta distribution: $\lambda \sim Beta(\alpha, \alpha)$ where α is a hyperparameter. It is because they have tried to generate training samples that exist somewhere between the source and target domain without any consideration of the domain gap. However, we propose to use two fixed mixup ratios λ_{sd} and λ_{td} to provide more clarity and less random-

ness. Given a pair of input samples and their corresponding one-hot labels in the source and target domain: (x_i^s, y_i^s) and $(x_i^t, \hat{y}_i^t)$, our mixup settings are defined as follows:

$$\begin{aligned} \tilde{x}_i^{st} &= \lambda x_i^s + (1-\lambda) x_i^t \\ \tilde{y}_i^{st} &= \lambda y_i^s + (1-\lambda) \hat{y}_i^t, \end{aligned} \tag{1}$$

where $\lambda \in \{\lambda_{sd}, \lambda_{td}\}$ $s.t.$ $\lambda_{sd} + \lambda_{td} = 1$. Note that $\hat{y}_i^t$ is the pseudo-labels obtained by the baseline model, e.g., DANN [8] or MSTN [42], for the unlabeled target samples. The detailed analysis of our fixed ratio-based mixup is covered in Section 4.2.

Taking advantage of the fixed ratio-based mixup, we construct two network models that act as bridges between the source and target domain. The key point here is to obtain two networks with different perspectives through our mixup strategies. For this purpose, we leverage two different models made by the proposed mixup ratios λ_{sd} and λ_{td}: *"source-dominant model"* (SDM) and *"target-dominant model"* (TDM). The source-dominant model has strong supervision for the source domain but relatively weak supervision for the target domain. By contrast, the target-dominant model has strong target supervision but weak source supervision. As both types of mixups are not confined to a single domain, they can serve as bridges between the two different domains.

Consequently, we apply two fixed ratios λ_{sd} for SDM and λ_{td} for TDM. Let $p(y|\tilde{x}_i^{st})$ denote the predicted class distribution produced by the model for an input $\tilde{x}_i^{st}$. Then the objective of our fixed ratio-based mixup is defined as follows:

$$\mathcal{L}_{fm} = \frac{1}{B} \sum_{i=1}^{B} \hat{y}_i^{st} \log(p(y|\tilde{x}_i^{st})), \tag{2}$$

where $\hat{y}_i^{st} = \operatorname{argmax} p(y|\tilde{x}_i^{st})$ and B is a mini-batch size.

3.2. Confidence-based Learning

Through our fixed ratio-based mixup, the two networks have different characteristics and can develop with mutually complementary learning. To utilize the two models as bridges from the source domain to the target domain, we propose a confidence-based learning where one model teaches the other model using the positive pseudo-labels or teach itself using the negative pseudo-labels.

Bidirectional Matching with positive pseudo-labels. Inspired by [34, 12, 4], when one network assigns the class probability of input above a certain threshold τ, we assume that this predicted label as a pseudo-label. Here, we refer to these labels as positive pseudo-labels. Then we train the peer network to make its predictions match these positive pseudo-labels via a standard cross-entropy loss. Let us denote probability distributions p and q of two models. Then the objective of our bidirectional matching is defined as follows:

$$\mathcal{L}_{bim} = \frac{1}{B} \sum_{i=1}^{B} \mathbb{1}(max(p(y|x_i^t) > \tau) \hat{y}_i^t \log(q(y|x_i^t)), \tag{3}$$

where $\hat{y}_i^t = \operatorname{argmax} p(y|x_i^t)$. Note that in [34], only one-way matching is used according to input augmentations. However, since our method derives the results from both networks for the same input, bidirectional matching is available.

Self-penalization with negative pseudo-labels. As well as the bidirectional matching that matches the positive pseudo-labels to the predictions of the peer network, each network learns through the self-penalization using the negative pseudo-labels. Here, the negative pseudo-label indicates the most confident label (*top-1* label) predicted by the network with a confidence lower than the threshold τ. Since the negative pseudo-label is unlikely to be a correct label, we need to increase the probability values of all other classes except for this negative pseudo-label. Therefore, we optimize the output probability corresponding to the negative pseudo-label to be close to zero. The objective of self-penalization is defined as follows:

$$\mathcal{L}_{sp} = \frac{1}{B} \sum_{i=1}^{B} \mathbb{1}(max(p(y|x_i^t) < \tau) \hat{y}_i^t \log(1 - p(y|x_i^t)). \tag{4}$$

Unlike the recent studies [12, 34, 44] that ignore the low-confidence predictions (or large loss samples), it is worth noting that we utilize the low-confidence predictions as the meaningful knowledges for learning the models. Furthermore, we apply the learnable temperature of the softmax to adjust the output distributions.

