Patch-Mix Transformer for Unsupervised Domain Adaptation: A Game Perspective

Jinjing Zhu\(^1\) \* Haotian Bai\(^1\) \* Lin Wang\(^{1,2}\)

\(^1\) AI Thrust, HKUST(GZ) \quad ^2\) Dept. of CSE, HKUST

zhujinjing.hkust@gmail.com, haotianwhite@outlook.com, linwang@ust.hk

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 targeted samples, it becomes less effective when the domain gap becomes large. We solve this problem from a game theory’s perspective with the proposed model dubbed as PMTrans, which bridges source and 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. This way, it learns to mix the patches from the 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 Nash Equilibria. 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 +3.6% on Office-Home, +1.4% on Office-31, and +17.7% on DomainNet, respectively. The classification accuracy of our PMTrans surpasses the SoTA methods by +17.7% on the most challenging DomainNet dataset. Note that the target tasks treat one domain of DomainNet as the target domain and the others as the source domains.

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 [50]. Unsupervised domain adaptation (UDA) tackles this issue by transferring knowledge from a labeled source domain to an unlabeled target domain [30]. 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 [14,53], adversarial training [12,17,34], and optimal transport [44]. Furthermore, several works [11, 36] explore the potential of ViT for the non-trivial UDA task. Recently, CDTrans [45] 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, due to the significant gap between the domain qdr and the other domains, aligning distributions directly between them 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 facili-
We conduct experiments on four benchmark datasets, including Office-31 [33], Office-Home [40], VisDA-2017 [32], and DomainNet [31]. The results show that the performance of PMTrans significantly surpasses that of the ViT-based [36,45,46] and CNN-based SoTA methods [18,29,35] by +3.6% on Office-Home, +1.4% on Office-31, and +17.7% on DomainNet (See Fig. 1), respectively.

Our main contributions are four-fold: (I) We propose a novel ViT-based UDA framework, PMTrans, to effectively bridge 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 min-max CE game. (IV) Our PMTrans surpasses 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 [14,25,26,52] use the maximum mean discrepancy (MMD) loss to measure the divergence between different domains. In addition, the central moment discrepancy (CMD) loss [48] and maximum density divergence (MDD) loss [16] 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 [13,42,43]. The third type of approach aims to minimize the cost transported from the source to the target distribution by finding an optimal coupling cost to mitigate the domain shift [6,7]. Unfortunately, these methods are not robust enough for the noisy pseudo target labels for accurate domain alignment. Different from these mainstream UDA methods and [2], we interpret the process of UDA as a min-max CE game and find the 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 by linearly interpolating two input data. Mixup types can be categorized into global mixup (e.g., Mixup [49] and Manifold-Mixup [41]) and local mixup (CutMix [47], saliency-CutMix [38], TransMix [5], and Tokenmix [22]). In CNN-based UDA tasks, several works [29,42,43] 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 which effectively builds up the intermediate domain by sampling patches from both domains based on the game-theoretical models.
Transformer. Vision Transformer (ViT) [39] has recently been introduced to tackle the challenges in various vision tasks [4, 23]. Several works have leveraged ViT for the non-trivial UDA task. TVT [46] proposes an adaptation module to capture domain data’s transferable and discriminative features. SSRT [36] proposes a framework with a transformer backbone and a safe self-refinement strategy to handle the issue in case of a large domain gap. More recently, CD-Trans [45] proposes a two-step framework that utilizes the cross-attention in ViT for direct feature alignment and 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.

3. Methodology

In UDA, denote a labeled source set \( D_s \) = \{ \( (x_i, y_i) \}_{i=1}^{n_s} \) with \( i \)-th sample \( x_i \) and its corresponding one-hot label \( y_i \) and an unlabeled target set \( D_t \) = \{ \( x_i \)\}_{i=1}^{n_t} \) with \( j \)-th sample \( x_j \), \( n_s \) and \( n_t \) as 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 divergence issue and smoothly transfer the knowledge from the source domain to the target domain. Firstly, we define and introduce PatchMix, and interpret the process of UDA as a min-max CE game. Secondly, we propose the proposed PMTrans which smoothly aligns the source and target domains by constructing an intermediate domain via a three-player game.

