PATCH-MIX TRANSFORMER FOR UNSUPERVISED DOMAIN ADAPTATION: A GAME PERSPECTIVE

Anonymous authors
Paper under double-blind review

ABSTRACT

Endeavors have been recently made to leverage the vision transformer (ViT) for the challenging unsupervised domain adaptation (UDA) task. They typically adopt the cross-attention in ViT for direct domain alignment. However, as the performance of cross-attention highly relies on the quality of pseudo labels for the targeted samples, it becomes less effective when the domain gap becomes large. We solve this problem from a game theory's perspective with the model called **PMTrans**, which bridges the source and the target domains with an intermediate domain. Specifically, we propose a novel ViT-based module called **PatchMix** that effectively builds up the intermediate domain, *i.e.*, probability distribution, by learning to sample patches from both domains based on the game-theoretical models. In this way, it learns to mix the patches from source and target domains to maximize the cross entropy (CE), while exploiting two semi-supervised mixup losses in the feature and label spaces to minimize it. As such, we interpret the process of UDA as a min-max CE game with three players, including the feature extractor, classifier, and PatchMix, to find the optimal Nash Equilibria solution. Moreover, we leverage attention maps from ViT to re-weight the label of each patch by its importance, making it possible to obtain more domain-discriminative feature representations. We conduct extensive experiments on four benchmark datasets, and the results show that PMTrans significantly surpasses the ViT-based and CNN-based SoTA methods by **+1.4%** on Office-31, **+3.5%** on Office-Home, and **+17.7%** on DomainNet, respectively.

1 INTRODUCTION

Convolutional neural networks (CNNs) have achieved tremendous success on numerous vision tasks; however, they still suffer from the limited generalization capability to a new domain due to the domain shift problem Zhang et al. (2022). Unsupervised domain adaptation (UDA) tackles this issue by transferring knowledge from a labeled source domain to an unlabeled target domain Pan & Yang (2010). A significant line of solutions reduces the domain gap based on the category-level alignment which produces pseudo labels for the target samples, such as metric learning Kang et al. (2019); Zhu et al. (2021), adversarial training Saito et al. (2018); Du et al. (2021); Li et al. (2021a), and optimal transport Xu et al. (2020b). Furthermore, several works Dosovitskiy et al. (2021); Sun et al. (2022) explore the potential of ViT for the non-trivial UDA task. Recently, CDTrans Xu et al. (2021) exploits the cross-attention in ViT for direct domain alignment, buttressed by the crafted pseudo labels for target samples. However, CDTrans has a distinct limitation: as the performance of cross-attention highly depends on the quality of pseudo labels, it becomes less effective when the domain gap becomes large. As shown in Fig. 1(a), due that the domain gap between the domain *qdr* and others is significant, aligning it directly with others performs poorly.

In this paper, we probe a new problem for UDA: *how to smoothly bridge the source and target domains by constructing an intermediate domain with an effective ViT-based solution?* The intuition behind this is that, compared to direct aligning domains, alleviating the domain gap between the intermediate and source/target domain can facilitate the domain alignment. Accordingly, we propose a novel and effective method, called **PMTrans** (PatchMix Transformer) to construct the intermediate representations. Overall, PMTrans interprets the process of domain alignment as a min-max cross entropy (CE) game with three players, *i.e.*, the feature extractor, a classifier, and a **PatchMix** module, to find the optimal Nash Equilibria. Importantly, the PatchMix module is proposed to effectively build up the intermediate domain, *i.e.*, probability distribution, by learning to sample patches from both

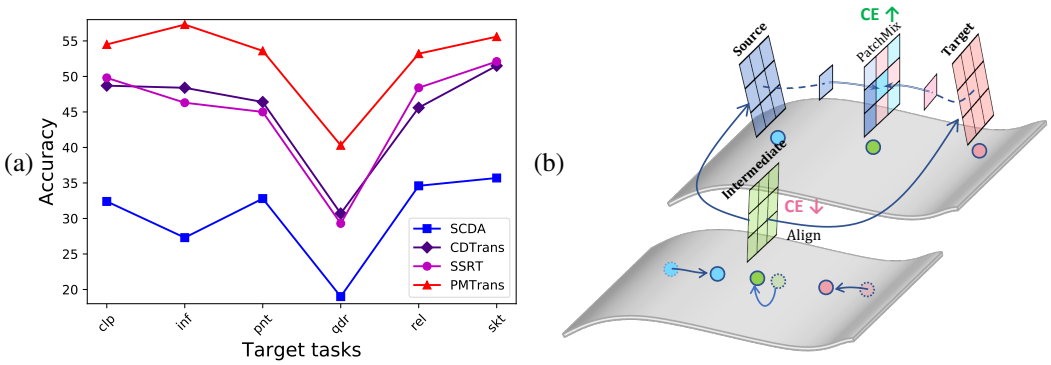

Figure 1: (a) The classification accuracy of our PMTrans surpasses the SoTA methods by **+17.7%** on the most challenging DomainNet dataset. Note that the target tasks take one domain of DomainNet as the target domain and other domains as source domains. (b) PMTrans builds up the intermediate domain (green patches) via a novel PatchMix module by learning to sample patches from the source (blue patches) and target (red patches) domains. PatchMix tries to maximize the CE (↑) between the intermediate domain and source/target domain, while other players try to minimize it (↓) by aligning their features.

domains with weights generated from a learnable Beta distribution based on the game-theoretical models Acuna et al. (2022b); Başar & Olsder (1982); Mazumdar et al. (2020), as shown in Fig. 1(b). That is, we aim to learn to mix patches from the two domains to maximize the CE between the intermediate domain and source/target domain. Moreover, two semi-supervised mixup losses in the feature and label spaces are proposed to minimize the CE. Interestingly, *we conclude that source and target domains are aligned if mixing the patch representations from two domains is equivalent to mixing the corresponding labels*. Therefore, the domain discrepancy can be measured based on the CE between the mixed patches and mixed labels. Eventually, the three players have no incentive to change their parameters to disturb CE, meaning the source and target domains are well aligned. Unlike existing mixup methods Zhang et al. (2018); Yun et al. (2019); Uddin et al. (2021), our proposed PatchMix subtly learns to combine the element-wise global and local mixture by mixing patches from the source and target domains for ViT-based UDA. Moreover, we leverage the class activation mapping (CAM) from ViT to allocate the semantic information to re-weight the label of each patch, thus enabling us to obtain more domain-discriminative features.

We conduct experiments on **four** benchmark datasets, including Office-31 Saenko et al. (2010), Office-Home Venkateswara et al. (2017), VisDA-2017 Peng et al. (2017), and DomainNet Peng et al. (2019). The results show that the performance of PMTrans significantly surpasses that of the ViT-based Sun et al. (2022); Xu et al. (2021); Yang et al. (2021) and CNN-based SoTA methods Na et al. (2021); Li et al. (2021c); Sharma et al. (2021) by **+1.4%** on Office-31, **+3.5%** on Office-Home, and **+17.7%** on DomainNet (See Fig. 1(a)), respectively.

In summary, the main contributions of our paper are four-fold: (**I**) We propose a novel ViT-based UDA framework, PMTrans, to effectively bridge the source and target domains by constructing the intermediate domain. (**II**) We propose PatchMix, a novel module to build up the intermediate domain via the game-theoretical models. (**III**) We propose two semi-supervised mixup losses in the feature and label spaces to reduce CE in the game. (**IV**) Our PMTrans suppresses the prior methods by a large margin on three benchmark datasets.

2 Related Work

Unsupervised Domain Adaptation The prevailing UDA methods focus on domain alignment and learning discriminative domain-invariant features via metric learning, domain adversarial training, and optimal transport. Firstly, the metric learning-based methods aim to reduce the domain discrepancy by learning the domain-invariant feature representations using various metrics. For instance, some methods Long et al. (2015; 2017); Zhu et al. (2019); Kang et al. (2019) use the maximum mean discrepancy (MMD) loss to measure the divergence between different domains. In addition, the central moment discrepancy (CMD) loss Zellinger et al. (2017) and maximum density divergence

(MDD) loss Li et al. (2021b) are also proposed to align the feature distributions. Secondly, the domain adversarial training methods learn the domain-invariant representations to encourage samples from different domains to be non-discriminative with respect to the domain labels via an adversarial loss Ganin et al. (2016); Xu et al. (2020a); Wu et al. (2020). The third type of approaches aim to minimize the cost transported from the source to the target distribution by finding an optimal coupling cost to mitigate the domain shift Courty et al. (2017b;a). Unfortunately, these methods are not robust enough to the noisy pseudo target labels for accurate domain alignment. Different from these mainstream UDA methods and Acuna et al. (2022a), we interpret the process of UDA as a min-max CE game and find the optimal Nash Equilibria for domain alignment with an intermediate domain and a pure ViT-based solution.

Mixup. It is an effective data augmentation technique to prevent models from over-fitting to the training data by linearly interpolating between two input data. Mixup types can be categorized into the global mixup (e.g., Mixup Zhang et al. (2018) and Manifold-Mixup Zhang et al. (2018)) and local mixup (CutMix Yun et al. (2019), saliency-CutMix Uddin et al. (2021), and TransMix Chen et al. (2021)). In CNN-based UDA tasks, several works Xu et al. (2020a); Wu et al. (2020); Na et al. (2021) also use the mixup technique by linearly mixing the source and target domain data. In comparison, we unify the global and local mixup in our PMTrans framework by learning to form a mixed patch from the source/target patch as the input to ViT. We learn the hyperparameters of the mixup ratio for each patch, which is the first attempt to interpolate patches based on the distribution estimation. Accordingly, we propose PatchMix that effectively builds up the intermediate domain by sampling patches from both domains based on the game-theoretical models.

Transformer. Transformer Vaswani et al. (2017) has recently been introduced to tackle the challenges in various vision tasks Caron et al. (2021); Liu et al. (2021b). Consequently, several works have leveraged the vision transformer (ViT) for the non-trivial UDA task. TVT Yang et al. (2021) proposes an adaptation module to capture the transferable and discriminative features of domain data. SSRT Sun et al. (2022) proposes a framework with a transformer backbone and a safe self-refinement strategy to handle the issues in case of a large domain gap. More recently, CDTrans Xu et al. (2021) proposes a two-step framework that utilizes the cross-attention in ViT for direct feature alignment, along with pre-generated pseudo labels for the target samples. Differently, we probe to construct an intermediate domain to bridge the source and target domains for better domain alignment. Our PMTrans effectively interprets the process of domain alignment as a min-max CE game, leading to a significant UDA performance enhancement (See Sec. 3).

3 METHODOLOGY

In UDA, given a labeled source set $\mathcal{D}_s = \{(\boldsymbol{x}_i^s, \boldsymbol{y}_i^s)\}_{i=1}^{n_s}$ with i-th sample $\boldsymbol{x}_i^s$ and its corresponding one-hot label $\boldsymbol{y}_i^s$ and an unlabeled target set $\mathcal{D}_t = \left\{\boldsymbol{x}_j^t\right\}_{j=1}^{n_t}$ with j-th sample $\boldsymbol{x}_j^t$, we use n_s and n_t to denote the size of samples in the source and target domains, respectively. Note that the data in two domains are sampled from two different distributions, and we assume that the two domains share the same label space. Our goal is to address the significant domain gap issue and transfer the knowledge from the source domain to the target domain well. In this section, we interpret the process of UDA from a game perspective, then describe the proposed PMTrans which smoothly aligns the source and target domain by constructing an intermediate domain.