Looking back to the confidence threshold τ, the basic strategy is to set a fixed value as a hyperparameter. However, a deep neural network (DNN) tends to start at a low level of confidence value and its value gradually increases as the network learns. The fixed threshold cannot properly reflect the confidence which is changed constantly during training, therefore the number of positive and negative pseudo-labels can be biased to one side. To overcome this problem, we adopt an adaptive threshold which is changed adaptively by the sample mean and standard deviation of a mini-batch, not a fixed one.

3.3. Consistency Regularization

Through our confidence-based learning, the two models with different characteristics gradually get closer to the target domain because they are trained with more reliable pseudo-labels of the target samples. We introduced a new consistency regularization to ensure a stable convergence of training both models. Here, we assume that the well-trained

models should be regularized to have consistent results in the same space. It helps to construct the domain bridging by ensuring that the two models trained from the different domain spaces maintain consistency in the same area between the source and target domain. For the intermediate space, both fixed mixup-ratios λ_{sd} and λ_{td} are set to 0.5. The consistency regularization loss can be defined as follows:

$$\mathcal{L}_{cr} = \frac{1}{B}\sum_{i=1}^{B} \|p(y|\tilde{x}_i^{st}) - q(y|\tilde{x}_i^{st})\|_2^2. \quad (5)$$

Algorithm 1: FixBi Training Procedure

Input : Network weights w_{sd} and w_{td}, total epochs E, mini batch B, warm-up epochs k, mixup-ratios λ_{sd}, λ_{td}, and $\lambda_{cr}(=0.5)$, source samples x^s, target samples x^t, and mixup samples $\tilde{M}$.

```
for e=1 to E do
    for i=1 to B do
        Obtain M̃_sd using Eq. (1) with λ_sd;
        Obtain M̃_td using Eq. (1) with λ_td;
        Update L_fm(M̃_sd; w_sd) and L_sp(x^t; w_sd);
        Update L_fm(M̃_td; w_td) and L_sp(x^t; w_td);
        if e > k then
            if max(y|x^t; w_td) > τ_td then
                Update L_bim(x^t; w_sd);
            end
            if max(y|x^t; w_sd) > τ_sd then
                Update L_bim(x^t; w_td);
            end
            Obtain M̃_cr using Eq. (1) with λ_cr;
            Update L_cr(M̃_cr; w_sd);
            Update L_cr(M̃_cr; w_td);
        end
    end
end
```

Output: Learned model parameters w_{sd} and w_{td}.

3.4. Training Procedure

The training process of our FixBi is summarized in Algorithm 1. First, we start to train our networks with pre-trained baseline weights, similar to [11]. Then, we copy the pre-trained weights to w_{sd} and w_{td}. In each iteration, we generate two types of samples with different mixup ratios λ_{sd} and λ_{td}. Initially, to ensure that the two networks have independent characteristics, we apply a warm-up period of k epochs where each network is independently trained only with the fixed ratio-based mixup and the self-penalization. After enough training, we begin to train with bidirectional matching that can teach each other. Note that one network is trained with pseudo-labels from the peer network which satisfies the confidence threshold constraint. At the same time, we also apply the consistency regularization loss to guarantee stable convergence in training.

Table 1. Comparison of three different mixup-ratio rules on the task A→W.

Type	w/o $\mathcal{L}_{bim}$		w/ $\mathcal{L}_{bim}$	
	SDM	TDM	SDM	TDM
Random	86.5±1.0	85.3±0.9	86.7±0.8	85.6±0.7
Range	86.0±1.7	29.6±6.8	83.3±6.2	81.0±5.4
Fixed (Ours)	86.3±0.6	86.0±0.7	89.3±0.4	90.1±0.3

4. Experiments

We evaluate our proposed method on three domain adaptation benchmarks such as Office-31, Office-Home and VisDA-2017, compared with state-of-the-art domain adaptation methods. In addition, we validate the contributions of the proposed method through extensive ablation studies.

4.1. Setups

Datasets. We evaluated our method in the following three standard benchmarks for UDA.

Office-31 [29] is the most popular dataset for real-world domain adaptation. It contains 4,110 images of 31 categories in three domains: Amazon (A), Webcam (W), DSLR (D). We evaluated all methods on six domain adaptation tasks.