3.1. PMTrans: Theoretical Analysis

3.1.1. PatchMix

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

\[
\begin{align*}
  x^i_k &= P_\lambda(x^*, x^t), \quad x^*_k = \lambda_k \odot x^*_k + (1 - \lambda_k) \odot x^t_k, \\
  y^i_k &= P_\lambda(y^*, y^t) = \frac{\sum_{k=1}^{n} \lambda_k y^* + \sum_{k=1}^{n} (1 - \lambda_k) y^t}{n}.
\end{align*}
\]

where \( x^i_k \) is the \( k \)-th patch of \( x^i \), and \( \odot \) denotes multiplication. In Definition 1, each image \( x^i \) of the intermediate domain composes the sampled patches \( x_k \) from the 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 utilized to interpolate their labels. As a result, we mix both samples \( (x^i, y^i) \) and \( (x^t, y^t) \) to construct a new intermediate domain \( D_i = \{ (x^i_k, y^i) \}_{k=1}^{n_i} \). To align the source and 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 \( D_t \), respectively. The domain divergence between source and target domains can be measured as

\[
D(P_S, P_T) = \inf_{f \in \mathcal{F}, \{ P_s \}, \{ P_t \} \in \mathcal{T}} \mathbb{E}_{(x^s, y^s) \sim P_s} \mathbb{E}_{(x^t, y^t) \sim P_t} [\ell \left( f \left( P_s \left( (x^s), f \left( (x^t) \right) \right), P_t \left( (y^s), y^t) \right) \right) ].
\]

where \( \mathcal{F} \) denotes a set of encoding functions i.e., the feature extractor and \( C \) denotes a set of decoding functions i.e. the classifier. Let \( \mathcal{P} \) be the set of functions to generate the mixup ratio for building the intermediate domain. Then we can reformulate Eq. 2 as

\[
D(P_S, P_T) = \inf_{h^{s_1}_1, ... h^{s_{n_s}_1} \in \mathcal{H}^s, h^{t_1}_1, ... h^{t_{n_t}_1} \in \mathcal{H}^t} \inf_{\lambda \in \Lambda} \frac{1}{n_s \times n_t} \sum_{i=1}^{n_s} \sum_{j=1}^{n_t} \left\{ \begin{array}{c}
\inf_{c \in \mathcal{C}} \ell \left( f \left( P_\lambda \left( (h^{s_i}_j), P_s \left( (y^{s_i}_j), y^{t_i}_j) \right) \right), P_\lambda \left( (y^{s_i}_j), y^{t_i}_j) \right) \right) p(\lambda) d\lambda \end{array} \right\},
\]

where \( \ell \) is CE loss, \( h^*_i = f(s_i) \) and \( h^t_j = f(t_j) \). Note \( \mathcal{H}^s \) and \( \mathcal{H}^t \) denote the representation spaces with dimensionality \( \dim(\mathcal{H}) \) for source and target domains, respectively. Let \( f^* \in \mathcal{F}, e^* \in \mathcal{C} \), and \( P^*_S \in \mathcal{P} \) be minimizers of Eq. 2.

Theorem 1 (Domain Distribution Alignment with PatchMix): Let \( d \in \mathbb{N} \) represent the number of classes contained in three sets \( D_s, D_t, \) and \( D_i \). If \( \dim(\mathcal{H}) \geq d - 1 \), \( P^*_S(\ell(e^*(f^*(x_i))), y^s) + (1 - P^*_S)\ell(e^*(f^*(x_i)), y^t) = 0 \), then \( D(P^*_S, P^*_T) = 0 \) and the corresponding minimizer \( e^* \) is a linear function from \( \mathcal{H} \) to \( \mathbb{R}^d \). Denote the scaled mixup ratio sampled from a learnable Beta distribution as \( P^*_\lambda \).

Theorem 1 indicates that the source and target domains are 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 (\( C \)), and PatchMix module (\( P \)), as shown in Fig. 3. To specify each player’s role, we define \( \omega_F \in \Omega_F, \omega_C \in \Omega_C \), and \( \omega_P \in \Omega_P \) as the parameters of \( \mathcal{F}, C, \) and \( P \), respectively. The joint domain is defined as \( \Omega = \Omega_F \times \Omega_C \times \Omega_P \) and their joint parameter set is defined as \( \omega = \{ \omega_F, \omega_C, \omega_P \} \). Then we use the subscript \( -m \) to denote all other parameters/players except \( m \), e.g., \( \omega_{-C} = \{ \omega_F, \omega_P \} \). 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

$$
J_F (\omega, \omega_m - x) := L_{FS}^C (\omega, \omega_m) + \alpha CE_{x,i,t} \omega(\omega),
$$

$$
J_C (\omega, \omega_m - c) := L_{CS}^S (\omega, \omega_m) + \alpha CE_{x,i,t} \omega(\omega),
$$

