Select, Label, and Mix: Learning Discriminative Invariant Feature Representations for Partial Domain Adaptation

Aadarsh Sahoo[1], **Rameswar Panda**[2], **Rogerio Feris**[2], **Kate Saenko**[2,3], **Abir Das**[1]

[1] IIT Kharagpur, [2] MIT-IBM Watson AI Lab, [3] Boston University

{sahoo_aadarsh@, abir@cse.}iitkgp.ac.in, {rpanda@, rsferis@us.}ibm.com, saenko@bu.edu

Abstract

Partial domain adaptation which assumes that the unknown target label space is a subset of the source label space has attracted much attention in computer vision. Despite recent progress, existing methods often suffer from three key problems: negative transfer, lack of discriminability and domain invariance in the latent space. To alleviate the above issues, we develop a novel 'Select, Label, and Mix' (SLM) framework that aims to learn discriminative invariant feature representations for partial domain adaptation. First, we present an efficient "select" module that automatically filters out the outlier source samples to avoid negative transfer while aligning distributions across both domains. Second, the "label" module iteratively trains the classifier using both the labeled source domain data and the generated pseudo-labels for the target domain to enhance the discriminability of the latent space. Finally, the "mix" module utilizes domain mixup jointly with the other two modules to explore more intrinsic structures across domains leading to a domain-invariant latent space for partial domain adaptation. Extensive experiments on several benchmark datasets demonstrate the superiority of our proposed framework over state-of-the-art methods.

Introduction

Deep neural networks usually do not generalize well to domains that are not distributed identically to the training data. Domain adaptation (Csurka 2017; Wang and Deng 2018) addresses this problem by transferring knowledge from a label-rich source domain to a target domain where labels are scarce or unavailable. However, standard domain adaptation algorithms often assume that the source and target domains share the same label space (Ganin and Lempitsky 2015; Gretton et al. 2012; Long et al. 2015, 2018, 2016). Since large-scale labelled datasets are readily accessible as source domain data, a more realistic scenario is partial domain adaptation (PDA), which assumes that the target label space is a subset of the source label space, that has received increasing research attention recently (Bucci, D'Innocente, and Tommasi 2019; Chen et al. 2019b, 2020; Hu et al. 2019).

Several methods have been proposed to solve partial domain adaptation by reweighting source samples (Bucci, D'Innocente, and Tommasi 2019; Chen et al. 2019b, 2020; Hu et al. 2019; Xu et al. 2019b; Zhang et al. 2018). However, (1) most of the existing methods still suffer from negative transfer due to presence of outlier source domain classes, which cripples domain-wise transfer with untransferable knowledge; (2) in absence of labels, they often neglect the class-aware information in target domain which fails to guarantee the discriminability of the latent space; and (3) given filtering of the outliers, limited number of samples from source and target domain are not alone sufficient to learn domain invariant features for such a complex problem. As a result, a domain classifier may falsely align unlabeled target samples with samples of a different class in the source domain, leading to inconsistent predictions.

To address these challenges, we propose a novel end-to-end **Select, Label, and Mix (`SLM`)** framework for learning discriminative invariant features while preventing negative transfer in PDA. Our framework consists of three unique modules working in concert, *i.e.*, select, label and mix, as shown in Figure 1. First, the select module facilitates the identification of relevant source samples preventing the negative transfer. To be specific, our main idea is to learn a model (referred to as selector network) that outputs probabilities of binary decisions for selecting or discarding each source domain sample before aligning source and target distributions using an adversarial discriminator (Ganin et al. 2016). As these decision functions are discrete and non-differentiable, we rely on Gumbel Softmax sampling (Jang, Gu, and Poole 2016) to learn the policy jointly with network parameters through standard back-propagation, without resorting to complex reinforcement learning settings, as in (Chen et al. 2019b, 2020). Second, we develop an efficient self-labeling strategy that iteratively trains the classifier using both labeled source domain data and generated soft pseudo-labels for target domain to enhance the discriminabilty of the latent space. Finally, the mix module utilizes both intra-domain and inter-domain mixup regularization (Zhang et al. 2017) to generate convex combinations of pairs of training samples and their corresponding labels in both domains. The mix strategy not only helps to explore more intrinsic structures across domains leading to an invariant latent space, but also helps to stabilize the domain discriminator while bridging distribution shift across domains.