3.1 PMTRANS: THEORETICAL ANALYSIS

3.1.1 PATCHMIX

Definition 1 *(PatchMix): Let $\mathcal{P}_\lambda$ be a linear interpolation operation on two pairs of randomly drawn samples $(\boldsymbol{x}^s, \boldsymbol{y}^s)$ and $(\boldsymbol{x}^t, \boldsymbol{y}^t)$. Then with $\lambda_k \sim \mathrm{Beta}(\beta, \gamma)$, it interpolates k-th source patch $\boldsymbol{x}_k^s$ and target patch $\boldsymbol{x}_k^t$ to reconstruct a mixed representation with n patches.*

$$\begin{aligned} \boldsymbol{x}^i &= \mathcal{P}_\lambda(\boldsymbol{x}^s, \boldsymbol{x}^t) = \sum_{k=1}^{n} (\lambda_k \boldsymbol{x}_k^s + (1-\lambda_k)\boldsymbol{x}_k^t), \\ \boldsymbol{y}^i &= \mathcal{P}_\lambda(\boldsymbol{y}^s, \boldsymbol{y}^t) = \frac{(\sum_{k=1}^{n} \lambda_k)\boldsymbol{y}^s + (\sum_{k=1}^{n}(1-\lambda_k))\boldsymbol{y}^t}{n}. \end{aligned} \tag{1}$$

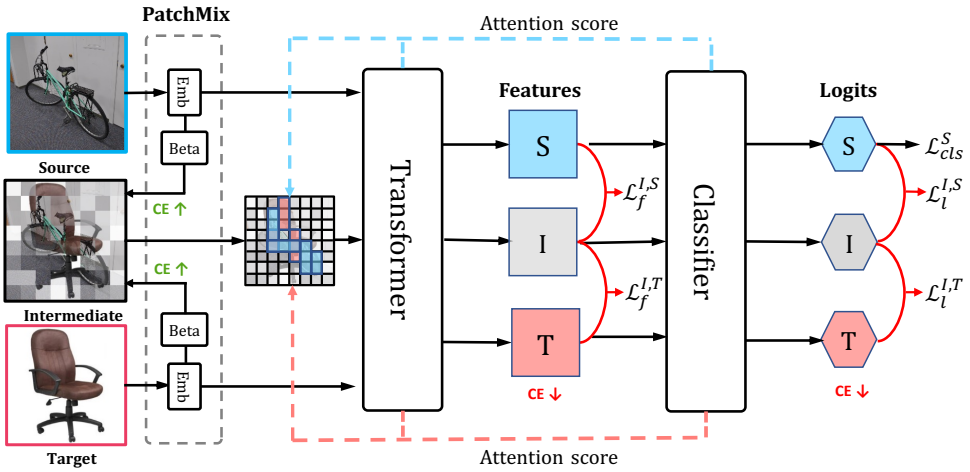

Figure 2: Overview of the proposed PMTrans framework. It consists of three players: the PatchMix module empowered by a patch embedding (**Emb**) layer and a learnable Beta distribution (**Beta**), ViT encoder, and classifier. The ViT encoder aims to extract features from the patch sequences obtained by **Emb**. The classifier maps the outputs of ViT encoder to make predictions, each of which is exploited to select the feature map to re-weight the patch sequences. We adopt CE to measure the effect of each player's strategy in the game.

In Definition 1, we assume the intermediate domain follows the Beta distribution, *i.e.*each image $\boldsymbol{x}^i$ composes the sampled patches $\boldsymbol{x}_k$ from source/target domain. Here, $\lambda_k \in [0, 1]$ is the random mixing proportion that denotes the patch-level sampling weights. Furthermore, we calculate the image-level importance by aggregating patch weights $\sum_{k=1}^{n}(1-\lambda_k)$, which is used to interpolate their labels. As a result, we mix both samples $(\boldsymbol{x}^s, \boldsymbol{y}^s)$ and $(\boldsymbol{x}^t, \boldsymbol{y}^t)$ to construct a new intermediate domain $\mathcal{D}_i = \left\{\left(\boldsymbol{x}_l^i, \boldsymbol{y}_l^i\right)\right\}_{l=1}^{n_i}$, which shares information from both the source domain $\mathcal{D}_s$ and the target domain $\mathcal{D}_t$.

To align the source and the target domains, we need to evaluate the gap numerically. In detail, let P_S and P_T be the empirical distributions defined by D_s and $\mathcal{D}_t$, respectively. $D(P_S, P_T)$ measures the divergence between the source and target domains, and can be defined as

$$D\left(P_S, P_T\right) = \inf_{\boldsymbol{h}_1^s,\ldots,\boldsymbol{h}_{n_s}^s \in \mathcal{H}^s, \boldsymbol{h}_1^t,\ldots,\boldsymbol{h}_{n_t}^t \in \mathcal{H}^t} \frac{1}{n_s \times n_t} \sum_i^{n_s} \sum_j^{n_t} \left\{ \inf_{c\in\mathcal{C}} \int_0^1 \ell\left(f\left(\mathcal{P}_\lambda\left(\boldsymbol{h}_i^s, \boldsymbol{h}_j^t\right)\right), \mathcal{P}_\lambda\left(\boldsymbol{y}_i^s, \boldsymbol{y}_j^t\right)\right) p(\lambda) \mathrm{d}\lambda \right\}, \quad (2)$$

where ℓ is the CE loss, $h_i^s = f(x_i^s)$ and $h_j^t = f(x_j^t)$. Note $\mathcal{H}^s$ and H^t denote the representation spaces with dimensionality $\dim(\mathcal{H})$ for the source and target domains, respectively. $\mathcal{F}$ denotes the set of encoding functions *i.e.*, the encoder and $\mathcal{C}$ denotes the set of decoding functions *i.e.*the classifier. Let $\mathcal{P}_\lambda$ be the set of functions to generate the mixup ratio for building the intermediate domain. Let $f^\star \in \mathcal{F}, c^\star \in \mathcal{C},$ and $\lambda^\star \in \mathcal{P}_\lambda$ be the minimizers of Eq.4 (*ref to the suppl. material.*)

Theorem 1 *(Domain Distribution Estimation with PatchMix): Let $d \in \mathbb{N}$ to represent the number of classes contained in three sets $\mathcal{D}_s$, $\mathcal{D}_t$, and $\mathcal{D}_i$. If $\dim(\mathcal{H}) \geq d-1$, $\lambda^\star \ell(c^\star(f^\star(\boldsymbol{x}_i)), \boldsymbol{y}^s) + (1-\lambda^\star)\ell(c^\star(f^\star(\boldsymbol{x}_i)), \boldsymbol{y}^t) = 0$, then $D\left(P_S, P_T\right) = 0$ and the corresponding minimizer $c^\star$ is a linear function from $\mathcal{H}$ to $\mathbb{R}^d$.*

Theorem 1 indicates that *the source and target domains will be aligned if mixing the patches from two domains is equivalent to mixing the corresponding labels*. Therefore, minimizing the CE between the mixed patches and mixed labels can effectively facilitate domain alignment. *For the proof of Theorem 1, refer to the suppl. material.*

3.1.2 A MIN-MAX CE GAME

We interpret UDA as a min-max CE game among three players, namely the feature extractor ($\mathcal{F}$), classifier ($\mathcal{C}$), and PatchMix module ($\mathcal{P}$), as shown in Fig. 2. To specify each player's role, we define $\boldsymbol{\omega}_1 \in \Omega_1$, $\boldsymbol{\omega}_2 \in \Omega_2$, and $\boldsymbol{\omega}_3 \in \Omega_3$ as the parameters of $\mathcal{F}$, $\mathcal{C}$, and $\mathcal{P}$, respectively. The joint domain is defined as $\Omega = \Omega_1 \times \Omega_2 \times \Omega_3$ and their joint parameter set is defined as $\boldsymbol{\omega} = \{\boldsymbol{\omega}_1, \boldsymbol{\omega}_2, \boldsymbol{\omega}_3\}$. Then

we use the subscript $_{-m}$ to denote all other parameters/players except m, *e.g.*, $\boldsymbol{\omega}_{-2} = \{\boldsymbol{\omega}_1, \boldsymbol{\omega}_3\}$. In our game, m-th player is endowed with a cost function J_m and strives to reduce its cost, which contributes to the change of CE. Each player' cost function J_m is represented as

$$\begin{aligned} J_1\left(\boldsymbol{\omega}_1, \boldsymbol{\omega}_{-1}\right) &:= \mathcal{L}_{cls}(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2) + \alpha CE_{s,i,t}(\boldsymbol{\omega}), \\ J_2\left(\boldsymbol{\omega}_2, \boldsymbol{\omega}_{-2}\right) &:= \mathcal{L}_{cls}(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2) + \alpha\lambda CE_{s,i,t}(\boldsymbol{\omega}), \\ J_3\left(\boldsymbol{\omega}_3, \boldsymbol{\omega}_{-3}\right) &:= -\alpha CE_{s,i,t}(\boldsymbol{\omega}), \end{aligned} \tag{3}$$

where α is the trade-off parameter, ℓ is the supervised classification loss for the source domain, and $CE_{s,i,t}(\boldsymbol{\omega})$ is the discrepancy between the intermediate domain and the source/target domain. The definitions of $\mathcal{L}_{cls}(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2)$ and $CE_{s,i,t}(\boldsymbol{\omega})$ are shown in Sec. 3.2. As illustrated in Eq. 3, the game is essentially a min-max process, *i.e.*, a competition for the player $\mathcal{P}$ against both players $\mathcal{F}$ and $\mathcal{C}$. Specifically, as depicted in Fig. 2, $\mathcal{P}$ strives to diverge while $\mathcal{F}$ and $\mathcal{C}$ try to align domain distributions, which is a min-max process on CE. In this min-max CE game, each player behaves selfishly to reduce its cost function, and this competition will possibly end with a situation where no one has anything to gain by changing only one's strategy. This situation is called Nash Equilibrium (NE) in game theory.

Definition 2 *(Nash Equilibrium): The equilibrium states each player's strategy is the best response to other players.*

$$\exists \boldsymbol{\omega}^* \in \Omega, \forall m \in \{1,2,3\}, s.t. J_m(\boldsymbol{\omega}_m^*, \boldsymbol{\omega}_{-m}^*) \leq J_m(\boldsymbol{\omega}_m, \boldsymbol{\omega}_{-m}^*).$$

Intuitively, in our case, NE means that no player has the incentive to change its own parameters, as there is no additional pay-off.

3.2 The Proposed Framework

Overview. Fig. 2 illustrates the framework of our proposed PMTrans, which consists of a ViT encoder, a classifier, and a PatchMix module. Firstly, the patch embedding (Emb) layer in PatchMix transforms input images from source/target domains into patches. Then, based on Definition. 1, PatchMix randomly samples patches from source and target domains to construct the intermediate domain, as shown in Fig. 1(b). Lastly, patches are refined with a ViT encoder, and the classifier uses the refined representations to make predictions. Next, we describe the technical details of PMTrans.

PatchMix. When exploiting PatchMix to construct the intermediate domain, it is worth noting that not all patches have equal contributions for the label assignment. As Chen *et al.*Chen et al. (2021) observed, the mixed image has no valid objects due to the random process while there is still a response in the label space. To address this issue, we re-weight $\mathcal{P}_\lambda(\boldsymbol{y}^s, \boldsymbol{y}^t)$ in Definition.1 with the normalized attention score a_k. For the implementation details of attention scores, refer to *the suppl. material*. The re-scaled $\mathcal{P}_\lambda(\boldsymbol{y}^s, \boldsymbol{y}^t)$ is defined as $\mathcal{P}_\lambda(\boldsymbol{y}^s, \boldsymbol{y}^t) = \lambda^s \boldsymbol{y}^s + \lambda^t \boldsymbol{y}^t$, where

$$\lambda^s = \frac{\sum_{k=1}^n \lambda_k a_k^s}{\sum_{k=1}^n \lambda_k a_k^s + \sum_{k=1}^n (1-\lambda_k) a_k^t}, \ \lambda^t = \frac{\sum_{k=1}^n (1-\lambda_k) a_k^t}{\sum_{k=1}^n \lambda_k a_k^s + \sum_{k=1}^n (1-\lambda_k) a_k^t}.$$

Semi-supervised mixup loss. As PatchMix tries to maximize the CE between the intermediate domain and source/target domain, we now need to find a way to minimize the CE in the game. In detail, two semi-supervised mixup losses are proposed in the feature and label spaces to align the domains.

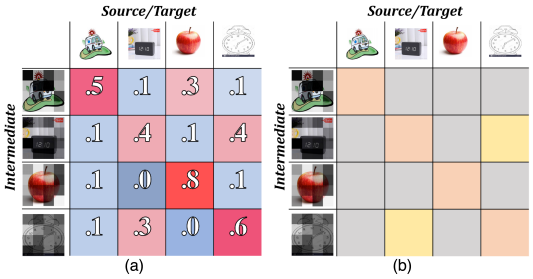

Figure 3: The illustration of the semi-supervised loss in the feature space.

Firstly, we compute the normalized cosine similarity between the intermediate domain (column) and source/target domain (row) in the feature space, as shown in Fig. 3(a). Each normalized score denotes the similarity between a sample of the intermediate domain and its source/target counterpart. For its supervision, intuitively, for the source domain, we exploit ground-truth information by the label similarity $y^{is} = y^s(y^s)^\intercal$, as a binary matrix to represent whether samples share the same labels. As shown in Fig.3 (b), the yellow and pink items indicate true, while others indicate false. Moreover, for the intermediate and

target domains, due to lack of supervision, we only use pink parts *i.e.*the identity matrix y^{it} as the label similarity. Then, we utilize the CE to measure the domain discrepancy based on the difference between the feature similarity and label similarity.