$$
J_P (\omega_m, \omega_m) := -\alpha CE_{x,i,t} \omega(\omega),
$$

where $\alpha$ is the trade-off parameter, $\ell$ is the supervised classification loss for the source domain, and $CE_{x,i,t} \omega(\omega)$ is the discrepancy between the intermediate domain and the source/target domain. The definitions of $L_{FS}^C (\omega, \omega_m)$ and $CE_{x,i,t} \omega(\omega)$ are shown in Sec. 3.2. As illustrated in Eq. 4, the game is essentially a min-max process, i.e., a competition for the player $P$ against both players $F$ and $C$. Specifically, as depicted in Fig. 3, $P$ strives to diverge while $F$ and $C$ 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. And a point $\omega^* \in \Omega$ is Nash Equilibrium if

$$
\forall \omega_m \in \Omega, \forall \omega \in \{F, C, P\}, s.t. J_m (\omega, \omega_m) \leq J_m (\omega_m, \omega_m).
$$

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. 3 illustrates the framework of our proposed PMTrans, which consists of a ViT encoder, a classifier, and a PatchMix module. PatchMix module is utilized to maximize the CE between the intermediate domain and source/target domain, conversely, two semi-supervised mixup losses in the feature and label spaces are proposed to minimize CE. Finally, a three-player game containing feature extractor, classifier, and PatchMix module, minimizes and maximizes the CE for aligning distributions between the source and target domains.

**PatchMix.** As shown in Fig. 3, the PatchMix module is proposed to construct the intermediate domain, buttressed by Definition 1. In detail, the patch embedding layer in PatchMix transforms input images from source/target domains into patches. And the ViT encoder aims to extract features from the patch sequences. 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. The PatchMix with a learnable $Beta$ distribution aims to maximize the CE between the intermediate and source/target domain, and is presented as follows.

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. [5] observed, the mixed image has no valid objects due to the random process while there is still a response in the label space. To remedy this issue, we re-weight $P_\lambda (y^*, y^*)$ 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 $P_\lambda (y^*, y^*)$ is defined as

$$
P_\lambda (y^*, y^*) = \lambda^* y^* + \lambda^t y^t,
$$

where

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

$$
\lambda^t = \frac{\sum_{k=1}^n (1 - \lambda_k) a_k^t}{\sum_{k=1}^n \lambda_k a_k^t + \sum_{k=1}^n (1 - \lambda_k) a_k^t}.
$$
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. Intuitively, we propose two semi-supervised mixup losses in the feature and label spaces to minimize the discrepancy between features of mixing patches and corresponding mixing labels based on Theorem 1. The objective of the proposed PMTrans is show in Fig. 4 (a) and consists of a classification loss of source data and two semi-supervised mixup losses.

1) Label space: 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 between the mixing logits and corresponding mixing labels (See green arrow in Fig. 4 (a)).

\[
\begin{align*}
L_i^{L,s}(\omega) &= E_{(x', y') \sim D_1} \lambda^s \ell \left( \mathcal{C} \left( \mathcal{F}(x') \right), y' \right), \\
L_i^{L,t}(\omega) &= E_{(x, y')} \sim D_1 \lambda^t \ell \left( \mathcal{C} \left( \mathcal{F}(x) \right), y' \right),
\end{align*}
\]

where \( y' \) is the pseudo label for target data. For convenience, we utilize the method, commonly used in [20, 21], to generate pseudo labels \( y' \) for samples via \( k \)-means clustering.

2) Feature space: Nonetheless, the supervised loss alone in the label space is not sufficient to diminish the domain divergence due to the less reliable pseudo labels of the target data. Therefore, we further propose to minimize the discrepancy between the similarity of the features and the similarity of labels in the feature space for aligning the intermediate and source/target domain without the supervised information of the target domain. The experimental results in Tab. 5 validate its effectiveness.

Specifically, we first compute the cosine similarity between the intermediate domain and source/target domain in the feature space. The feature similarity is defined as

\[
d(x', x^*) = \cos(\mathcal{F}(x'), \mathcal{F}(x^*)),
\]

where \( \cos \) denotes the cosine similarity. As shown in Fig. 4(b), for the source domain, we exploit the ground-truth to calculate the label similarity, \( y'^s = y^s (y^s)^T \), as a binary matrix to represent whether samples share the same labels. For example, the intermediate image is constructed by sampling patches from the source image, e.g., clock; therefore, the label similarity is set as 1 if it is calculated between the intermediate image and the source class ‘clock’, otherwise, it is set as 0. 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 (See red arrow in Fig. 4 (a)) is formulated as

\[
L_i^{L,f}(\omega_f, \omega_p) = E_{(x', y') \sim D_1} \lambda^f \ell \left( \mathcal{D}(x', x^*), y'^s \right).
\]