Office-Home [40] is a more challenging benchmark than Office-31. It consists of images of everyday objects organized into four domains: artistic images (Ar), clip art (Cl), product images (Pr), and real-world images (Rw). It contains 15,500 images of 65 classes.

VisDA-2017 [27] is a large-scale dataset for synthetic-to-real domain adaptation. It contains 152,397 synthetic images for the source domain and 55,388 real-world images for the target domain.

Baselines. Since our proposed method can be flexibly applied in any UDA methods, we use DANN [8] as a baseline for a detail analysis of our contributions and MSTN [42] for performance comparisons with the state-of-the-art methods.

Implementation details. Following the standard protocol for UDA, we use all labeled source data and all unlabeled target data. For Office-31 and Office-Home, we use ResNet-50 [13, 14] as the backbone network. We use mini-batch stochastic gradient descent (SGD) with a momentum of 0.9, an initial learning rate of 0.001, and a weight decay of 0.005. We follow the same learning rate schedule as in [8]. The mixup ratios are set up with $\lambda_{sd} = 0.7$ and $\lambda_{td} = 0.3$. The confidence threshold τ is calculated as $(mean - 2 \times std)$ across all mini-batches. We train

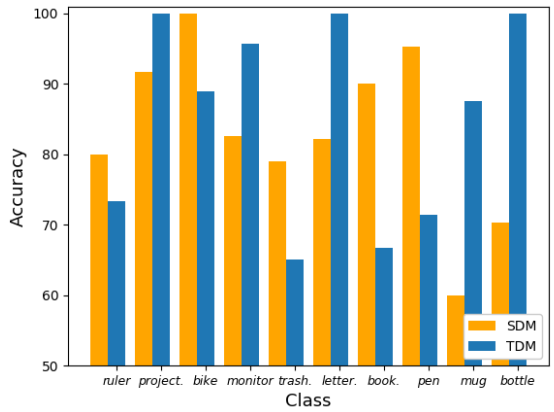
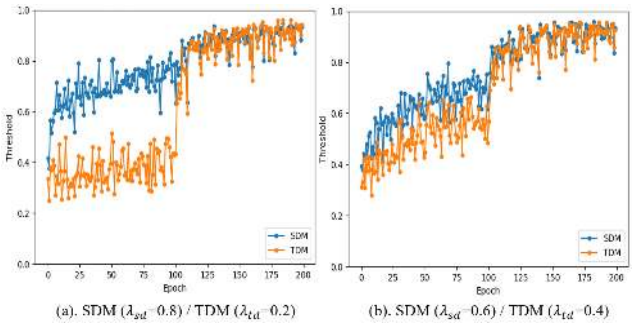
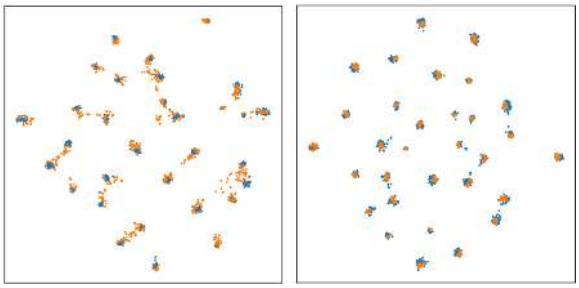


Figure 3. Class-wise accuracy (%) on the task A→W of the Office-31. Best viewed in color.



(a). SDM (λ_{sd} 0.8) / TDM (λ_{td} 0.2) (b). SDM (λ_{sd} 0.6) / TDM (λ_{td} 0.4)

Figure 4. Visualization of the confidence threshold τ on the task A→W of the Office-31. Best viewed in color.



(a) Baseline (DANN) (b) FixBi (Ours)

Figure 5. The visualization of embedded features on the task A→W. Blue and orange points denote the source and target domains, respectively. Best viewed in color.

the model for a total of 200 epochs and set the warm-up epochs to 100. For VisDA-2017, we use ResNet-101 as the backbone architecture. We use the SGD optimizer with a momentum of 0.9, an initial learning rate of 0.0001, and a weight decay of 0.005. We train the model for 25 epochs with warm-up epochs of 10.

4.2. Ablation studies and discussions

For a more detailed analysis of our proposed method, we conducted ablation studies on the Office-31 dataset with DANN [8] as the baseline model.