The supervised mixup loss in the feature space is formulated as:

$$\mathcal{L}_f^{I,S}(\boldsymbol{\omega}_1, \boldsymbol{\omega}_3) = \mathbb{E}_{(\boldsymbol{x}^i, \boldsymbol{y}^i)\sim D^i} \lambda^s \ell \left(d\left(\mathcal{F}\left(\boldsymbol{x}^i\right), \mathcal{F}\left(\boldsymbol{x}^s\right)\right), \boldsymbol{y}^{is}\right),$$

where $d(\cdot,\cdot)$ denotes the normalized cosine similarity, as shown in Fig. 3(a), between features across domains and ℓ denotes the CE loss. Similarly, to measure the divergence between the intermediate and target domains in the feature space, we propose an unsupervised mixup loss, which is defined as:

$$\mathcal{L}_f^{I,T}(\boldsymbol{\omega}_1, \boldsymbol{\omega}_3) = \mathbb{E}_{(\boldsymbol{x}^i, \boldsymbol{y}^i)\sim D^i} \lambda^t \ell \left(d\left(\mathcal{F}\left(\boldsymbol{x}^i\right), \mathcal{F}\left(\boldsymbol{x}^t\right)\right), \boldsymbol{y}^{it}\right),$$

Moreover, as introduced in Theorem 1, we apply a supervised mixup loss in the label space to measure the domain divergence based on the CE loss.

$$\mathcal{L}_l^{I,S}(\boldsymbol{\omega}) = \mathbb{E}_{(\boldsymbol{\omega}^i, \boldsymbol{y}^i)\sim D^i} \lambda^s \ell \left(\mathcal{C}\left(\mathcal{F}\left(\boldsymbol{x}^i\right)\right), \boldsymbol{y}^s\right), \ \mathcal{L}_l^{I,T}(\boldsymbol{\omega}) = \mathbb{E}_{(\boldsymbol{x}^i, \boldsymbol{y}^i)\sim D^i} \lambda^t \ell \left(\mathcal{C}\left(\mathcal{F}\left(\boldsymbol{x}^i\right)\right), \hat{\boldsymbol{y}^t}\right),$$

where $\hat{\boldsymbol{y}^t}$ is pseudo label for target data. For convenience, we utilize the method, commonly used in Liang et al. (2020; 2021), to generate the pseudo label $\hat{\boldsymbol{y}^t}$ for each sample via k-means cluster. Finally, the two semi-supervised mixup losses in the feature and label spaces are formulated as:

$$\mathcal{L}_f(\boldsymbol{\omega}_1, \boldsymbol{\omega}_3) = \mathcal{L}_f^{I,S}(\boldsymbol{\omega}_1, \boldsymbol{\omega}_3) + \mathcal{L}_f^{I,T}(\boldsymbol{\omega}_1, \boldsymbol{\omega}_3); \mathcal{L}_l(\boldsymbol{\omega}) = \mathcal{L}_l^{I,S}(\boldsymbol{\omega}) + \mathcal{L}_l^{I,T}(\boldsymbol{\omega}).$$

We also apply the classification loss to the labeled source domain data, formulated as:

$$\mathcal{L}_{cls}(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2) = \mathbb{E}_{(\boldsymbol{x}^s, \boldsymbol{y}^s)\sim D^s} \ell \left(\mathcal{C}\left(\mathcal{F}\left(\boldsymbol{x}^s\right)\right), \boldsymbol{y}^s\right).$$

A Three-Player Game. Finally, the min-max CE game aims to align distributions in the feature and label spaces. The total CE between the intermediate domain and source/target domain is:

$$CE_{s,i,t}(\boldsymbol{\omega}) = \mathcal{L}_f(\boldsymbol{\omega}_1, \boldsymbol{\omega}_3) + \mathcal{L}_l(\boldsymbol{\omega}).$$

Note that, instead of using gradient reverse layers Ganin & Lempitsky (2015) for the domain classification to increase the domain gap, we adopt the *random* mixup-ratio from Beta distribution in our PatchMix module to maximize the CE between the intermediate domain and source/target domain. Moreover, the feature extractor and classifier have the same objective to minimize the CE between the intermediate domain and source/target domain. Therefore, the total objective of PMTrans is achieved by reformulating Eq.3 as

$$J\left(\boldsymbol{\omega}\right) := \mathcal{L}_{cls}(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2) + \alpha CE_{s,i,t}(\boldsymbol{\omega}),$$

where α is trade-off parameter. As such, we can obtain the solution of the game with only one-step optimization, which is more efficient than that proposed in CDTrans Xu et al. (2021). After optimizing the objective, the PatchMix module with the ideal Beta distribution will not maximize the CE anymore. Meanwhile, the feature extractor and classifier have no incentive to change their parameters to minimize the CE. Finally, the discrepancy between the intermediate domain and source/target domain is nearly zero, indicating that the source and target domains are well aligned.

4 Experiments

4.1 Datasets, Implementations, and Compared Methods

Datasets. To evaluate the proposed method, we conduct extensive experiments on four popular UDA benchmarks, including Office-31 Saenko et al. (2010), Office-Home Venkateswara et al. (2017), VisDA-2017 Peng et al. (2017), and DomainNet Peng et al. (2019). *Due to page limit, the details of the datasets and the construction of transfer tasks on these datasets are put in the suppl. material.*
Implementations. In all experiments, we use the Swin-Base transformer Liu et al. (2021a) pre-trained on ImageNet Deng et al. (2009) as the backbone for our PMTrans. The base learning rate is $5e^{-6}$ with a batch size of 32, and we train models by 50 epochs. For VisDA-2017, we use lower learning rate $1e^{-6}$. We adopt AdamW Loshchilov & Hutter (2019) with a momentum of 0.9, and a weight decay of 0.05 as the optimizer for all our experiments. Furthermore, for fine-tuning purposes, we set

Table 1: Comparison with SoTA methods on Office-31. The best performance is marked as **bold**.

Method	A → W	D→ W	W→ D	A→ D	D→ A	W→ A	Avg
ResNet-50	68.9	68.4	62.5	96.7	60.7	99.3	76.1
BNM	91.5	98.5	**100.0**	90.3	70.9	71.6	87.1
DWL	89.2	99.2	**100.0**	91.2	73.1	69.8	87.1
MDD	94.5	98.4	**100.0**	93.5	74.6	72.2	88.9
TSA	94.8	99.1	**100.0**	92.6	74.9	74.4	89.3
ILA+CDAN	95.7	99.2	**100.0**	93.4	72.1	75.4	89.3
PCT	94.6	98.7	99.9	93.8	77.2	76.0	90.0
SCDA	94.2	98.7	99.8	95.2	75.7	76.2	90.0
FixBi	96.1	99.3	**100.0**	95.0	78.7	79.4	91.4
TVT	96.4	99.4	**100.0**	96.4	84.9	86.0	93.9
Deit-Base	89.2	98.9	**100.0**	88.7	80.1	79.8	89.5
CDTrans-Deit	96.7	99.0	**100.0**	97.0	81.1	81.9	92.6
PMTrans-Deit	99.0	99.4	**100.0**	96.5	81.4	82.1	93.1
ViT-Base	91.2	99.2	**100.0**	90.4	81.1	80.6	91.1
SSRT-ViT	97.7	99.2	**100.0**	98.6	83.5	82.2	93.5
PMTrans-ViT	99.1	**99.6**	**100.0**	99.4	85.7	86.3	95.0
Swin-Base	97.0	99.2	**100.0**	95.8	82.4	81.8	92.7
PMTrans-Swin	**99.5**	99.4	**100.0**	**99.8**	**86.7**	**86.5**	**95.3**

the classifier with a higher learning rate $1e^{-5}$ for our main tasks and learn the trade-off parameter adaptively. The classifier is implemented as an MLP. For a fair comparison with prior works, we conduct experiments with the same backbone Deit-Base Touvron et al. (2020) as CDTrans Xu et al. (2021), and ViT-base Dosovitskiy et al. (2021) as SSRT Sun et al. (2022) on Office-31, Office-Home, and VisDA-2017. Both studies are trained for 60 epochs with a learning rate of $1e^{-5}$.

Baseline Methods. We compare PMTrans with the SoTA methods, including ResNet- and ViT-based methods. The ResNet-based methods are FixBi Na et al. (2021), CGDM Du et al. (2021), MCD Saito et al. (2018), SWD Lee et al. (2019), SCDA Li et al. (2021d), BNM Cui et al. (2020), MDD Zhang et al. (2019b), CKB Luo & Ren (2021), TSA Li et al. (2021c), DWL Xiao & Zhang (2021), ILA Sharma et al. (2021), Symnets Zhang et al. (2019a), CAN Kang et al. (2019), and PCT Tanwisuth et al. (2021). The ViT-based methods are SSRT Sun et al. (2022), CDTrans Xu et al. (2021), and TVT Yang et al. (2021), and we directly quote the results in their original papers for fair comparison.

4.2 RESULTS

For the ResNet-based methods, we utilize ResNet-50 as the backbone for the Office-31, Office-Home, and DomainNet datasets, and we adopt ResNet-101 for VisDA-2017 dataset. Note that each backbone is trained with the source data only and then tested with the target data.

Results on Office-31. Table 1 shows the quantitative comparison with the CNN-based and ViT-based methods. Overall, our PMTrans achieves the best performance on each task and outperforms the SoTA methods with the same backbones. Numerically, PMTrans noticeably surpass the SoTA methods with an increase of **+2.9%** accuracy over CDTrans, **+1.4%** accuracy over TVT, and **+1.8%** accuracy over SSRT, respectively.

Results on Office-Home. Table 2 shows the quantitative results using different backbones. As expected, our PMTrans framework achieves noticeable performance gains and surpasses TVT, SSRT, and CDTrans by a large margin. Importantly, our PMTrans achieves an improvement more than **4.9%** accuracy over the Swin backbone Liu et al. (2021a). Interestingly, our proposed PMTrans can decrease domain divergence effectively even without the Swin backbone. The results indicate that our method can obtain more robust transferable representations than the CNN-based and ViT-based methods.

Results on VisDA-2017. As shown in Table 3, our PMTrans achieves **88.0%** accuracy and outperforms the baseline by **11.2%**. In particular, for the 'hard' categories, such as "person", our method consistently achieves a much higher performance boost from **29.0%** to **70.3%**. These improvements indicate that our method shows an excellent generalization capability and achieves comparable performance (**88.0%**) with the SoTA methods (**88.7%**). PMTrans also surpasses the SoTA methods on several sub-categories, such as "horse" and "sktbrd". In particular, it is shown that the SoTA methods, *e.g.*, CDTrans and SSRT, achieve better results on this dataset. The reason

Table 2: Comparison with SoTA methods on Office-Home. The best performance is marked as **bold**.

Method	A→ C	A→ P	A → R	C → A	C → P	C → R	P→ A	P→ C	P→ R	R→ A	R→ C	R→ P	Avg
ResNet-50	44.9	66.3	74.3	51.8	61.9	63.6	52.4	39.1	71.2	63.8	45.9	77.2	59.4
MCD	48.9	68.3	74.6	61.3	67.6	68.8	57.0	47.1	75.1	69.1	52.2	79.6	64.1
Symnets	47.7	72.9	78.5	64.2	71.3	74.2	64.2	48.8	79.5	74.5	52.6	82.7	67.6
MDD	54.9	73.7	77.8	60.0	71.4	71.8	61.2	53.6	78.1	72.5	60.2	82.3	68.1
TSA	53.6	75.1	78.3	64.4	73.7	72.5	62.3	49.4	77.5	72.2	58.8	82.1	68.3
CKB	54.7	74.4	77.1	63.7	72.2	71.8	64.1	51.7	78.4	73.1	58.0	82.4	68.5
BNM	56.7	77.5	81.0	67.3	76.3	77.1	65.3	55.1	82.0	73.6	57.0	84.3	71.1
PCT	57.1	78.3	81.4	67.6	77.0	76.5	68.0	55.0	81.3	74.7	60.0	85.3	71.8
FixBi	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
TVT	74.9	86.8	89.5	82.8	88.0	88.3	79.8	71.9	90.1	85.5	74.6	90.6	83.6
Deit-Base	61.8	79.5	84.3	75.4	78.8	81.2	72.8	55.7	84.4	78.3	59.3	86.0	74.8
CDTrans-Deit	68.8	85.0	86.9	81.5	87.1	87.3	79.6	63.3	88.2	82.0	66.0	90.6	80.5
PMTrans-Deit	71.8	87.3	88.3	83.0	87.7	87.8	78.5	67.4	89.3	81.7	70.7	92.0	82.1
ViT-Base	67.0	85.7	88.1	80.1	84.1	86.7	79.5	67.0	89.4	83.6	70.2	91.2	81.1
SSRT-ViT	75.2	89.0	91.1	85.1	88.3	89.9	85.0	74.2	91.2	85.7	78.6	91.8	85.4
PMTrans-ViT	**81.2**	91.6	92.4	**88.9**	91.6	**93.0**	**88.5**	**80.0**	**93.4**	**89.5**	**82.4**	94.5	**88.9**
Swin-Base	72.7	87.1	90.6	84.3	87.3	89.3	80.6	68.6	90.3	84.8	69.4	91.3	83.6
PMTrans-Swin	79.7	**92.3**	**92.6**	88.3	**93.1**	92.8	87.3	**80.0**	92.8	88.8	79.8	**94.6**	88.5

Table 3: Comparison with SoTA methods on VisDA-2017. The best performance is marked as **bold**.