Our proposed modules are simple yet very effective which explore three unique aspects for the first time in partial domain adaptation setting in an end-to-end manner. Specifically, in each mini-batch, our framework simultaneously eliminates negative transfer by removing outlier source sam-

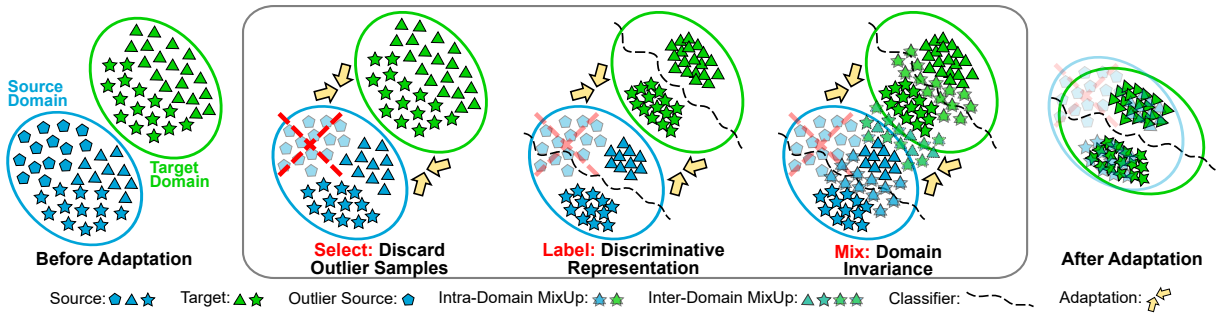

Figure 1: **A conceptual overview of our approach**. Our proposed approach adopts three unique modules namely Select, Label and Mix in a unified framework to mitigate domain shift and generalize the model to an unlabelled target domain with a label space which is a subset of that of the labelled source domain. Our Select module discards outlier samples from the source domain to eliminate negative transfer of untransferable knowledge. On the other hand, Label and Mix modules ensure discriminability and invariance of the latent space respectively while adapting the source classifier to the target domain in partial domain adaptation setting. Best viewed in color.

ples and learns discriminative invariant features by labeling and mixing samples. Our key contributions include:

- We propose a novel Select, Label, and Mix (**SLM**) framework for learning discriminative and invariant feature representation while preventing intrinsic negative transfer in partial domain adaptation.
- We develop a simple and efficient source sample selection strategy where the selector network is jointly trained with the domain adaptation model using backpropagation through Gumbel Softmax sampling.
- We conduct extensive experiments on four benchmark datasets, including Office31 (Saenko et al. 2010), Office-Home (Venkateswara et al. 2017), ImageNet-Caltech, and VisDA-2017 (Peng et al. 2017) to demonstrate the superiority of our approach over state-of-the-art methods.

Related Work

Unsupervised Domain Adaptation. Various strategies have been developed for unsupervised domain adaptation, including methods for reducing cross-domain divergence (Gretton et al. 2012; Long et al. 2015; Shen et al. 2017; Sun and Saenko 2016), adding domain discriminators for adversarial training (Chen et al. 2019a; Ganin and Lempitsky 2015; Ganin et al. 2016; Long et al. 2015, 2018, 2016; Pei et al. 2018; Tzeng et al. 2017), and image-to-image translation techniques (Hoffman et al. 2018; Hu et al. 2018; Murez et al. 2018) (see reviews (Csurka 2017; Wang and Deng 2018)). Despite remarkable progress, UDA methods assume that label spaces across source and target domains are identical unlike the practical problem we consider in this work.

Partial Domain Adaptation. Representative PDA methods train domain discriminators (Cao et al. 2018a,b; Zhang et al. 2018) with weighting, or use residual correction blocks (Li et al. 2020; Liang et al. 2020), or use source examples based on their similarities to target domain (Cao et al. 2019). Most relevant to our approach is the work in (Chen et al. 2019b, 2020) which uses Reinforcement Learning (RL) for source data selection in partial domain adaptation. RL policy gradients are often complex, unwieldy to train and require techniques to reduce variance during training. By contrast, our approach utilizes a gradient based optimization for relevant source sample selection which is extremely fast and computationally efficient. Moreover, while prior PDA methods try to reweigh source samples in some form or other, they often do not take class-aware information in target domain into consideration. Our approach instead, ensures discriminability and invariance of the latent space by considering both pseudo-labeling and cross-domain mixup with sample selection in an unified framework for PDA.