Comparison of different mixup-ratio rules. We compare our fixed-ratio mixup with two existing general ratios in Table 1. The *"Random"* refers to the ratio randomly sampled from the beta distribution, as in the traditional mixup approaches [46, 44, 26, 43]. The *"Range"* refers to the ratio randomly sampled from the beta distribution and limited to a specific range. In this case, the mixup ratio λ' is determined by $\lambda' \sim \max(\lambda, 1-\lambda)$, where $\lambda \sim \text{Beta}(\alpha, \alpha)$. In this experiment, SDM and TDM are trained with λ' and $1-\lambda'$, respectively, intending to give them different perspectives. We set the hyperparameter α of the beta distribution to 1.0 for *"Random"* and *"Range"*, as used in [44, 26]. Lastly, the *"Fixed"* refers to using two fixed mixup ratios $(\lambda_{sd}, \lambda_{td})$. We set $\lambda_{sd} = 0.7$ and $\lambda_{td} = 0.3$, which satisfies $\lambda_{sd} + \lambda_{td} = 1$. Note that our final accuracy is the result of an ensemble of the output probabilities of both models.

The left side of Table 1 shows the accuracy when only a fixed ratio mixup is applied without the bidirectional matching. In the case of *"Random"*, the accuracy is similar to that of *"Fixed"*. However, in the case of *"Range"*, extreme accuracy degradation is noticeable in TDM. It shows that a mixup which is too target-biased negatively affects learning when the target labels are incorrect. The right side of Table 1 presents the accuracy when applying the bidirectional matching with the fixed ratio mixup. In the case of *"Random"* and *"Range"*, it is difficult to expect performance improvement through the bidirectional matching. By contrast, we observe that the networks trained through the fixed ratio-based mixup can benefit from the bidirectional matching.

Why is it better to use a fixed ratio? We claim that this difference occurs because the two networks have different perspectives through our fixed ratio-based mixup. To verify this, we analyzed the class-wise accuracy of SDM and TDM. We apply $\mathcal{L}_{fm}$ with fixed ratios $\lambda_{sd} = 0.7$ and $\lambda_{td} = 0.3$ to the models, respectively. We pick the top-10 accuracies with the largest difference between class-wise accuracies for the two network models. As shown in Figure 3, we observe that these two models have different strengths and weaknesses from the viewpoint of class-wise accuracy. Note that the performances of these two models are similar at 86.3% and 86.0%.

Comparison with simple ensemble models. To show that our two perspectives have a different meaning to a simple ensemble of a single-perspective, we compare the ensemble of a single-perspective with our proposed method. For the single-perspective, we train models with the mixup ratios of (0.3, 0.3) and (0.7, 0.7), respectively. For a fair comparison, we apply only $\mathcal{L}_{fm}$ and $\mathcal{L}_{bim}$. As shown in Table 2, the accuracy of our two-perspective model is higher than that of the other single-perspective models.

Table 2. Comparison of ensemble networks on Office-31.

Method	$(\lambda_{sd}, \lambda_{td})$	A→W	D→W	W→D	A→D	D→A	W→A	Avg
Single-perspective	(0.3, 0.3)	88.6	96.5	100.0	85.6	69.4	65.1	84.2
	(0.7, 0.7)	89.2	96.5	100.0	85.5	69.1	67.8	84.7
Two-perspective (Ours)	(0.7, 0.3)	**90.1**	**98.5**	**100.0**	**88.4**	**72.5**	**72.5**	**87.0**

Table 3. Ablation results (%) of investigating the effects of our components on Office-31.

$\mathcal{L}_{DANN}$	$\mathcal{L}_{fm}$	$\mathcal{L}_{bim}$	$\mathcal{L}_{sp}$	$\mathcal{L}_{cr}$	A→W	D→W	W→D	A→D	D→A	W→A	Avg
✓					82.0	96.9	99.1	79.7	68.2	67.4	82.2
✓	✓				86.5	98.4	**100.0**	85.5	71.4	71.5	85.5
✓	✓	✓			90.1	98.5	**100.0**	88.4	72.5	72.5	87.0
✓	✓	✓	✓		92.3	98.6	**100.0**	90.4	76.3	74.1	88.6
✓	✓	✓	✓	✓	**94.2**	**99.3**	**100.0**	**91.3**	**76.5**	**74.3**	**89.3**

Table 4. Accuracy (%) on Office-31 for unsupervised domain adaptation (ResNet-50). The best accuracy is indicated in bold and the second best one is underlined. * Reproduced by [5]