Method	plane	bcycl	bus	car	horse	knife	mcycl	person	plant	sktbrd	train	truck	Avg
ResNet-50	55.1	53.3	61.9	59.1	80.6	17.9	79.7	31.2	81.0	26.5	73.5	8.5	52.4
BNM	89.6	61.5	76.9	55.0	89.3	69.1	81.3	65.5	90.0	47.3	89.1	30.1	70.4
MCD	87.0	60.9	83.7	64.0	88.9	79.6	84.7	76.9	88.6	40.3	83.0	25.8	71.9
SWD	90.8	82.5	81.7	70.5	91.7	69.5	86.3	77.5	87.4	63.6	85.6	29.2	76.4
DWL	90.7	80.2	86.1	67.6	92.4	81.5	86.8	78.0	90.6	57.1	85.6	28.7	77.1
CGDM	93.4	82.7	73.2	68.4	92.9	94.5	88.7	82.1	93.4	82.5	86.8	49.2	82.3
CAN	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	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
TVT	82.9	85.6	77.5	60.5	93.6	98.2	89.4	76.4	93.6	92.0	91.7	55.7	83.1
Deit-Base	98.2	73.0	82.5	62.0	97.3	63.5	96.5	29.8	68.7	86.7	96.7	23.6	73.2
CDTrans-Deit	97.1	90.5	82.4	77.5	96.6	96.1	93.6	**88.6**	**97.9**	86.9	90.3	**62.8**	88.4
PMTrans-Deit	98.2	92.2	88.1	77.0	97.4	95.8	94.0	72.1	97.1	95.2	94.6	51.0	87.7
ViT-Base	99.1	60.7	70.1	82.7	96.5	73.1	97.1	19.7	64.5	94.7	97.2	15.4	72.6
SSRT-ViT	98.9	87.6	**89.1**	**84.8**	98.3	**98.7**	96.3	81.1	94.8	97.9	94.5	43.1	**88.8**
PMTrans-ViT	98.9	**93.7**	84.5	73.3	**99.0**	98.0	96.2	67.8	94.2	**98.4**	96.6	49.0	87.5
Swin-Base	99.3	63.4	85.9	68.9	95.1	79.6	**97.1**	29.0	81.4	94.2	**97.7**	29.6	76.8
PMTrans-Swin	**99.4**	88.3	88.1	78.9	98.8	98.3	95.8	70.3	94.6	98.3	96.3	48.5	88.0

is that CDTrans and SSRT are trained with a batch size of 64 while PMTrans's batch size is 32. It indicates that when the input size is much bigger, the input can represent the data distributions better. *A detailed ablation study for this issue can be found in the suppl. material.*.

Results on DomainNet. PMTrans achieves a very high average accuracy on the most challenging DomainNet dataset, as shown in Table 4. Overall, our proposed PMTrans outperforms the SoTA methods by **+17.7%** accuracy. Incredibly, PMTrans surpasses the SoTA methods in all the 30 sub-tasks, which demonstrates the strong ability of PMTrans to alleviate the large domain gap. Moreover, transferring knowledge is much more difficult when the domain gap becomes significant. *When taking more challenging qdr as target domain while others as the source domain, our PMTrans achieves an average accuracy of **27.0%**, while SSRT and CDTrans only achieve an average accuracy of **13.7%** and **19.6%**, respectively.* The comparisons on DomainNet dataset demonstrate that our PMTrans yields the best generalization ability for the challenging UDA problem.

4.3 Ablation Study

Semi-supervised mixup loss. As shown in Table 5, Swin with the semi-supervised mixup loss in the feature and label spaces outperforms the counterpart built on Swin with only source training by **+0.3%** and **+4.3%** on Office-Home dataset, respectively. The results indicate the effectiveness of the semi-supervised mixup loss for minimizing the domain discrepancy. Moreover, we observe that the CE loss yields better performance on the label space than that on the feature space. The reason is that the CE loss on the label space utilizes the class information better than on the feature space. *Due to the page limit, more experiments and analyses can be found in the suppl. material.*

Table 4: Comparison with SoTA methods on DomainNet. The best performance is marked as **bold**.

MCD	clp	inf	pnt	qdr	rel	skt	Avg	SWD	clp	inf	pnt	qdr	rel	skt	Avg	BNM	clp	inf	pnt	qdr	rel	skt	Avg
clp	-	15.4	25.5	3.3	44.6	31.2	24.0	clp	-	14.7	31.9	10.1	45.3	36.5	27.7	clp	-	12.1	33.1	6.2	50.8	40.2	28.5
inf	24.1	-	24.0	1.6	35.2	19.7	20.9	inf	22.9	-	24.2	2.5	33.2	21.3	20.0	inf	26.6	-	28.5	2.4	38.5	18.1	22.8
pnt	31.1	14.8	-	1.7	48.1	22.8	23.7	pnt	33.6	15.3	-	4.4	46.1	30.7	26.0	pnt	39.9	12.2	-	3.4	54.5	36.2	29.2
qdr	8.5	2.1	4.6	-	7.9	7.1	6.0	qdr	15.5	2.2	6.4	-	11.1	10.2	9.1	qdr	17.8	1.0	3.6	-	9.2	8.3	8.0
rel	39.4	17.8	41.2	1.5	-	25.2	25.0	real	41.2	18.1	44.2	4.6	-	31.6	27.9	rel	48.6	13.2	49.7	3.6	-	33.9	29.8
skt	37.3	12.6	27.2	4.1	34.5	-	23.1	skt	44.2	15.2	37.3	10.3	44.7	-	30.3	skt	54.9	12.8	42.3	5.4	51.3	-	33.3
Avg	28.1	12.5	24.5	2.4	34.1	21.2	20.5	Avg	31.5	13.1	28.8	6.4	36.1	26.1	23.6	Avg	37.6	10.3	31.4	4.2	40.9	27.3	25.3
CGDM	clp	inf	pnt	qdr	rel	skt	Avg	MDD	clp	inf	pnt	qdr	rel	skt	Avg	SCDA	clp	inf	pnt	qdr	rel	skt	Avg
clp	-	16.9	35.3	10.8	53.5	36.9	30.7	clp	-	20.5	40.7	6.2	52.5	42.1	32.4	clp	-	18.6	39.3	5.1	55.0	44.1	32.4
inf	27.8	-	28.2	4.4	48.2	22.5	26.2	inf	33.0	-	33.8	2.6	46.2	24.5	28.0	inf	29.6	-	34.0	1.4	46.3	25.4	27.3
pnt	37.7	14.5	-	4.6	59.4	33.5	30.0	pnt	43.7	20.4	-	2.8	51.2	41.7	32.0	pnt	44.1	19.0	-	2.6	56.2	42.0	32.8
qdr	14.9	1.5	6.2	-	10.9	10.2	8.7	qdr	18.4	3.0	8.1	-	12.9	11.8	10.8	qdr	30.0	4.9	15.0	-	25.4	19.8	19.0
rel	49.4	20.8	47.2	4.8	-	38.2	32.0	rel	52.8	21.6	47.8	4.2	-	41.2	33.5	rel	54.0	22.5	51.9	2.3	-	42.5	34.6
skt	50.1	16.5	43.7	11.1	55.6	-	35.4	skt	54.3	17.5	43.1	5.7	54.2	-	35.0	skt	55.6	18.5	44.7	6.4	53.2	-	35.7
Avg	36.0	14.0	32.1	7.1	45.5	28.3	27.2	Avg	40.4	16.6	34.7	4.3	43.4	32.3	28.6	Avg	42.6	16.7	37.0	3.6	47.2	34.8	30.3
CDTrans	clp	inf	pnt	qdr	rel	skt	Avg	SSRT	clp	inf	pnt	qdr	rel	skt	Avg	**PMTrans**	clp	inf	pnt	qdr	rel	skt	Avg
clp	-	29.4	57.2	26.0	72.6	58.1	48.7	clp	-	33.8	60.2	19.4	75.8	59.8	49.8	clp	-	34.2	62.7	32.5	79.3	63.7	54.5
inf	57.0	-	54.4	12.8	69.5	48.4	48.4	inf	55.5	-	54.0	9.0	68.2	44.7	46.3	inf	67.4	-	61.1	22.2	78.0	57.6	57.3
pnt	62.9	27.4	-	15.8	72.1	53.9	46.4	pnt	61.7	28.5	-	8.4	71.4	55.2	45.0	pnt	69.7	33.5	-	23.9	79.8	61.2	53.6
qdr	44.6	8.9	29.0	-	42.6	28.5	30.7	qdr	42.5	8.8	24.2	-	37.6	33.6	29.3	qdr	54.6	17.4	38.9	-	49.5	41.0	40.3
rel	66.2	31.0	61.5	16.2	-	52.9	45.6	rel	69.9	37.1	66.0	10.1	-	58.9	48.4	rel	74.1	35.3	70.0	25.4	-	61.1	53.2
skt	69.0	29.6	59.0	27.2	72.5	-	51.5	skt	70.6	32.8	62.2	21.7	73.2	-	52.1	skt	73.8	33.0	62.6	30.9	77.5	-	55.6
Avg	59.9	25.3	52.2	19.6	65.9	48.4	45.2	Avg	60.0	28.2	53.3	13.7	65.3	50.4	45.2	Avg	67.9	30.7	59.1	27.0	72.8	56.9	**62.9**

Table 5: Effect of semi-supervised loss. The best performance is marked as **bold**.

$\mathcal{L}_{cls}$	$\mathcal{L}_f$	$\mathcal{L}_l$	A→C	A→P	A→R	C→A	C→P	C→R	P→A	P→C	P→R	R→A	R→C	R→P	Avg
✓			72.7	87.1	90.6	84.3	87.3	89.3	80.6	68.6	90.3	84.8	69.4	91.3	83.6
✓	✓		73.3	87.2	90.8	84.8	87.5	89.5	81.5	71.1	90.5	85.2	72.9	92.0	83.9
✓		✓	79.2	91.8	92.3	88.0	92.6	**93.0**	87.1	77.8	92.5	88.2	78.4	93.9	87.9
✓	✓	✓	**79.7**	**92.3**	**92.6**	**88.3**	**93.1**	92.8	**87.3**	**80.0**	**92.8**	**88.8**	**79.8**	**94.6**	**88.5**

Table 6: Effect of learning parameters. The best performance is marked as **bold**.

Method	A→C	A→P	A→R	C→A	C→P	C→R	P→A	P→C	P→R	R→A	R→C	R→P	Avg
Beta(1,1)	**79.9**	92.0	92.3	**88.6**	92.6	92.4	86.9	79.0	92.4	88.2	79.3	94.0	88.1
Beta(2,2)	79.9	92.1	92.7	88.4	92.4	92.7	86.9	79.5	92.1	88.1	79.6	94.3	88.2
Learning	79.7	**92.3**	**92.6**	88.3	**93.1**	**92.8**	**87.3**	**80.0**	**92.8**	**88.8**	**79.8**	**94.6**	**88.5**

Table 7: Effect of PatchMix. The best performance is marked as **bold**.

Method	A→C	A→P	A→R	C→A	C→P	C→R	P→A	P→C	P→R	R→A	R→C	R→P	Avg
Mixup	79.4	**92.4**	**92.6**	87.5	92.8	92.4	86.8	**80.3**	92.5	88.2	79.7	**95.4**	88.3
CutMix	79.2	91.2	92.2	87.6	91.8	91.8	86.0	77.8	92.6	88.2	78.4	94.1	87.6
PatchMix	**79.7**	92.3	**92.6**	**88.3**	**93.1**	**92.8**	**87.3**	80.0	**92.8**	**88.8**	**79.8**	**94.6**	**88.5**

Learning hyperparameters of mixup. Table 6 shows the ablation results for the effects of learning the hyperparameters of the Beta distribution using the Office-Home dataset. We compared the learning hyperparameters of mixup with fixed parameters, such as Beta(1,1) and Beta(2,2). The proposed method achieves **+0.4%** and **+0.3%** accuracy increment compared with that based on Beta(1,1) and Beta(2,2), respectively. The results demonstrate that learning to estimate the distribution to build up the intermediate domain can benefit domain alignment.