Self-Training with Pseudo-Labels. Deep self-training methods that focus on iteratively training the model by using both labeled source data and generated target pseudo-labels have been proposed for aligning both domains (Inoue et al. 2018; Mei et al. 2020; Saito, Ushiku, and Harada 2017; Zhang et al. 2020). Majority of the methods directly choose hard pseudo-labels with high prediction confidence. The works in (Zou et al. 2019, 2018) use class-balanced confidence regularizers to generate soft pseudo-labels for unsupervised domain adaptation that share same label space across domains. Our work on the other hand iteratively utilizes soft pseudo-labels within a batch by smoothing one-hot pseudo-label to a conservative target distribution for PDA.

Mixup Regularization. Mixup regularization (Zhang et al. 2017) that train models on virtual examples constructed as convex combinations of pairs of inputs and labels are recently used to improve the generalization of neural networks. A few recent methods apply Mixup, but mainly for UDA to stabilize domain discriminator (Wu, Inkpen, and El-Roby 2020; Xu et al. 2019a; Yan et al. 2020) or to smoothen the predictions (Mao et al. 2019). Our proposed SLM strategy can be regarded as an extension of this line of research by introducing both intra-domain and inter-domain mixup not only to stabilize the discriminator but also to guide the classifier in enriching the intrinsic structure of the latent space to solve the more challenging PDA task.

Proposed Method

Partial domain adaptation aims to mitigate the domain shift and generalize the model to an unlabelled target domain with a label space which is a subset of that of the labelled source domain. Formally, we define the set of labelled source domain samples as $\mathcal{D}_{source} = \{(\mathbf{x}_i^s, y_i)\}_{i=1}^{N_s}$ and unlabelled tar-