Method	A→W	D→W	W→D	A→D	D→A	W→A	Avg
ResNet-50 [13]	68.4±0.2	96.7±0.1	99.3±0.1	68.9±0.2	62.5±0.3	60.7±0.3	76.1
DANN [8]	82.0±0.4	96.9±0.2	99.1±0.1	79.7±0.4	68.2±0.4	67.4±0.5	82.2
MSTN* [42]	91.3	98.9	**100.0**	90.4	72.7	65.6	86.5
CDAN+E [23]	94.1±0.1	98.6±0.1	**100.0±0.0**	92.9±0.2	71.0±0.3	69.3±0.3	87.7
DMRL [41]	90.8±0.3	99.0±0.2	**100.0±0.0**	93.4±0.5	73.0±0.3	71.2±0.3	87.9
SymNets [47]	90.8±0.1	98.8±0.3	**100.0±0.0**	93.9±0.5	74.6±0.6	72.5±0.5	88.4
GSDA [16]	95.7	99.1	**100**	94.8	73.5	74.9	89.7
CAN [17]	94.5±0.3	99.1±0.2	99.8±0.2	95.0±0.3	78.0±0.3	77.0±0.3	90.6
SRDC [36]	95.7±0.2	99.2±0.1	**100.0±0.0**	**95.8±0.2**	76.7±0.3	77.1±0.1	90.8
RSDA-MSTN [11]	**96.1±0.2**	**99.3±0.2**	**100.0±0.0**	95.8±0.3	77.4±0.8	78.9±0.3	91.1
FixBi (Ours)	**96.1±0.2**	**99.3±0.2**	**100.0±0.0**	95.0±0.4	**78.7±0.5**	**79.4±0.3**	**91.4**

Effects of the components of our FixBi. We conduct ablation studies to investigate the effectiveness of the components of our proposed method. In Table 3, our fixed ratio-based mixup improves the baseline DANN [8] on average by 3.3%. The bidirectional matching provides an additional 1.5% improvement on average. Especially, in the task of A→W and A→D, we observe that the bidirectional matching has an impressive impact on performance improvement. We also observe that self-penalization has a significant impact on both task D→A and W→A. In addition, our consistency regularization loss helps to improve performance. Overall, FixBi improves the baseline DANN by an average of 7.1%. This shows that each component is effective in improving performance.

Analysis of the confidence threshold. We visualize our confidence threshold τ in Figure 4. Note that our confidence threshold τ is changed adaptively within each mini-batch, and gradually increased with learning iterations. We observe a sharp increase in the confidence of TDM at 100 epoch, the point at which $\mathcal{L}_{bim}$ starts to be applied. It can be seen more clearly, especially when the difference in the mixup ratio between SDM and TDM is large.

Feature visualization. Figure 5 visualizes the embedded features on the task A→W by t-SNE [39]. For the baseline (e.g, DANN [8]), the target domain features are embedded around the clusters of the source domain features, but it fails to form the clusters of the target domain features. On the other hand, our FixBi constructs the compact clusters of the target domain features close to the source domain features. From this result, we confirm that our proposed method works successfully for an unsupervised domain adaptation task.

4.3. Comparison with the state-of-the-art methods

We compare our method with the various state-of-the-art methods on three public benchmarks. The results of Office-31, Office-Home, and VisDA-2017 are reported in Tables 4, 5, and 6, respectively. We use MSTN [42] as a baseline network. Note that our final accuracy is obtained by the sum of the softmax results of two network models.

Table 5. Accuracy (%) on Office-Home for unsupervised domain adaptation (ResNet-50). The best accuracy is indicated in bold and the second best one is underlined. * Reproduced by [11]

Method	Ar→Cl	Ar→Pr	Ar→Rw	Cl→Ar	Cl→Pr	Cl→Rw	Pr→Ar	Pr→Cl	Pr→Rw	Rw→Ar	Rw→Cl	Rw→Pr	Avg
ResNet-50 [13]	34.9	50	58	37.4	41.9	46.2	38.5	31.2	60.4	53.9	41.2	59.9	46.1
DANN [8]	45.6	59.3	70.1	47	58.5	60.9	46.1	43.7	68.5	63.2	51.8	76.8	57.6
CDAN [23]	49	69.3	74.5	54.4	66	68.4	55.6	48.3	75.9	68.4	55.4	80.5	63.8
MSTN* [42]	49.8	70.3	76.3	60.4	68.5	69.6	61.4	48.9	75.7	70.9	55	81.1	65.7
SymNets [47]	47.7	72.9	78.5	64.2	71.3	74.2	63.6	47.6	79.4	73.8	50.8	82.6	67.2
GSDA [16]	**61.3**	76.1	79.4	65.4	73.3	74.3	65	53.2	80	72.2	60.6	83.1	70.3
GVB-GD [7]	57	74.7	79.8	64.6	74.1	74.6	65.2	55.1	81	74.6	59.7	84.3	70.4
RSDA-MSTN [11]	53.2	**77.7**	**81.3**	66.4	74	76.5	67.9	53	**82**	75.8	57.8	85.4	70.9
SRDC [36]	52.3	76.3	81	**69.5**	76.2	78	**68.7**	53.8	81.7	76.3	57.1	85	71.3
FixBi (Ours)	58.1	77.3	80.4	67.7	**79.5**	**78.1**	65.8	**57.9**	81.7	**76.4**	**62.9**	**86.7**	**72.7**