PatchMix. Comparisons of PMTrans with Mixup Zhang et al. (2018) and CutMix Yun et al. (2019) are shown in Table 7. PMTrans outperforms Mixup and CutMix by **+0.2%** and **+0.9%** accuracy on the Office-Home dataset, demonstrating that PatchMix can capture the global and local mixture information better than the global mixture Mixup and local mixture CutMix methods.

5 Conclusion and Future Work

In this paper, we proposed a novel method, PMTrans, an optimization solution for UDA from a game perspective. Specifically, we first proposed a novel ViT-based module called PatchMix that effectively built up the intermediate domain to learn discriminative domain-invariant representations for domains. And the two semi-supervised mixup losses were proposed to assist in finding the optimal Nash Equilibria. Moreover, we leveraged attention maps from ViT to re-weight the label of each patch by its significance. PMTrans achieved the SoTA results on four benchmark UDA datasets, outperforming the SoTA methods by a large margin. In the near future, we plan to implement our PatchMix and the two semi-supervised mixup losses to solve self-supervised and semi-supervised learning problems. We will also exploit our method to tackle the challenging downstream tasks, *e.g.*, semantic segmentation and object detection.

REFERENCES

David Acuna, Marc T. Law, Guojun Zhang, and Sanja Fidler. Domain adversarial training: A game perspective. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022a.

David Acuna, Marc T. Law, Guojun Zhang, and Sanja Fidler. Domain adversarial training: A game perspective. *CoRR*, abs/2202.05352, 2022b.

Tamer Başar and Geert Jan Olsder. Dynamic noncooperative game theory. 1982.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pp. 9630–9640. IEEE, 2021.

Jieneng Chen, Shuyang Sun, Ju He, Philip H. S. Torr, Alan L. Yuille, and Song Bai. Transmix: Attend to mix for vision transformers. *CoRR*, abs/2111.09833, 2021.

Nicolas Courty, Rémi Flamary, Amaury Habrard, and Alain Rakotomamonjy. Joint distribution optimal transportation for domain adaptation. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 3730–3739, 2017a.

Nicolas Courty, Rémi Flamary, Devis Tuia, and Alain Rakotomamonjy. Optimal transport for domain adaptation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(9):1853–1865, 2017b.

Shuhao Cui, Shuhui Wang, Junbao Zhuo, Liang Li, Qingming Huang, and Qi Tian. Towards discriminability and diversity: Batch nuclear-norm maximization under label insufficient situations. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pp. 3940–3949. Computer Vision Foundation / IEEE, 2020.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pp. 248–255. IEEE Computer Society, 2009.

Terrance Devries and Graham W. Taylor. Improved regularization of convolutional neural networks with cutout. *CoRR*, abs/1708.04552, 2017.

Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *Proceedings of the 31th International Conference on Machine Learning, ICML*, volume 32 of *JMLR Workshop and Conference Proceedings*, pp. 647–655. JMLR.org, 2014.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *ArXiv*, abs/2010.11929, 2021.

Zhekai Du, Jingjing Li, Hongzu Su, Lei Zhu, and Ke Lu. Cross-domain gradient discrepancy minimization for unsupervised domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pp. 3937–3946. Computer Vision Foundation / IEEE, 2021.

Yaroslav Ganin and Victor S. Lempitsky. Unsupervised domain adaptation by backpropagation. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pp. 1180–1189. JMLR.org, 2015.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor S. Lempitsky. Domain-adversarial training of neural networks. *J. Mach. Learn. Res.*, 17:59:1–59:35, 2016.

Wei Gao, Fang Wan, Xingjia Pan, Zhiliang Peng, Qi Tian, Zhenjun Han, Bolei Zhou, and Qixiang Ye. TS-CAM: token semantic coupled attention map for weakly supervised object localization. *CoRR*, abs/2103.14862, 2021. URL https://arxiv.org/abs/2103.14862.

Guoliang Kang, Lu Jiang, Yi Yang, and Alexander G. Hauptmann. Contrastive adaptation network for unsupervised domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pp. 4893–4902. Computer Vision Foundation / IEEE, 2019.

Chen-Yu Lee, Tanmay Batra, Mohammad Haris Baig, and Daniel Ulbricht. Sliced wasserstein discrepancy for unsupervised domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pp. 10285–10295. Computer Vision Foundation / IEEE, 2019.

Jichang Li, Guanbin Li, Yemin Shi, and Yizhou Yu. Cross-domain adaptive clustering for semi-supervised domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pp. 2505–2514. Computer Vision Foundation / IEEE, 2021a.

Jingjing Li, Erpeng Chen, Zhengming Ding, Lei Zhu, Ke Lu, and Heng Tao Shen. Maximum density divergence for domain adaptation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 43(11):3918–3930, 2021b.

Shuang Li, Mixue Xie, Kaixiong Gong, Chi Harold Liu, Yulin Wang, and Wei Li. Transferable semantic augmentation for domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pp. 11516–11525. Computer Vision Foundation / IEEE, 2021c.

Shuang Li, Mixue Xie, Fangrui Lv, Chi Harold Liu, Jian Liang, Chen Qin, and Wei Li. Semantic concentration for domain adaptation. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pp. 9082–9091. IEEE, 2021d.

Jian Liang, Dapeng Hu, and Jiashi Feng. Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pp. 6028–6039. PMLR, 2020.

Jian Liang, Dapeng Hu, and Jiashi Feng. Domain adaptation with auxiliary target domain-oriented classifier. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pp. 16632–16642. Computer Vision Foundation / IEEE, 2021.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9992–10002, 2021a.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pp. 9992–10002. IEEE, 2021b.

Mingsheng Long, Yue Cao, Jianmin Wang, and Michael I. Jordan. Learning transferable features with deep adaptation networks. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pp. 97–105. JMLR.org, 2015.

Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I. Jordan. Deep transfer learning with joint adaptation networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pp. 2208–2217. PMLR, 2017.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

You-Wei Luo and Chuan-Xian Ren. Conditional bures metric for domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pp. 13989–13998. Computer Vision Foundation / IEEE, 2021.

Eric Mazumdar, Lillian J. Ratliff, and S. Shankar Sastry. On gradient-based learning in continuous games. *SIAM J. Math. Data Sci.*, 2(1):103–131, 2020.

Jaemin Na, Heechul Jung, Hyung Jin Chang, and Wonjun Hwang. Fixbi: Bridging domain spaces for unsupervised domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pp. 1094–1103. Computer Vision Foundation / IEEE, 2021.

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

Xingchao Peng, Ben Usman, Neela Kaushik, Judy Hoffman, Dequan Wang, and Kate Saenko. Visda: The visual domain adaptation challenge. *CoRR*, abs/1710.06924, 2017.

Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pp. 1406–1415. IEEE, 2019.

Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *Computer Vision - ECCV 2010, 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part IV*, volume 6314 of *Lecture Notes in Computer Science*, pp. 213–226. Springer, 2010.

Kuniaki Saito, Kohei Watanabe, Yoshitaka Ushiku, and Tatsuya Harada. Maximum classifier discrepancy for unsupervised domain adaptation. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pp. 3723–3732. Computer Vision Foundation / IEEE Computer Society, 2018.

Astuti Sharma, Tarun Kalluri, and Manmohan Chandraker. Instance level affinity-based transfer for unsupervised domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pp. 5361–5371. Computer Vision Foundation / IEEE, 2021.

Tao Sun, Cheng Lu, Tianshuo Zhang, and Haibin Ling. Safe self-refinement for transformer-based domain adaptation. *CoRR*, abs/2204.07683, 2022.

Korawat Tanwisuth, Xinjie Fan, Huangjie Zheng, Shujian Zhang, Hao Zhang, Bo Chen, and Mingyuan Zhou. A prototype-oriented framework for unsupervised domain adaptation. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pp. 17194–17208, 2021.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. *CoRR*, abs/2012.12877, 2020. URL `https://arxiv.org/abs/2012.12877`.

A. F. M. Shahab Uddin, Mst. Sirazam Monira, Wheemyung Shin, TaeChoong Chung, and Sung-Ho Bae. Saliencymix: A saliency guided data augmentation strategy for better regularization. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 5998–6008, 2017.

Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pp. 5385–5394. IEEE Computer Society, 2017.

Yuan Wu, Diana Inkpen, and Ahmed El-Roby. Dual mixup regularized learning for adversarial domain adaptation. In *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XXIX*, volume 12374 of *Lecture Notes in Computer Science*, pp. 540–555. Springer, 2020.

Ni Xiao and Lei Zhang. Dynamic weighted learning for unsupervised domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pp. 15242–15251. Computer Vision Foundation / IEEE, 2021.

Minghao Xu, Jian Zhang, Bingbing Ni, Teng Li, Chengjie Wang, Qi Tian, and Wenjun Zhang. Adversarial domain adaptation with domain mixup. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pp. 6502–6509. AAAI Press, 2020a.

Renjun Xu, Pelen Liu, Liyan Wang, Chao Chen, and Jindong Wang. Reliable weighted optimal transport for unsupervised domain adaptation. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pp. 4393–4402. Computer Vision Foundation / IEEE, 2020b.

Tongkun Xu, Weihua Chen, Pichao Wang, Fan Wang, Hao Li, and Rong Jin. Cdtrans: Cross-domain transformer for unsupervised domain adaptation. *CoRR*, abs/2109.06165, 2021.

Jinyu Yang, Jingjing Liu, Ning Xu, and Junzhou Huang. TVT: transferable vision transformer for unsupervised domain adaptation. *CoRR*, abs/2108.05988, 2021.

Sangdoo Yun, Dongyoon Han, Sanghyuk Chun, Seong Joon Oh, Youngjoon Yoo, and Junsuk Choe. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pp. 6022–6031. IEEE, 2019.

Werner Zellinger, Thomas Grubinger, Edwin Lughofer, Thomas Natschläger, and Susanne Saminger-Platz. Central moment discrepancy (CMD) for domain-invariant representation learning. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

Hongyi Zhang, Moustapha Cissé, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

Yabin Zhang, Hui Tang, Kui Jia, and Mingkui Tan. Domain-symmetric networks for adversarial domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pp. 5031–5040. Computer Vision Foundation / IEEE, 2019a.

Yabin Zhang, Bin Deng, Hui Tang, Lei Zhang, and Kui Jia. Unsupervised multi-class domain adaptation: Theory, algorithms, and practice. *IEEE Trans. Pattern Anal. Mach. Intell.*, 44(5): 2775–2792, 2022.

Yuchen Zhang, Tianle Liu, Mingsheng Long, and Michael I. Jordan. Bridging theory and algorithm for domain adaptation. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pp. 7404–7413. PMLR, 2019b.

Bolei Zhou, Aditya Khosla, Àgata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. *CoRR*, abs/1512.04150, 2015. URL `http://arxiv.org/abs/1512.04150`.

Yongchun Zhu, Fuzhen Zhuang, and Deqing Wang. Aligning domain-specific distribution and classifier for cross-domain classification from multiple sources. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pp. 5989–5996. AAAI Press, 2019.

Yongchun Zhu, Fuzhen Zhuang, Jindong Wang, Guolin Ke, Jingwu Chen, Jiang Bian, Hui Xiong, and Qing He. Deep subdomain adaptation network for image classification. *IEEE Trans. Neural Networks Learn. Syst.*, 32(4):1713–1722, 2021.

A APPENDIX

In this supplementary material, we first prove theorem 2 and its inference in Section B. Then, Section D shows the algorithm of the proposed PMTrans, and Section C introduces the details of the proposed method. Finally, Section E and Section F show the results, analyses, and ablation experiments to prove the effectiveness of the proposed PMTrans.