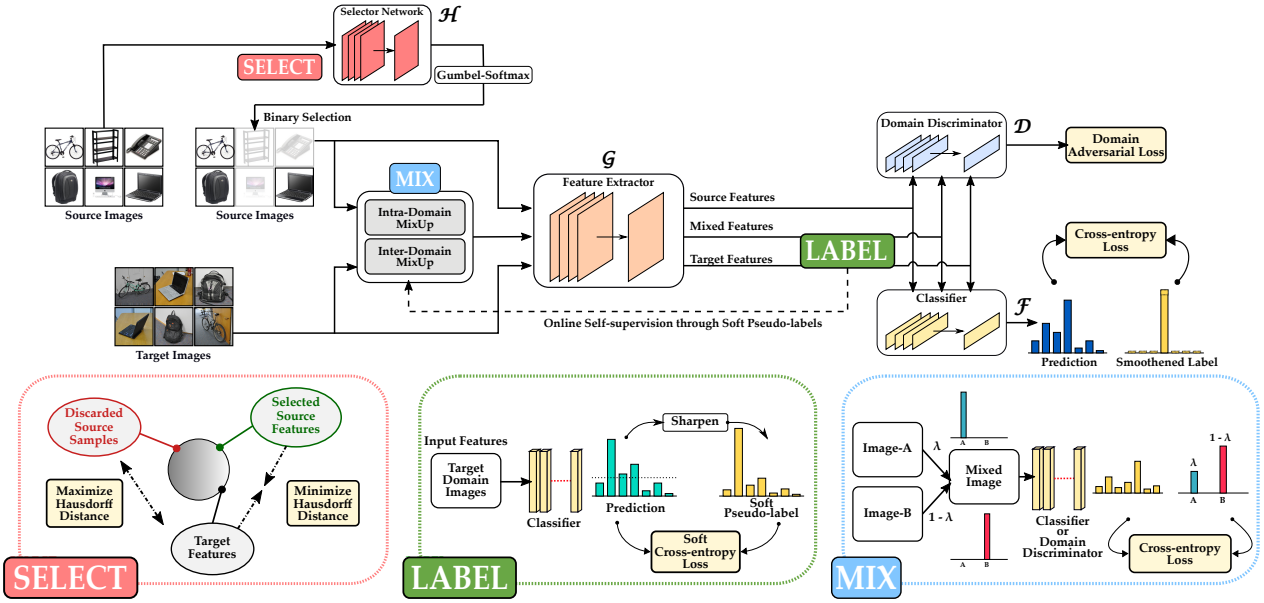

Figure 2: **Illustration of our proposed framework**. Our framework consists of a feature extractor $\mathcal{G}$ which maps the images to a common latent feature space, a classifier network $\mathcal{F}$ to provide class-wise predictions, a domain discriminator $\mathcal{D}$ to reduce domain discrepancy, and a selector network $\mathcal{H}$ for discarding outlier source samples ("Select") to mitigate the problem of negative transfer in partial domain adaptation. Our approach also comprises of two additional modules namely "Label" and "Mix" that works in conjunction with the "Select" module to ensure the discriminability and domain invariance of the latent space. Given a mini-batch of source and target domain images, all the components are optimized jointly in an iterative manner. See Section for more details. Best viewed in color.

get domain samples as $\mathcal{D}_{target} = \{\mathbf{x}_i^t\}_{i=1}^{N_T}$, with label spaces $\mathcal{L}_{source}$ and $\mathcal{L}_{target}$, respectively, where $\mathcal{L}_{source} \subsetneq \mathcal{L}_{target}$. N_S and N_T represent the number of samples in source and target domain respectively. Let p and q represent the probability distribution of data in source and target domain respectively. In PDA, we further have $p \neq q$ and $p_{\mathcal{L}_{target}} \neq q$, where $p_{\mathcal{L}_{target}}$ is the distribution of source domain data in $\mathcal{L}_{target}$. Our goal is to develop an approach with the above given data to improve the performance of a model on $\mathcal{D}_{target}$.

Approach Overview. Figure 2 illustrates an overview of our approach. Our framework consists of a feature extractor $\mathcal{G}$, a classifier network $\mathcal{F}$, a domain discriminator $\mathcal{D}$ and a selector network $\mathcal{H}$. Our goal is to improve classification performance of the combined network $\mathcal{F}(\mathcal{G}(.))$ on $\mathcal{D}_{target}$. While the feature extractor $\mathcal{G}$ maps the images to a common latent space, the task of classifier $\mathcal{F}$ is to output a probability distribution over the classes for a given feature from $\mathcal{G}$. Given a feature from $\mathcal{G}$, the discriminator $\mathcal{D}$ helps in minimizing domain discrepancy by identifying the domain (either source or target) to which it belongs. The selector network $\mathcal{H}$ helps in reducing negative transfer by learning to identify outlier source samples from $\mathcal{D}_{source}$ using Gumbel-Softmax sampling (Jang, Gu, and Poole 2016). On the other hand, label module utilizes predictions of $\mathcal{F}(\mathcal{G}(.))$ to obtain soft pseudo-labels for target samples. Finally, the mix module leverages both pseudo-labeled target samples and source samples to generate augmented images for achieving domain invariance in the latent space. During training, for a mini-batch of images, all the components are trained jointly and during testing, we evaluate performance using classification accuracy of the network $\mathcal{F}(\mathcal{G}(.))$ on target domain data $\mathcal{D}_{target}$. The individual modules are discussed below.

Select Module. This module stands in the core of our framework with an aim to get rid of the outlier source samples in order to minimize negative transfer. Instead of using different heuristically designed criteria for weighting source samples, we develop a novel selector network $\mathcal{H}$, that takes images from the source domain as input and makes instance-level binary predictions to obtain *relevant* source samples for adaptation, as shown in Figure 3. Specifically, the selector network $\mathcal{H}$ provides a discrete binary output of either a 0 (*discard*) or 1 (*select*) for each source sample, *i.e.*, $\mathcal{H} : \mathcal{D}_{source} \rightarrow \{0, 1\}$. We leverage Gumbel-Softmax operation to design the learning protocol of the selector network, as described next. Given the selection, we forward only the selected samples to the successive modules.