Table 6. Accuracy (%) on VisDA-2017 for unsupervised domain adaptation (ResNet-101). The best accuracy is indicated in bold and the second best one is underlined. * Reproduced by [5]

Method	aero	bicycle	bus	car	horse	knife	motor	person	plant	skate	train	truck	Avg
ResNet-101 [13]	72.3	6.1	63.4	91.7	52.7	7.9	80.1	5.6	90.1	18.5	78.1	25.9	49.4
DANN [8]	81.9	77.7	82.8	44.3	81.2	29.5	65.1	28.6	51.9	54.6	82.8	7.8	57.4
DAN [22]	68.1	15.4	76.5	87	71.1	48.9	82.3	51.5	88.7	33.2	88.9	42.2	61.1
MSTN* [42]	89.3	49.5	74.3	67.6	90.1	16.6	**93.6**	70.1	86.5	40.4	83.2	18.5	65.0
JAN [24]	75.7	18.7	82.3	86.3	70.2	56.9	80.5	53.8	92.5	32.2	84.5	54.5	65.7
DM-ADA [43]	-	-	-	-	-	-	-	-	-	-	-	-	75.6
DMRL [41]	-	-	-	-	-	-	-	-	-	-	-	-	75.5
MODEL [20]	94.8	73.4	68.8	74.8	93.1	95.4	88.6	84.7	89.1	84.7	83.5	48.1	81.6
STAR [25]	95	84	84.6	73	91.6	91.8	85.9	78.4	94.4	84.7	87	42.2	82.7
CAN [17]	**97**	87.2	82.5	74.3	**97.8**	**96.2**	90.8	80.7	96.6	**96.3**	87.5	**59.9**	**87.2**
FixBi (Ours)	96.1	**87.8**	**90.5**	**90.3**	96.8	95.3	92.8	**88.7**	**97.2**	94.2	**90.9**	25.7	**87.2**

Office-31. Table 4 shows the comparative performance on the Office-31 dataset based on ResNet-50. The average accuracy of our method is 91.4%, which outperforms the other methods such as SRDC [36] and RSDA-MSTN [11]. Our method shows a significant performance improvement over the baseline MSTN [42] method in situations with very large domain shifts, e.g., A→W, W→A, A→D, and D→A tasks. In particular, compared to baseline, the task with the most improved accuracy was W→A, achieving a performance improvement of 13.8%. Compared with the mixup-based method DMRL [41], a large performance improvement is also observed.

Office-Home. In Table 5, we compare our method with recent UDA methods on the Office-Home dataset based on ResNet-50. Our FixBi shows particularly strong performances in tasks with the large domain discrepancy between the real-world domain and the artificial domain, e.g., Rw→Ar, Rw→Cl, Rw→Pr, and Cl→Rw tasks. Our method has an average accuracy of 72.7%, which outperforms the state-of-the-art results achieved by SRDC [36]. Note that our method achieves approximately 2% better accuracy compared with RSDA-MSTN [11] on Office-Home.

VisDA-2017. Table 6 presents the classification accuracy for the VisDA-2017 dataset based on ResNet-101. Our FixBi achieves 22.2% performance improvement on an average compared with the baseline method [42]. Above all, our method shows about 12% better accuracy than other mixup-based methods [41, 43] and achieves comparable performance compared to the state-of-the-art methods.

5. Conclusion

In this paper, we proposed a FixBi algorithm that bridging the domain spaces to deal with the large domain discrepancy problem in an unsupervised domain adaptation scenario. Our main methodology is to construct an intermediate domain with different characteristics between the source domain and the target domain. We completed this through our fixed ratio-based mixup with different mixup-ratios, and further proposed bidirectional matching, self-penalization, and consistency regularization for efficient use of intermediate space. Extensive ablation studies demonstrate the effectiveness of our proposed algorithm and experiments on the three standard benchmarks show that our proposed method achieves competitive performance to the state-of-the-art methods.