B PROOFS

B.1 DOMAIN DISTRIBUTION ESTIMATION WITH PATCHMIX

Let $\mathcal{H}$ denote the representation spaces with dimensionality $\dim(\mathcal{H})$, $\mathcal{F}$ denote the set of encoding functions *i.e.*, the encoder and $\mathcal{C}$ be the set of decoding functions *i.e.*the classifier. Let $\mathcal{P}_\lambda$ be the set of functions to generate mixup ratio for building the intermediate domain. Furthermore, let P_S, P_T, and P_I be the empirical distributions of data $\mathcal{D}_s$, $\mathcal{D}_t$, and $\mathcal{D}_i$. Define $f^\star \in \mathcal{F}, c^\star \in \mathcal{C},$ and $\lambda^\star \in \mathcal{P}$ be the minimizers of Eq. 4 and $D(P_S, P_T)$ as the measure of the domain divergence between P_S and P_T:

$$D(P_S, P_T) = \inf_{f\in\mathcal{F}, c\in\mathcal{C}, \lambda\in\mathcal{P}_\lambda} \mathop{\mathbb{E}}_{(x^s,y^s),(x^t,y^t)} \ell\left(c\left(\mathcal{P}_\lambda\left(f(x^s), f\left(x^t\right)\right)\right), \mathcal{P}_\lambda\left(y^s, y^t\right)\right), \tag{4}$$

where ℓ is the CE loss. Then, we can reformulate Eq.4 as:

$$D\left(P_S, P_T\right) = \inf_{\boldsymbol{h}_1^s,\ldots,\boldsymbol{h}_{n_s}^s\in\mathcal{H}^s, \boldsymbol{h}_1^t,\ldots,\boldsymbol{h}_{n_t}^t\in\mathcal{H}^t} \frac{1}{n_s\times n_t}\sum_i^{n_s}\sum_j^{n_t}\left\{\inf_{c\in\mathcal{C}}\int_0^1 \ell\left(f\left(\mathcal{P}_\lambda\left(\boldsymbol{h}_i^s, \boldsymbol{h}_j^t\right)\right), \mathcal{P}_\lambda\left(\boldsymbol{y}_i^s, \boldsymbol{y}_j^t\right)\right) p(\lambda)\mathrm{d}\lambda\right\},$$

where $h_i^s = f(x_i^s)$ and $h_j^t = f(x_j^t)$.

Theorem 2 *:Let $\mathcal{H}^s$ and $\mathcal{H}^t$ be a vector space with $\dim(\mathcal{H})$ for the source and target domains, respectively. Let $d \in \mathbb{N}$ be the number of classes. If $\dim(\mathcal{H}) \geq d-1$, $\lambda^\star\ell(c^\star(f^\star(\boldsymbol{x}_i)), \boldsymbol{y}^s) + (1-\lambda^\star)\ell(c^\star(f^\star(\boldsymbol{x}_i)), \boldsymbol{y}^t) = 0$, then $D\left(P_S, P_T\right) = 0$ and the corresponding minimizer $c^\star$ is a linear function from $\mathcal{H}$ to $\mathbb{R}^d$.*

Proof: First, the following statement is true if $\dim(\mathcal{H}) \geq d-1$:

$$\exists A, H \in \mathbb{R}^{\dim(\mathcal{H})\times d}, b \in \mathbb{R}^d : A^\top H + b_d^\top = I_{d\times d},$$

where $I_{d\times d}$ and 1_d denote the d-dimensional identity matrix and all-one vector, respectively. In fact, $b_d^\top$ is a rank-one matrix, while the rank of identity matrix is d. So $A^\top H$ only needs to be a matrix with the rank $d-1$.

Then, let $c^\star(h) = A^\top h + b$, $\forall h \in \mathcal{H}$, $f^\star\left(x_i^s\right) = H_{\zeta_i^s,:}$ and $f^\star\left(x_j^t\right) = H_{\zeta_j^t,:}$ be the ζ_i-th and ζ_j-th slice of H, respectively, where $\zeta_i^s, \zeta_i^t \in \{1,\ldots,d\}$ stands for the class-index of the examples x_i^s and x_j^t. These choices minimize Eq.4, since:

$$\ell\left(c^\star\left(\mathcal{P}_{\lambda^\star}\left(f^\star\left(x_i^s\right), f^\star\left(x_j^t\right)\right)\right), \mathcal{P}_{\lambda^\star}\left(y_i^s, y_j^t\right)\right) = \lambda\ell\left(c^\star\left(f^\star\left(x_{ij}^i\right)\right), y_i^s\right) + (1-\lambda)\ell\left(c^\star\left(f^\star\left(x_{ij}^i\right)\right), y_j^t\right),$$

where the intermediate domain sample x_{ij}^i is obtained by mixing the sample x_i^s and x_j^t with PatchMix $\mathcal{P}_{\lambda^\star}$.

If $\lambda^\star\ell\left(c^\star\left(f^\star\left(x_{ij}^i\right)\right), y_i^s\right) + (1-\lambda^\star)\ell\left(c^\star\left(f^\star\left(x_{ij}^i\right)\right), y_j^t\right) = 0,$

then we can get $\ell\left(c^{\star}\left(\mathcal{P}_{\lambda^{\star}}\left(f^{\star}\left(x_i^s\right), f^{\star}\left(x_j^t\right)\right)\right), \mathcal{P}_{\lambda^{\star}}\left(y_i^s, y_j^t\right)\right)=0$,

and

$$\ell\left(c^{\star}\left(\mathcal{P}_{\lambda^{\star}}\left(f^{\star}\left(x_i^s\right), f^{\star}\left(x_j^t\right)\right)\right), \mathcal{P}_{\lambda^{\star}}\left(y_i^s, y_j^t\right)\right)=\ell\left(A^{\top} \mathcal{P}_{\lambda^{\star}}\left(H_{\zeta_i,:}^s, H_{\zeta_j,:}^t\right)+b, \mathcal{P}_{\lambda^{\star}}\left(y_{i,\zeta_i}^s, y_{j,\zeta_j}^t\right)\right)=0 .$$

The result follows from $A^{\top} H_{\zeta_i^s,:}+b=y_{i,\zeta_i^s}$ for all i, and $A^{\top} H_{\zeta_j^t,:}+b=y_{i,\zeta_j^t}$ for all j. Then, in the feature space, $f(x^s)$ and $f(x^t)$ can be mapped into the output with the same linear function, which means that P_S and P_T are the same distribution and the two domains are aligned well. Therefore, in this work, we utilize the $\lambda^{\star}\ell(c^{\star}(f^{\star}(\boldsymbol{x}_i)), \boldsymbol{y}^s)+(1-\lambda^{\star})\ell(c^{\star}(f^{\star}(\boldsymbol{x}_i)), \boldsymbol{y}^t)$ to measure the domain gaps between the intermediate domain and other domains, and finally decrease the domain divergence between the source and target domains.

We measure the domain gap in the feature space based on the above analysis. Specifically, we use the cross entropy loss ℓ to measure the discrepancy between the intermediate and other two domains.

C Details

C.1 Datasets

To evaluate the proposed method, we conduct extensive experiments on four popular UDA benchmarks, including Office-31 Saenko et al. (2010), Office-Home Venkateswara et al. (2017), VisDA-2017 Peng et al. (2017), and DomainNet Peng et al. (2019). **Office-31** consists of 4110 images of 31 categories, with three domains: Amazon (A), Webcam (W), and DSLR (D). **Office-Home** is collected from four domains: Artistic images (A), Clip Art (C), Product images (P), and Real-World images (R) and consists of 15500 images from 65 classes. **VisDA-2017** is a more challenging dataset for synthetic-to-real domain adaptation. We set 152397 synthetic images as the source domain data and 55388 real-world images as the target domain data. **DomainNet** is a large-scale benchmark dataset, which has 345 classes from six domains (Clipart (clp), Infograph (inf), Painting (pnt), Quickdraw (qdr), Real (rel), and Sketch (skt)).

C.2 Attention Map

We calculate the attention score in two ways based on whether the CLS token is present in the sequence. For Swin Transformer, we adopt a method similar to CAM Zhou et al. (2015) instead of changing the backbone from CNN to Transformer. Specifically, for a given image, let $f_k(x, y)$ represent the encoded patch k in the last layer at spatial location (x, y). The output of Transformer is followed by a global average pooling (GAP) layer$\sum(x, y)$ and a linear classification head. For the specific class C_i, the classification score S_{C_i} is:

$$S_{C_i}=\sum_j w_j^{C_i} \sum_{x, y} f_k(x, y), \tag{5}$$

where $w_j^{C_i}$ represents the weight corresponding to class C_i for unit j in the hidden dimension. Eq.5 ensembles the semantics over both spatial contexts $\sum(x, y)$ and the linear head units $\sum(j)$. Then given Eq.5, as shown in Fig.4(a), for a given C_i, we reallocate the semantic information from the output of linear head unit of C_i. In detail, we define the semantic activation map at location (x, y) for a specific class C_i as:

$$M_{C_i}(x, y)=\sum_j w_j^{C_i} f_k(x, y),$$

where $M_{C_i} \in \mathbb{R}^2$ is the activation for class C_i, and we infer C_i by the ground-truth label in the source domain and the pseudo-label in the target domain to obtain the corresponding class activation map to build the intermediate domain. Then, we use M_{C_i} as the attention map after the softmax operation.

On the other hand, when the CLS token is present in the output sequence of Transformer like Deit/ViT, we simply take the attention scores from the self-attention, *i.e.* the similarity matrix of each layer i in Transformer $Attn_i \in \mathcal{R}^{H \times N \times N}$, and take the average in the head dimension H:

$$Attn_i=\frac{1}{H} \sum_h Attn_h,$$

where N is the sequence length. Next, we only take the CLS token's attention after the softmax operation, as shown in Fig.4(a), and then summarize each layer's scores to obtain the final attention scores $Attn$.

$$Attn = \frac{1}{I}\sum_{i} Attn_i.$$

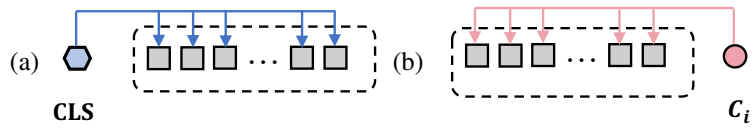

Figure 4: (a) Deit/ViT attention scores with the CLS token. (b) Swin attention scores with an output unit of Classifier that refers to C_i . The dashed line denotes the sequence with each square representing a patch.

C.3 Semi-supervised mixup loss in the feature space

In Fig.5, we illustrate the semi-supervised loss in the feature space by similarity between features (in Fig. 5 (a)) and label spaces(in Fig.5 (b)). To compute the similarity of features, we use the normalized cosine similarity loss between the intermediate domain (column) and source/target domain(row) in the feature space, as shown in Fig.5(a). Each row denotes the normalized similarity between a sample of the intermediate domain and counterparts from the source domains. For example, we first use the cosine similarity to calculate the similarities between one intermediate sample "car" and four sources (or target) samples (car, clock, apple, sketch clock). Then we normalize these similarities. As for the similarity of outputs (or) labels, since the source samples are labeled, and the target samples are unlabeled, we design two different methods to calculate the supervised and unsupervised label similarities. As for the label similarity between the intermediate and source domains, the intermediate and source samples both share the same labels. Therefore, we define the label similarity $y^{is} = y^s(y^s)^\intercal$, as shown in Fig.5 (b). Specifically, y^{is}, denoted by the yellow and pink colors, indicates that the label similarity between samples is one for these samples with the same labels (zero for different labels). For example, the label similarities between one intermediate sample "sketch clock" and four sources (or target) samples car, clock, apple, and sketch clock are zero, one, zero, and one. As for the label similarity between the intermediate and target samples, we only know that the intermediate and source samples both share overlapped patches due to lack of supervision. Therefore, the label similarity y^{it} between samples with overlapped patches should be one (pink color), and others should be zero. And we define the label similarity y^{it} as identity matrix. For example, the label similarities between one unlabeled intermediate sample "sketch clock" and four unlabeled target samples car, real clock, apple, and sketch clock are zero, zero, zero, and one. After obtaining the feature and label similarities, we utilize the CE loss ℓ to measure the discrepancy between these similarities as the domain gap between the intermediate and other domains.

C.4 Optimization

In our game, m-th player is endowed with a cost function J_m and strives to reduce its cost, which contributes to the change of CE. We now define each player's cost function J_m as

$$\begin{aligned} J_1\left(\boldsymbol{\omega}_1, \boldsymbol{\omega}_{-1}\right) &:= \mathcal{L}_{cls}(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2) + \alpha CE_{s,i,t}(\boldsymbol{\omega}), \\ J_2\left(\boldsymbol{\omega}_2, \boldsymbol{\omega}_{-2}\right) &:= \mathcal{L}_{cls}(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2) + \alpha CE_{s,i,t}(\boldsymbol{\omega}), \\ J_3\left(\boldsymbol{\omega}_3, \boldsymbol{\omega}_{-3}\right) &:= -\alpha CE_{s,i,t}(\boldsymbol{\omega}), \end{aligned} \tag{6}$$

where α is the trade-off parameter, $\mathcal{L}_{cls}(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2)$ is the supervised classification loss for the source domain, and $CE_{s,i,t}(\boldsymbol{\omega})$ is the discrepancy between the intermediate domain and source/target domain.