Training using Gumbel-Softmax Sampling. Our select module makes decisions about whether a source sample belongs to an outlier class or not. However, the fact that the decision policy is discrete makes the network non-differentiable and therefore difficult to optimize via standard backpropagation. To resolve the non-differentiability and enable gradient descent optimization for the selector in an efficient way, we adopt Gumbel-Softmax trick (Jang, Gu, and Poole 2016; Maddison, Mnih, and Teh 2016) and draw samples from a categorical distribution parameterized by α_0, α_1, where α_0, α_1 are the output logits of the selector network for a sample to be selected and discarded respectively. Specifically, as illustrated in Figure 3, the selector network $\mathcal{H}$ takes a batch of source

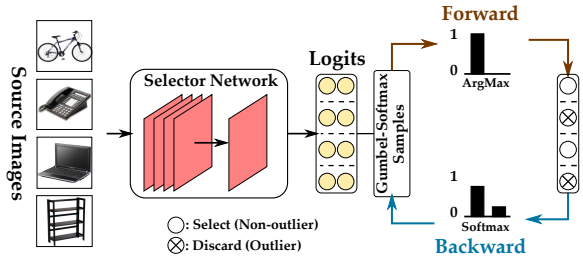

Figure 3: **Learning with Gumbel Softmax Sampling.** The figure illustrates the Gumbel-Softmax trick used for enabling gradient-based optimization for discrete output space. Best viewed in color.

images (say, $\mathcal{B}_s$ of size b) as input, and outputs a two-dimensional matrix $\beta \in \mathbb{R}^{b\times 2}$, where each row corresponds to $[\alpha_0, \alpha_1]$ for an image. We then draw i.i.d. samples G_0, G_1 from $Gumbel(0,1) = -log(-log(U))$, where $U \sim$ Uniform$[0,1]$ and generate discrete samples in the forward pass as: $X = \arg\max_i[\log \alpha_i + G_i]$ resulting in hard binary predictions, while during backward pass, we approximate gradients using continuous softmax relaxation as:

$$\mathcal{Y}_i = \frac{\exp((\log \alpha_i + G_i)/\tau)}{\sum_{j\in\{0,1\}} \exp((\log \alpha_j + G_j)/\tau)} \quad \text{for } i \in \{0,1\} \quad (1)$$

where G_i's are i.i.d samples from standard Gumbel distribution $Gumbel(0,1)$ and τ denotes temperature of softmax. Clearly, when $\tau > 0$, the Gumbel-Softmax distribution is smooth and hence gradients can be computed with respect to logits α_i's to train the selector network using backpropagation. As τ approaches 0, $\mathcal{Y}_i$ becomes one-hot and discrete.

Learning to Discard the Outlier Distribution. With the unsupervised nature of this decision problem, the design of the loss function for the selector network is challenging. We propose a novel Hausdorff distance-based triplet loss function for the select module which ensures that the selector network learns to distinguish between the outlier and the non-outlier distribution in the source domain. For a given batch of source domain images $\mathcal{D}^b_{source}$ and target domain images $\mathcal{D}^b_{target}$, each of size b, the selector results in two subsets of source samples $\mathcal{D}^b_{sel} = \{x \in \mathcal{D}^b_{source} : \mathcal{H}(x) = 1\}$ and $\mathcal{D}^b_{dis} = \{x \in \mathcal{D}^b_{source} : \mathcal{H}(x) = 0\}$. The idea is to pull the *selected* source samples $\mathcal{D}^b_{sel}$ & target samples $\mathcal{D}^b_{target}$ closer while pushing *discarded* source samples $\mathcal{D}^b_{dis}$ & $\mathcal{D}^b_{target}$ apart in the output latent feature space of $\mathcal{G}$. To achieve this, we formulate the loss function as follows:

$$\begin{aligned} d_{sel} &= d_H(\mathcal{G}(\mathcal{D}^b_{sel}), \mathcal{G}(\mathcal{D}^b_{target})) \\ d_{dis} &= d_H(\mathcal{G}(\mathcal{D}^b_{dis}), \mathcal{G}(\mathcal{D}^b_{target})) \\ \mathcal{L}_{select} &= \lambda_s max(d_{sel} - d_{dis} + margin, 0) + \mathcal{L}_{reg} \end{aligned} \quad (2)$$