Moreover, for the intermediate and target domains, due to lack of supervision, we utilize identity matrix \( y'^t \) as the label similarity. For example, in Fig. 4 (b), as the intermediate image is built by sampling patches from the target image, e.g., bottle; therefore, the label similarity between the intermediate image and the corresponding target image is set as 1 and vice versa. To measure the divergence between the intermediate and target domains in the feature space, we propose an unsupervised mixup loss as

\[
L_i^{L,u}(\omega_f, \omega_p) = E_{(x', y') \sim D_1} \lambda^u \ell \left( \mathcal{D}(x', x^*), y'^t \right).
\]

Finally, the two semi-supervised mixup losses in the feature and label spaces are formulated as

\[
\begin{align*}
L_i(\omega) &= L_i^{L,f}(\omega_f, \omega_p) + L_i^{L,t}(\omega_f, \omega_p), \\
&= L_i^{L,S}(\omega) + L_i^{L,T}(\omega).
\end{align*}
\]

Moreover, the classification loss is applied to the labeled source domain data (See blue arrow in Fig. 4 (a)) and is formulated as

\[
L_i^{C,S}(\omega_f, \omega_c) = E_{(x, y) \sim D_1} \lambda^c \ell \left( \mathcal{C} \left( \mathcal{F}(x) \right), y \right).
\]

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

\[
CE_{s,t,i}(\omega) = L_f(\omega_f, \omega_p) + L_i(\omega).
\]

We adopt the random mixup-ratio from a learnable 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. 4 as

\[
J(\omega) := L_i^{C,S}(\omega_f, \omega_c) + \alpha CE_{s,t,i}(\omega),
\]

where \( \alpha \) is trade-off parameter. 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, further indicating that the source and target domains are well aligned.
4. Experiments

4.1. Datasets and implementation

Datasets. To evaluate the proposed method, we conduct experiments on four popular UDA benchmarks, including Office-Home [40], Office-31 [33], VisDA-2017 [32], and DomainNet [31]. The details of the datasets and transfer tasks on these datasets can be found in the suppl. material.

Implementation. In all experiments, we use the Swin-based transformer [24] pre-trained on ImageNet [9] as the backbone for our PMTrans. The base learning rate is $5 \times 10^{-6}$ with a batch size of 32, and we train models by 50 epochs. For VisDA-2017, we use a lower learning rate $1 \times 10^{-6}$. We adopt AdamW [27] with a momentum of 0.9, and a weight decay of 0.05 as the optimizer. Furthermore, for fine-tuning purposes, we set the classifier (MLP) with a higher learning rate $1 \times 10^{-5}$ for our main tasks and learn the trade-off parameter adaptively. For a fair comparison with prior works, we also conduct experiments with the same backbone Deit-based [37] as CDTrans [45], and ViT-based [11] as SSRT [36] on Office-31, Office-Home, and VisDA-2017. These two studies are trained for 60 and 100 epochs separately.

4.2. Results

We compare PMTrans with the SoTA methods, including ResNet-based and ViT-based methods. The ResNet-based methods are FixBi [29], MCD [34], SWD [15], SCDA [19], BNM [8], and MDD [51]. The ViT-based methods are SSRT [36], CDTrans [45], and TVT [46].

For the ResNet-based methods, we utilize ResNet-50 as the backbone for the Office-Home, Office-31, 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-Home. Tab. 1 shows the quantitative results of methods 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 5.4% accuracy over the Swin backbone and yields 89.0% accuracy. Interestingly, our proposed PMTrans can decrease the domain divergence effectively with Deit-based and ViT-based backbones. The results indicate that our method can obtain more robust transferable representations than the CNN-based and ViT-based methods.

Results on Office-31. Tab. 2 shows the quantitative comparison with the CNN-based and ViT-based methods. Overall, our PMTrans achieves the best performance on each task with 95.3% accuracy and outperforms the SoTA methods with identical backbones. Numerically, PMTrans noticeably surpasses the SoTA methods with an increase of +1.4% accuracy over TVT, +2.7% accuracy over CDTrans, and +1.8% accuracy over SSRT, respectively.

Results on VisDA-2017. As shown in Tab. 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 model shows an excellent generalization capability and achieves comparable performance (88.0%)
Table 3. Comparison with SoTA methods on VisDA-2017. The best performance is marked as bold.

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

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 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.