To clarify the min-max process, we introduce the game's vector field $v(w)$, which is identical to the gradient for every player.

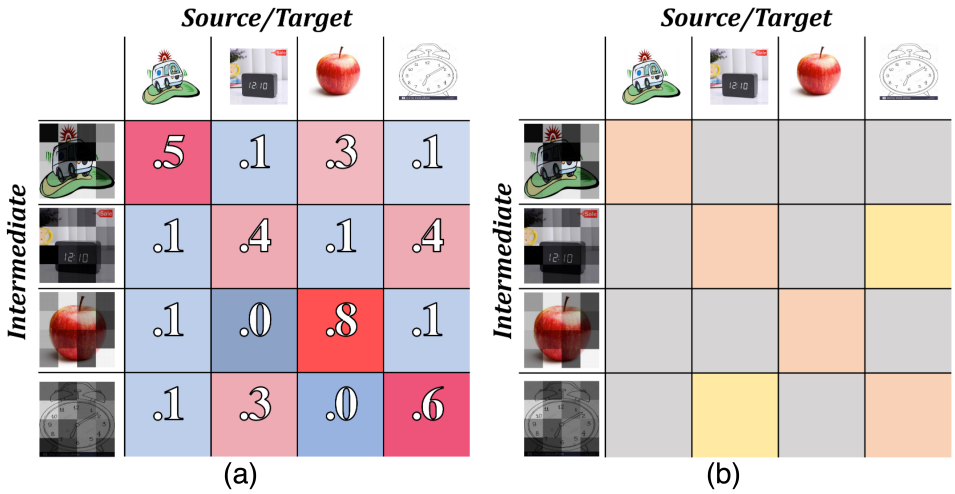

Figure 5: The illustration of the semi-supervised loss in the feature space.

Definition 3 *(Vector field):* .

$$v(\boldsymbol{\omega}) := (\bigtriangledown_{\omega_1} J_1, \bigtriangledown_{\omega_2} J_2, \bigtriangledown_{\omega_3} J_3)$$

By examining Definition.3 with respect to Eq.(6), the process can be categorized into both cooperation and competition Acuna et al. (2022b).

$$v(w) = \begin{pmatrix} \bigtriangledown_{\boldsymbol{\omega}_1} \mathcal{L}_{cls}(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2) \\ \bigtriangledown_{\boldsymbol{\omega}_2} \mathcal{L}_{cls}(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2) \\ 0 \end{pmatrix} + \begin{pmatrix} \alpha \bigtriangledown_{\boldsymbol{\omega}_1} CE_{s,i,t}(\boldsymbol{\omega}) \\ \alpha \bigtriangledown_{\boldsymbol{\omega}_2} CE_{s,i,t}(\boldsymbol{\omega}) \\ -\alpha \bigtriangledown_{\boldsymbol{\omega}_3} CE_{s,i,t}(\boldsymbol{\omega}) \end{pmatrix}, \tag{7}$$

where the left part is related to the gradient of $\mathcal{L}_{cls}(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2)$, and the right part denotes the adversarial behavior on producing or consuming CE in the network. In this Min-max CE Game, each player behaves selfishly to reduce their cost function. This competition on the network's CE will possibly end with a situation where no one has anything to gain by changing only one's strategy, called NE. Note that our method does not require explicit usage of gradient reverse layers as the prior GAN-based game design Ganin & Lempitsky (2015). Our training is optimized as

$$v(\boldsymbol{\omega}) = \bigtriangledown_{(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2)} \mathcal{L}_{cls}(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2) + \alpha \bigtriangledown_{\boldsymbol{\omega}} CE_{s,i,t}(\boldsymbol{\omega}) \tag{8}$$

C.5 Comparisons with Mixup variants

In Fig. 6, we show the visual comparisons between the PatchMix and mainstream Mixup variants. Mixup Zhang et al. (2018) mixes two samples by interpolating both the images and labels, which suffers from the local ambiguity. CutOut Devries & Taylor (2017) proposes to randomly mask out square regions of input during training to improve the robustness of the CNNs. Since CutOut decreases the ImageNet localization or object detection performances, CutMix Yun et al. (2019) is further introduced to randomly cut and paste the regions in an image, where the ground truth labels are also mixed proportionally to the area of the regions. However, sometimes there is no valid object in the mixed image due to the random process in augmentation, but there is still a response in the label space. Therefore, not all pixels are created equal, and the labels of pixels should be re-weighted. TransMix Chen et al. (2021) is proposed to utilize the attention map to assign the confidence for the mixed samples and re-weighted the labels of pixels. In comparison, we unify these global and local mixup techniques in our PatchMix by learning to combine two patches to form a mixed patch and obtain mixed samples. Furthermore, we also learn the hyperparameters of the mixup ratio for each patch and effectively build up the intermediate domain samples.

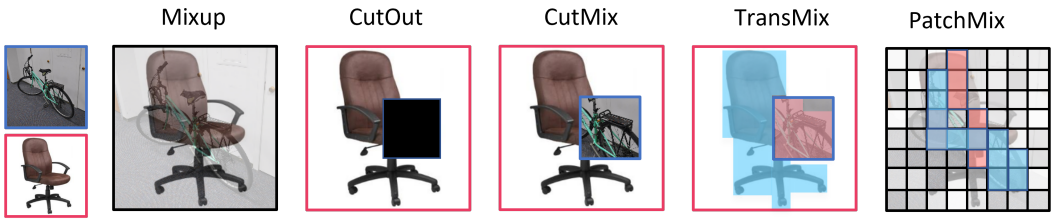

Figure 6: PMTrans and Mixup variants

D ALGORITHM

In summary, the whole algorithm to train the proposed PMTrans is shown in Algorithm 1.

Algorithm 1 Patch-Mix Transformer for Unsupervised Domain Adaptation

Require: source domain data $\mathcal{D}_s$ and target domain data $\mathcal{D}_t$.
Ensure: learned parameters of feature extractor $\boldsymbol{\omega}_1$, classifier $\boldsymbol{\omega}_2$, and PatchMix $\boldsymbol{\omega}_3$.

for $k = 0$ to MaxIter **do**
  Sample a batch of input from source data and target data.
  Encode the patches of source and target inputs by the patch embedding (Emb) layer.
  Calculate the normalized attention score for each patch as Section C.2.
  Sample the mixup ratio from Beta($\boldsymbol{\omega}_3$)
  Construct the intermediate domain input as shown in Eq. 9.
  Calculate the semi-supervised mixup loss in the feature space via Eq. 11.
  Calculate the semi-supervised mixup loss in the label space via Eq. 12.
  Measure the domain divergence between the intermediate domain and other two domains via Eq. 14.
  Update network parameters $\boldsymbol{\omega}$ by optimization (8) via a AdamW Loshchilov & Hutter (2019) optimizer.
end for
return $\boldsymbol{\omega}_1$, $\boldsymbol{\omega}_2$, and $\boldsymbol{\omega}_3$

where the related loss functions are shown as follows.

$$\begin{aligned}\mathcal{P}_\lambda(\boldsymbol{x}^s, \boldsymbol{x}^t) &= \sum_{k=1}^{n}(\lambda_k \boldsymbol{x}_k^s + (1-\lambda_k)\boldsymbol{x}_k^t),\\ \mathcal{P}_\lambda(\boldsymbol{y}^s, \boldsymbol{y}^t) &= \lambda^s \boldsymbol{y}^s + \lambda^t \boldsymbol{y}^t,\end{aligned} \tag{9}$$

$$\begin{aligned}\lambda^s &= \frac{\sum_{k=1}^{n}\lambda_k a_k^s}{\sum_{k=1}^{n}\lambda_k a_k^s + \sum_{k=1}^{n}(1-\lambda_k)a_k^t},\\ \lambda^t &= \frac{\sum_{k=1}^{n}(1-\lambda_k) a_k^t}{\sum_{k=1}^{n}\lambda_k a_k^s + \sum_{k=1}^{n}(1-\lambda_k)a_k^t}.\end{aligned} \tag{10}$$

$$\begin{aligned}\mathcal{L}_f^{I,S}(\boldsymbol{\omega}_1, \boldsymbol{\omega}_3) &= \mathbb{E}_{(\boldsymbol{x}^i, \boldsymbol{y}^i)\sim D^i}\lambda^s \ell\left(d\left(\mathcal{F}\left(\boldsymbol{x}^i\right), \mathcal{F}\left(\boldsymbol{x}^s\right)\right), \boldsymbol{y}^{is}\right),\\ \mathcal{L}_f^{I,T}(\boldsymbol{\omega}_1, \boldsymbol{\omega}_3) &= \mathbb{E}_{(\boldsymbol{x}^i, \boldsymbol{y}^i)\sim D^i}\lambda^t \ell\left(d\left(\mathcal{F}\left(\boldsymbol{x}^i\right), \mathcal{F}\left(\boldsymbol{x}^t\right)\right), \boldsymbol{y}^{it}\right).\end{aligned} \tag{11}$$

$$\begin{aligned}\mathcal{L}_l^{I,S}(\boldsymbol{\omega}) &= \mathbb{E}_{(\boldsymbol{x}^i, \boldsymbol{y}^i)\sim D^i}\lambda^s \ell\left(\mathcal{C}\left(\mathcal{F}\left(\boldsymbol{x}^i\right)\right), \boldsymbol{y}^s\right),\\ \mathcal{L}_l^{I,T}(\boldsymbol{\omega}) &= \mathbb{E}_{(\boldsymbol{x}^i, \boldsymbol{y}^i)\sim D^i}\lambda^t \ell\left(\mathcal{C}\left(\mathcal{F}\left(\boldsymbol{x}^i\right)\right), \hat{\boldsymbol{y}}^t\right).\end{aligned} \tag{12}$$

$$CE_{s,i,t}(\boldsymbol{\omega}) = \mathcal{L}_f(\boldsymbol{\omega}_1, \boldsymbol{\omega}_3) + \mathcal{L}_l(\boldsymbol{\omega}). \tag{13}$$

$$J(\boldsymbol{\omega}) := \mathcal{L}_{cls}(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2) + \alpha CE_{s,i,t}(\boldsymbol{\omega}). \tag{14}$$

Note that these above equations are introduced in detail in the main paper.

Table 8: Comparison with SoTA methods on DomainNet. The best performance is marked as **bold**.

CDTrans	clp	inf	pnt	qdr	rel	skt	Avg
clp	-	29.4	57.2	26.0	72.6	58.1	48.7
inf	57.0	-	54.4	12.8	69.5	48.4	48.4
pnt	62.9	27.4	-	15.8	72.1	53.9	46.4
qdr	44.6	8.9	29.0	-	42.6	28.5	30.7
rel	66.2	31.0	61.5	16.2	-	52.9	45.6
skt	69.0	29.6	59.0	27.2	72.5	-	51.5
Avg	59.9	25.3	52.2	19.6	65.9	48.4	45.2
SSRT	clp	inf	pnt	qdr	rel	skt	Avg
clp	-	33.8	60.2	19.4	75.8	59.8	49.8
inf	55.5	-	54.0	9.0	68.2	44.7	46.3
pnt	61.7	28.5	-	8.4	71.4	55.2	45.0
qdr	42.5	8.8	24.2	-	37.6	33.6	29.3
rel	69.9	37.1	66.0	10.1	-	58.9	48.4
skt	70.6	32.8	62.2	21.7	73.2	-	52.1
Avg	60.0	28.2	53.3	13.7	65.3	50.47	45.2
PMTrans	clp	inf	pnt	qdr	rel	skt	Avg
clp	-	34.2	62.7	32.5	79.3	63.7	54.5
inf	67.4	-	61.1	22.2	78.0	57.6	57.3
pnt	69.7	33.5	-	23.9	79.8	61.2	53.6
qdr	54.6	17.4	38.9	-	49.5	41.0	40.3
rel	74.1	35.3	70.0	25.4	-	61.1	53.2
skt	73.8	33.0	62.6	30.9	77.5	-	55.6
Avg	67.9	30.7	59.1	27.0	72.8	56.9	62.9

E Results and Analyses

E.1 The comparisons on the DomainNet

To take a closer look at the results, we choose the results of Transformer-based methods *e.g.*, CDTrans Xu et al. (2021), SSRT Sun et al. (2022), and PMTrans, for a fair comparison, as shown in Table. 8. The qualitative results show that our proposed PMTrans outperform other Transformer-based methods on each sub-tasks. The results indicate that our PMTrans is most effective in measuring the domain gaps and demonstrate the effectiveness of bridging the domains in the min-max CE game.