where $d_H(X,Y)$ represents the average Hausdorff distance between the set of features X and Y. $\mathcal{L}_{reg} = \lambda_{reg1} \sum_{x\in\mathcal{D}^b_{source}} \mathcal{H}(x)\log(\mathcal{H}(x)) + \lambda_{reg2}\{\sum_{\hat{p}} l_{ent}(\hat{p}) - l_{ent}(\hat{p}_m)\}$, with l_{ent} being the entropy loss, $\hat{p}$ is the Softmax prediction of $\mathcal{F}(\mathcal{G}(\mathcal{D}_{target}))$ and $\hat{p}_m$ is mean prediction for the target domain. $\mathcal{L}_{reg}$ is a regularization to restrict $\mathcal{H}$ from producing trivial all-0 or all-1 outputs as well as ensuring confident and diverse predictions by $\mathcal{F}(\mathcal{G}(.))$ for $\mathcal{D}_{target}$. Note that only $\mathcal{D}^b_{sel}$ is used to train the classifier, domain discriminator and is utilised by other modules to perform subsequent operations. Furthermore, to avoid any interference from the backbone feature extractor $\mathcal{G}$, we use a separate feature extractor for the select module, while making these decisions. In summary, the supervision signal for the selector module comes from (a) the discriminator directly, (b) through interactions with other modules via joint learning, and (c) the triplet loss using Hausdorff distance.

Label Module. While our select module helps in removing source domain outliers, it fails to guarantee the discriminability of the latent space due to the absence of class-aware information in the target domain. Specifically, given our main objective is to improve the classification performance on target domain samples, it becomes essential for the classifier to learn confident decision boundaries in the target domain. To this end, we propose a label module that provides additional self-supervision for target domain samples. Motivated by the effectiveness of confidence guided self-training (Zou et al. 2019), we generate soft pseudo-labels for the target domain samples that efficiently attenuates the unwanted deviations caused by false and noisy one-hot pseudo-labels. For a target domain sample $\mathbf{x}^t_k \in \mathcal{D}_{target}$, the soft-pseudo-label $\hat{y}^t_k$ is computed as follows:

$$\hat{y}^{t(i)}_k = \frac{p(i|\mathbf{x}^t_i)^{\frac{1}{\alpha}}}{\sum_{j=1}^{|\mathcal{L}_{source}|} p(j|\mathbf{x}^t_i)^{\frac{1}{\alpha}}} \quad (3)$$

where $p(j|\mathbf{x}^t_i)$ is the softmax probability of the classifier for class j given $\mathbf{x}^t_i$ as input, and α is a hyper-parameter that controls the softness of the label. The soft pseudo-label $\hat{y}^t_i$ is then used to compute the loss $\mathcal{L}_{label}$ for a given batch of target samples $\mathcal{D}^b_{target}$ as follows:

$$\mathcal{L}_{label} = \mathbb{E}_{\mathbf{x}^t_i \in \mathcal{D}^b_{target}} l_{ce}(\mathcal{F}(\mathcal{G}(\mathbf{x}^t_i)), \hat{y}^t_i) \quad (4)$$

where $l_{ce}(.)$ represents the cross-entropy loss.

Mix Module. Learning a domain-invariant latent space is crucial for effective adaptation of a classifier from source to target domain. However, with limited samples per batch and after discarding the outlier samples, it becomes even more challenging in preventing over-fitting and learning domain invariant representation. To mitigate this problem, we apply MixUp (Zhang et al. 2017) on the selected source samples and the target samples for discovering ingrained structures in establishing domain invariance. Given $\mathcal{D}^b_{sel}$ from select module and $\mathcal{D}^b_{target}$ with corresponding labels $\hat{y}^t$ from label module, we perform convex combinations of images belonging to these two sets on pixel-level in three different ways namely, *inter-domain*, *intra-source* domain and *intra-target* domain to obtain the following sets of augmented data:

$$\begin{aligned} \mathcal{D}^b_{inter\_mix} &= \{(\lambda \mathbf{x}^s_i + (1-\lambda)\mathbf{x}^t_j, \lambda y_i + (1-\lambda)\hat{y}^t_j)\} \\ \mathcal{D}^b_{intra\_mix\_s} &= \{(\lambda \mathbf{x}^s_i + (1-\lambda)\mathbf{x}^s_j, \lambda y_i + (1-\lambda) y_j)\} \\ \mathcal{D}^b_{intra\_mix\_t} &= \{(\lambda \mathbf{x}^t_i + (1-\lambda)\mathbf{x}^t_j, \lambda \hat{y}^t_i + (1-\lambda)\hat{y}^t_j)\} \\ \mathcal{D}^b_{mix} &= \mathcal{D}^b_{inter\_mix} \cup \mathcal{D}^b_{intra\_mix\_s} \cup \mathcal{D}^b_{intra\_mix\_t} \end{aligned} \quad (5)$$