Acknowledgement. This work was partially supported by NRF-2020R1F1A1066049 and Technology Innovation Program (TIP-20000316) funded by the Ministry of Trade, Industry & Energy (MOTIE, Korea).

References

[1] S. Ao, X. Li, and C. X. Ling. Fast generalized distillation for semi-supervised domain adaptation. In *Thirty-First AAAI Conference on Artificial Intelligence (AAAI)*, 2017. 2

[2] D. Berthelot, N. Carlini, E. D. Cubuk, A. Kurakin, H. Zhang, and C. Raffel. Remixmatch: Semi-supervised learning with distribution alignment and augmentation anchoring. In *International Conference on Learning Representations (ICLR)*, Apr. 2020. 2

[3] D. Berthelot, N. Carlini, I. Goodfellow, N. Papernot, A. Oliver, and C. Raffel. Mixmatch: A holistic approach to semi-supervised learning. In *Thirty-third Conference on Neural Information Processing Systems (NeurIPS)*, Dec. 2019. 1, 2, 3

[4] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Eleventh Annual Conference on Computational Learning Theory (COLT)*, 1998. 4

[5] W.-G. Chang, T. You, S. Seo, S. Kwak, and B. Han. Domain-specific batch normalization for unsupervised domain adaptation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 7, 8

[6] J. Chen, V. Shah, and A. Kyrillidis. Negative sampling in semi-supervised learning. In *International Conference on Machine Learning (ICML)*, 2020. 2

[7] S. Cui, S. Wang, J. Zhou, C. Su, Q. Huang, and Q. Tian. Gradually vanishing bridge for adversarial domain adaptation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2, 8

[8] Y. Ganin and V. Lempitsky. Unsupervised domain adaptation by back propagation. In *International Conference on Machine Learning (ICML)*, 2015. 1, 2, 3, 4, 5, 6, 7, 8

[9] S. Gidaris, P. Singh, and N. Komodakis. Unsupervised representation learning by predicting image rotations. In *arXiv preprint arXiv:1803.07728*, 2018. 1

[10] R. Gopalan, R. Li, and R. Chellappa. Domain adaptation for object recognition: An unsupervised approach. In *The IEEE International Conference on Computer Vision (ICCV)*, pages 999–1006, 2011. 2

[11] X. Gu, J. Sun, and Z. Xu. Spherical space domain adaptation with robust pseudo-label loss. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9101–9110, 2020. 3, 5, 7, 8

[12] B. Han, Q. Yao, X. Yu, G. Niu, M. Xu, W. Hu, I. Tsang, and M. Sugiyama. Robust training of deep neural networks with extremely noisy labels. In *Thirty-fourth Conference on Neural Information Processing Systems (NeurIPS)*, 2020. 2, 4

[13] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 5, 7, 8

[14] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. In *The European Conference on Computer Vision (ECCV)*, 2016. 5

[15] J. Hoffman, E. Tzeng, T. Park, J.-Y. Zhu, P. Isola, K. Saenko, A. A. Efros, and T. Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *arXiv preprint arXiv:1711.03213*, 2017. 2

[16] L. Hu, M. Kan, S. Shan, and X. Chen. Unsupervised domain adaptation with hierarchical gradient synchronization. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 7, 8

[17] G. Kang, L. Jiang, Y. Yang, and A. G. Hauptmann. Contrastive adaptation network for unsupervised domain adaptation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 3, 7, 8

[18] D. Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *ICML Workshop on Challenges in Representation Learning*, 2013. 2

[19] D. Li and T. Hospedales. Online meta-learning for multi-source and semi-supervised domain adaptation. In *arXiv preprint arXiv:2004.04398*, 2020. 2

[20] R. Li, Q. Jiao, W. Cao, H.-S. Wong, and S. Wu. Model adaptation: Unsupervised domain adaptation without source data. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 8

[21] M. Long, Y. Cao, J. Wang, and M. I. Jordan. Learning transferable features with deep adaptation network. In *arXiv preprint arXiv:1502.0279*1,*, 2015. 1, 2

[22] M. Long, Y. Cao, J. Wang, and M. I. Jordan. Learning transferable features with deep adaptation networks. In *International Conference on Machine Learning (ICML)*, 2015. 8

[23] M. Long, Z. CAO, J. Wang, and M. I. Jordan. Conditional adversarial domain adaptation. In *Thirty-second Conference on Neural Information Processing Systems (NeurIPS)*, 2018. 7, 8