E.2 Representation visualization

We plot in Fig. 7 feature representations learned by Swin-Base, PMTrans-Swin, PMTrans-ViT, and PMTrans-Deit on task A $\rightarrow$ C from the Office-Home dataset via the t-SNE method Donahue et al. (2014). Compared with Swin-Base and PMTrans-Swin, the proposed PMTrans model can better align two domains by constructing the intermediate domain for bridging the two domains. Moreover, comparisons between PMTrans with different transformer backbones reveal that PMTrans works successfully for different backbones on UDA tasks.

E.3 Attention map visualization for target data

We randomly sample four images from Product (P) of Office-Home and use the pre-trained models $C \rightarrow P$ including PMTrans-Swin and PMTrans-Deit to infer the attention maps following the methods described in Sec.C.2. In Fig. 8, we compare the two PMTrans with their counterparts trained with only source classification loss. We observe that after domain alignment, the attention

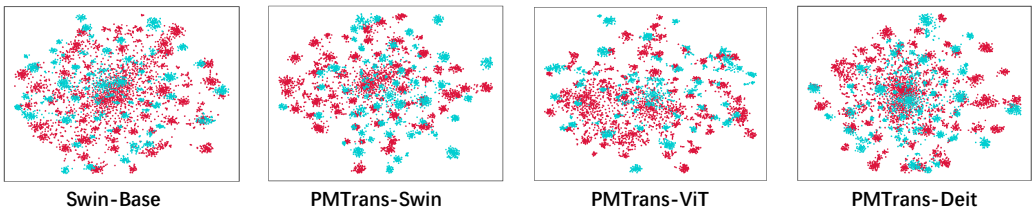

Figure 7: t-SNE visualizations for the transfer task A→C on the Office-Home dataset. Source and target instances are shown in red and blue, respectively.

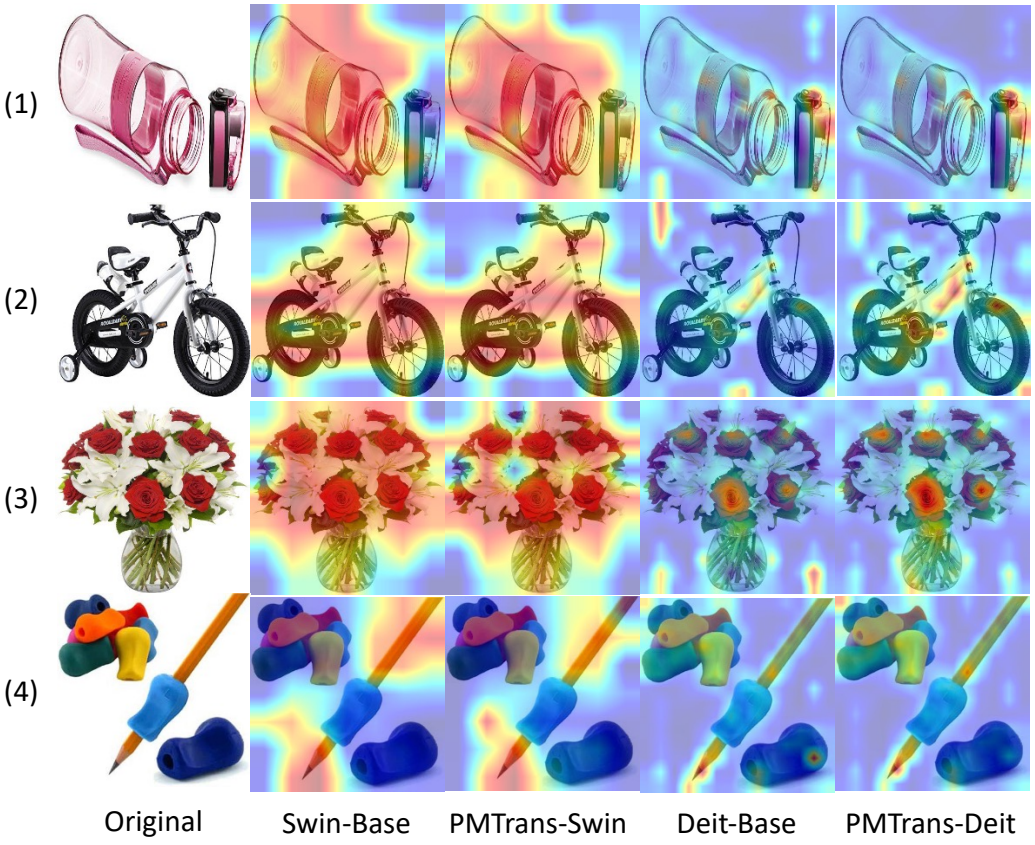

Figure 8: Attention visualization on Swin-based and Deit-based backbones.

maps tend to be more focused on the objects *i.e.*less noise around them. Interestingly, for the image whose ground truth label is `pencil` in the fourth row, Swin-based backbone can distinguish it from `plasticine` around or attached to it. At the same time, Deit-based attention covers them all, which may bring negative effects. When the attention scores are used to scale the weights of patches during constructing the intermediate domain in Eq.10, Swin-based architecture can focus more on semantics while others may not. That may be one of the reasons why PMTrans-Swin gets superior performance on many datasets. Similarly, TS-CAM Gao et al. (2021) names the original attention scores from Transformer like Fig.4(a) as semantic-agnostic, while what we do in Fig.4(b) is to reallocate the semantics from Classifier back into the patches and make it be aware of specific class activation.

E.4 TRAINING

We show the progress of training on PMTrans-Swin, PMTrans-Deit, and PMTrans-ViT. To specify how each loss changes, including semi-supervised mixup loss in the label space $\mathcal{L}_l$, semi-supervised

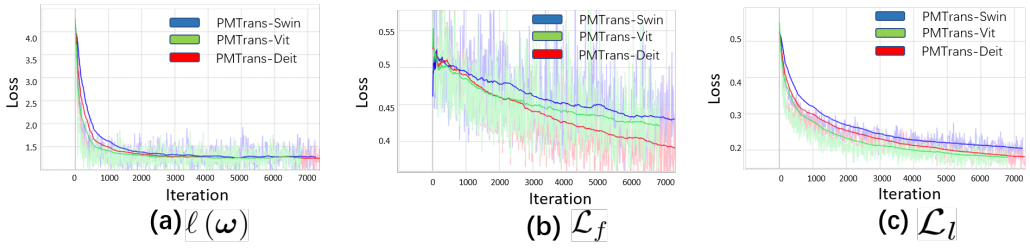

Figure 9: Loss on the task A $\rightarrow$ C (Office-Home). Lines are smoothed for clarity.

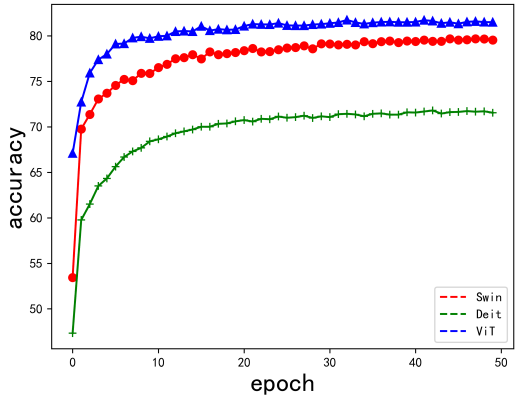

Figure 10: Accuracy on the task A $\rightarrow$ C (Office-Home)

mixup loss in the feature space $\mathcal{L}_f$, and source classification loss $\mathcal{L}_{cls}(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2)$, we conduct the experiment on task $A \rightarrow C$ on Office-Home for above architectures, and the results are shown in Fig.9. We observe that for all models, both $\mathcal{L}_f$ and $\mathcal{L}_l$ drop constantly, which means the domain gap is reducing as the training evolves. Significantly, $\mathcal{L}_f$ fluctuates more than $\mathcal{L}_l$ as it aligns the domains in the feature space with a higher dimension.

E.5 Testing

In Fig.10, we testify PMTrans-Swin, PMTrans-ViT, and PMTrans-Deit on the task A $\rightarrow$ C on the Office-Home dataset. From Fig. 10, with the same number of epochs, PMTrans-ViT achieves faster convergence than PMTrans-Swin and PMTrans-Deit. Besides, the results further reveal that our proposed PMTrans with different transformer backbones can bridge the source and target domains well and decrease domain divergence effectively.

E.6 Complexity

We compare our computational budget with the typical work CDTrans Xu et al. (2021) on aligning the source and target domains, excluding the choice of backbone. Precisely, CDTrans compute the similarity between patches from two domains by the multi-head self-attention. We are given n as the sequence length, d as the representation dimension, and c as the number of classes. The per-layer complexity is $O(n^2 d)$. While in PMTrans, we adopt CE loss to close the domain gap on both the feature and label spaces of the out, whose complexity is $O(d)+O(c)$. When building the intermediate domain, PatchMix samples patches element-wisely, and its complexity is $O(n)$. As attention scores we use are taken directly from the parameters of Transformer and Classifier, so it brings no additional cost. PMTrans's complexity is $O(d + c + n)$, so it is much more lightweight than the cross attention in CDTrans.

Table 9: Comparisons between different backbones with different batch sizes on VisDA-2017. The best performance is marked as **bold**.

backbone	plane	bcycl	bus	car	horse	knife	mcycl	person	plant	sktbrd	train	truck	Avg
ViT-bs8	99.0	92.7	84.3	68.0	**99.1**	**98.5**	96.4	37.6	93.6	98.5	96.7	48.2	84.4
ViT-bs16	99.1	91.9	85.9	69.7	99.0	**98.5**	96.5	43.1	93.8	**99.2**	**96.9**	**50.5**	85.3
ViT-bs24	98.8	92.8	84.5	71.1	**99.1**	98.3	**96.7**	58.9	93.8	98.8	96.7	47.7	86.4
ViT-bs32	98.9	93.7	84.5	73.3	99.0	98.0	96.2	67.8	94.2	98.4	96.6	49.0	87.5
Deit-bs8	98.1	89.5	86.9	73.5	97.5	96.9	95.7	71.8	96.3	92.1	95.6	45.5	86.6
Deit-bs16	98.3	90.0	87.0	74.2	97.4	96.9	95.7	72.2	96.7	92.2	95.8	46.5	86.9
Deit-bs24	98.2	90.2	87.0	74.8	97.5	96.8	95.7	**73.2**	96.8	92.1	95.6	46.9	87.1
Deit-bs32	98.2	**92.2**	**88.1**	77.0	97.4	95.8	94.0	72.1	**97.1**	95.2	94.6	51.0	87.7
Swin-bs8	99.3	87.3	87.7	66.9	98.8	98.1	96.4	57.5	95.2	98.0	96.5	44.2	85.5
Swin-bs16	99.2	87.6	87.5	66.4	98.8	98.3	96.3	58.4	95.4	98.0	96.5	44.6	85.6
Swin-bs24	99.2	88.1	87.3	67.1	98.7	98.2	96.1	67.1	94.0	97.9	96.3	44.2	86.2
Swin-bs32	**99.4**	88.3	**88.1**	**78.9**	98.8	98.3	95.8	70.3	94.6	98.3	96.3	48.5	**88.0**

Table 10: Effect of semi-supervised loss with class information. The best performance is marked as **bold**.

Method	A→ C	A→ P	A → R	C → A	C → P	C → R	P→ A	P→ C	P→ R	R→ A	R→ C	R→ P	Avg
w/o class information	79.7	**92.3**	92.6	88.3	**93.1**	92.8	87.3	**80.0**	92.8	88.8	**79.8**	94.6	88.5
w/ class information	**80.1**	92.2	**92.9**	**88.7**	92.8	**93.5**	**87.9**	79.9	**93.0**	**89.2**	79.0	**95.0**	**88.7**

F Ablation Study

F.1 Batch size

In Table 9, we study the effect of the batch size with different backbones in our proposed PMTrans framework. As shown in Table 9, when the batch size is bigger, the input can represent the data distributions better, and therefore the proposed PMTrans based on different backbones with larger batch sizes generally achieves better performance in UDA tasks. Considering the hardware limit, we cannot train models with a batch size of more than 32, so our performance may be lower than it could be, especially when putting in the same condition with a 64 batch size as many previous works do.

F.2 Semi-supervised mixup loss with class information

Table 10 shows the comparisons between PMTrans, where the semi-supervised mixup loss combines the class information of target data or not. Note that we use the pseudo labels of target data to calculate the discrepancy between the features and labels. We agree that the semi-supervised mixup loss with class information decreases the domain gaps by reducing the disparity between the feature and label similarities with supervised techniques.