4.3. Ablation Study

Semi-supervised mixup loss. As shown in Tab. 5, Swin with the semi-supervised mixup loss in the feature and label spaces outperforms the counterpart trained on Swin with only source training by +1.0% and +4.3% on Office-Home dataset, respectively. The results indicate the effectiveness of the semi-supervised mixup loss for diminishing the domain discrepancy. Moreover, we observe that the CE loss yields better performance on the label space than on the feature space. The reason is that the CE loss on the label space utilizing the class information performs better than on the feature space without the class information.
Figure 5. t-SNE visualizations for task A→C on the Office-Home dataset. Source and target instances are shown in blue and red, respectively.

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

<table>
<thead>
<tr>
<th>Method</th>
<th>A→C</th>
<th>A→P</th>
<th>A→R</th>
<th>C→A</th>
<th>C→P</th>
<th>C→R</th>
<th>P→A</th>
<th>P→C</th>
<th>P→R</th>
<th>R→A</th>
<th>R→C</th>
<th>R→P</th>
<th>Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>Beta(1,1)</td>
<td>79.9</td>
<td>92.0</td>
<td>92.3</td>
<td>88.6</td>
<td>92.6</td>
<td>92.4</td>
<td>86.9</td>
<td>79.0</td>
<td>92.4</td>
<td>88.2</td>
<td>79.3</td>
<td>94.0</td>
<td>88.1</td>
</tr>
<tr>
<td>Beta(2,2)</td>
<td>79.9</td>
<td>92.1</td>
<td>92.7</td>
<td>88.4</td>
<td>92.4</td>
<td>92.7</td>
<td>86.9</td>
<td>79.5</td>
<td>92.1</td>
<td>88.1</td>
<td>79.6</td>
<td>94.3</td>
<td>88.2</td>
</tr>
<tr>
<td>Learning</td>
<td>81.3</td>
<td>92.9</td>
<td>92.8</td>
<td>88.4</td>
<td>93.4</td>
<td>93.2</td>
<td>87.9</td>
<td>80.4</td>
<td>93.0</td>
<td>89.0</td>
<td>80.9</td>
<td>94.8</td>
<td>89.0</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Method</th>
<th>A→C</th>
<th>A→P</th>
<th>A→R</th>
<th>C→A</th>
<th>C→P</th>
<th>C→R</th>
<th>P→A</th>
<th>P→C</th>
<th>P→R</th>
<th>R→A</th>
<th>R→C</th>
<th>R→P</th>
<th>Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mixup</td>
<td>79.4</td>
<td>92.4</td>
<td>92.6</td>
<td>87.5</td>
<td>92.8</td>
<td>92.4</td>
<td>86.8</td>
<td>80.3</td>
<td>92.5</td>
<td>88.2</td>
<td>79.7</td>
<td>95.4</td>
<td>88.3</td>
</tr>
<tr>
<td>CutMix</td>
<td>79.2</td>
<td>91.2</td>
<td>92.2</td>
<td>87.6</td>
<td>91.8</td>
<td>91.8</td>
<td>86.0</td>
<td>77.8</td>
<td>92.6</td>
<td>88.2</td>
<td>78.4</td>
<td>94.1</td>
<td>87.6</td>
</tr>
<tr>
<td>PatchMix</td>
<td>81.3</td>
<td>92.9</td>
<td>92.8</td>
<td>88.4</td>
<td>93.4</td>
<td>93.2</td>
<td>87.9</td>
<td>80.4</td>
<td>93.0</td>
<td>89.0</td>
<td>80.9</td>
<td>94.8</td>
<td>89.0</td>
</tr>
</tbody>
</table>

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

**Learning hyperparameters of mixup.** Tab. 6 shows the ablation results for the effects of learning hyperparameters of the Beta distribution on the Office-Home. We compare the learning hyperparameters of mixup with fixed parameters, such as Beta(1,1) and Beta(2,2). The proposed method achieves +0.9% and +0.8% accuracy increment compared with that based on Beta(1,1) and Beta(2,2). The results demonstrate that learning to estimate the distribution to build up the intermediate domain facilitates domain alignment.

**PatchMix.** Comparisons of PMTrans with Mixup [49] and CutMix [47] are shown in Tab. 7. PMTrans outperforms Mixup and CutMix by +0.7% and +1.4% 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.

**Visualization.** In Fig. 5, we visualize the features learned by Swin-based, PMTrans-Swin, PMTrans-ViT, and PMTrans-Deit on task A→C from the Office-Home dataset via the t-SNE [10]. Compared with Swin-based and PMTrans-Swin, our PMTrans model can better align the two domains by constructing the intermediate domain to bridge them. Moreover, comparisons between PMTrans with different transformer backbones reveal that PMTrans works successfully with different backbones on UDA tasks.

**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 Nash Equilibrium. 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.

**Acknowledgment.** This work was supported by the National Natural Science Foundation of China (NSFC) under Grant No. NSFC22FY34.
References