[24] M. Long, H. Zhu, J. Wang, and M. I. Jordan. Deep transfer learning with joint adaptation network. In *Thirty-fourth International Conference on Machine Learning (ICML)*, 2017. 1, 2, 8

[25] Z. Lu, Y. Yang, X. Zhu, C. Liu, Y.-Z. Song, and T. Xiang. Stochastic classifiers for unsupervised domain adaptation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 8

[26] X. Mao, Y. Ma, Z. Yang, Y. Chen, and Q. Li. Virtual mixup training for unsupervised domain adaptation. In *arXiv preprint arXiv:1905.04215*, 2019. 3, 6

[27] X. Peng, B. Usman, N. Kaushik, J. Hoffman, D. Wang, and K. Saenko. Visda: The visual domain adaptation challenge. In *arXiv preprint arXiv:1710.06924*, 2017. 2, 5

[28] C. Qin, L. Wang, Q. Ma, Y. Yin, H. Wang, and Y. Fu. Opposite structure learning for semi-supervised domain adaptation. In *arXiv preprint arXiv:2002.02545*, 2020. 2

[29] K. Saenko, B. Kulis, M. Fritz, and T. Darrell. Adapting visual category models to new domains. In *The European Conference on Computer Vision (ECCV)*, 2010. 2, 5

[30] K. Saito, D. Kim, S. Sclaroff, T. Darrell, and K. Saenko. Semi-supervised domain adaptation via minimax entropy. In *The IEEE International Conference on Computer Vision (ICCV)*, 2019. 2

[31] K. Saito, Y. Ushiku, T. Harada, and K. Saenko. Adversarial dropout regularization. In *International Conference on Learning Representations (ICLR)*, 2018. 2

[32] K. Saito, K. Watanabe, Y. Ushiku, and T. Harada. Maximum classifier discrepancy for unsupervised domain adaptation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3723–3732, 2018. 3

[33] M. Sajjadi, M. Javanmardi, and T. Tasdizen. Regularization with stochastic transformations and pertuations for dep semi-supervised learning. In *Thirtieth Conference on Neural Information Processing Systems (NeurIPS)*, 2016. 2

[34] K. Shon, D. Berthelot, C. Li, Z. Zhang, N. Carlini, E. D. Cubuk, A. Kurakin, H. Zhang, and C. Raffel. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In *arXiv preprint arXiv:2001.07685*, 2020. 1, 2, 3, 4

[35] B. Sun and K. Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *The European Conference on Computer Vision (ECCV)*, 2016. 2

[36] H. Tang, K. Chen, and K. Jia. Unsupervised domain adaptation via structurally regularized deep clustering. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 3, 7, 8

[37] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell. Adversarial discriminative domain adaptation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7167–7176, 2017. 1, 2, 3

[38] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell. Adversarial discriminative domain adaptation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7167–7176, 2017. 2, 3

[39] L. van der Maaten and G. Hinton. Visualizing data using t-sne. In *Journal of Machine Learning Research (JMLR)*, 2008. 7

[40] H. Venkateswara, J. Eusebio, S. Chakraborty, , and S. Panchanathan. Deep hashing network for unsupervised domain adaptation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2, 5

[41] Y. Wu, D. Inkpen, and A. El-Roby. Dual mixup regularized learning for adversarial domain adaptation. In *The European Conference on Computer Vision (ECCV)*, 2020. 2, 3, 7, 8

[42] S. Xie, Z. Zheng, L. Chen, and C. Chen. Learning semantic representations for unsupervised domain adaptation. In *Thirty-fifth International Conference on Machine Learning (ICML)*, 2018. 4, 5, 7, 8

[43] M. Xu, J. Zhang, B. Ni, T. Li, C. Wang, Q. Tian, and W. Zhang. Adversarial domain adaptation with domain mixup. In *Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI)*, 2020. 2, 3, 6, 8

[44] L. Yang, Y. Wang, M. Gao, A. Shrivastava, K. Q. Weinberger, W.-L. Chao, and S.-N. Lim. Mico: Mixup co-training for semi-supervised domain adaptation. In *arXiv preprint arXiv:2007.12684*, 2020. 2, 3, 4, 6

[45] X. Zhai, A. Oliver, A. Kolesnikov, and L. Beyer. S4l: Self-supervised semi-supervised learning. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct. 2019. 1, 2

[46] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz. Mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations (ICLR)*, Apr. 2018. 3, 6

[47] Y. Zhang, H. Tang, K. Jia, and M. Tan. Domain-symmetric networks for adversarial domain adaptation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 7, 8