where $(\mathbf{x}^s_{i/j}, y_{i/j}) \in \mathcal{D}^b_{sel}$, while $\mathbf{x}^t_{i/j} \in \mathcal{D}^b_{target}$ with $\hat{y}^t_{i/j}$ being the corresponding soft-pseudo-labels. λ is the mix-ratio randomly sampled from a beta distribution $Beta(\alpha, \alpha)$ for $\alpha \in (0, \infty)$. We use $\alpha = 2.0$ in all our experiments. We utilize the new augmented images in training both the classifier $\mathcal{F}$ and the domain discriminator $\mathcal{D}$ as follows:

$$\begin{aligned}
\mathcal{L}_{mix_cls} &= \mathbb{E}_{(\mathbf{x}_i, y_i) \in \mathcal{D}^b_{mix}} l_{ce}(\mathcal{F}(\mathcal{G}(\mathbf{x}_i)), y_i) \\
\mathcal{L}_{mix_dom} &= \mathbb{E}_{\mathbf{x}_i \sim \mathcal{D}^b_{inter_mix}} [\lambda \log(\mathcal{D}(\mathcal{G}(\mathbf{x}_i))) \\
&\quad + (1-\lambda) \log(1-\mathcal{D}(\mathcal{G}(\mathbf{x}_i)))] \\
&\quad + \mathbb{E}_{\mathbf{x}_i \sim \mathcal{D}^b_{intra_mix_s}} \log(\mathcal{D}(\mathcal{G}(\mathbf{x}_i))) \\
&\quad + \mathbb{E}_{\mathbf{x}_i \sim \mathcal{D}^b_{intra_mix_t}} \log(1-\mathcal{D}(\mathcal{G}(\mathbf{x}_i))) \\
\mathcal{L}_{mix} &= \mathcal{L}_{mix_cls} + \mathcal{L}_{mix_dom} \qquad (6)
\end{aligned}$$

where $\mathcal{L}_{mix\_cls}$ and $\mathcal{L}_{mix\_dom}$ represent loss for classifier and domain discriminator respectively. Our mix strategy with the combined loss $\mathcal{L}_{mix}$ not only helps to explore more intrinsic structures across domains leading to an invariant latent space, but also helps to stabilize the domain discriminator while bridging the distribution shift across domains.

Optimization Besides the above three modules that are tailored for PDA, we use standard supervised loss on the labeled source data and domain adversarial loss as follows:

$$\begin{aligned}
\mathcal{L}_{sup} &= \mathbb{E}_{(\mathbf{x}_i, y_i) \in \mathcal{D}^b_{sel}} l_{ce}(\mathcal{F}(\mathcal{G}(\mathbf{x}_i)), y_i) \\
\mathcal{L}_{adv} &= \mathbb{E}_{\mathbf{x}^s \sim \mathcal{D}^b_{sel}} w^s \log(\mathcal{D}(\mathcal{G}(\mathbf{x}^s))) \\
&\quad + \mathbb{E}_{\mathbf{x}^t \sim \mathcal{D}^b_{target}} w^t \log(1-\mathcal{D}(\mathcal{G}(\mathbf{x}^t))) \qquad (7)
\end{aligned}$$

where $\mathcal{L}_{adv}$ is entropy-conditioned domain adversarial loss with weights w^s and w^t for source and target domain respectively (Long et al. 2018). The overall loss $\mathcal{L}_{total}$ is

$$\mathcal{L}_{total} = \mathcal{L}_{sup} + \mathcal{L}_{adv} + \mathcal{L}_{select} + \mathcal{L}_{label} + \mathcal{L}_{mix} \qquad (8)$$

where $\mathcal{L}_{select}$, $\mathcal{L}_{label}$, and $\mathcal{L}_{mix}$ are given by Equations (2), (4), and (6) respectively. We integrate all the modules into one framework, as shown in the Figure 2 and train the network jointly for partial domain adaptation.

Experiments

Datasets. We evaluate our approach using four datasets, namely Office31 (Saenko et al. 2010), Office-Home (Venkateswara et al. 2017), ImageNet-Caltech and VisDA-2017 (Peng et al. 2017). Office31 contains 4,110 images of 31 classes from three distinct domains. Following (Chen et al. 2020), we select 10 classes shared by Office31 and Caltech256 (Griffin, Holub, and Perona 2007) as target categories. Office-Home is a challenging dataset that contains images from four domains. We follow (Chen et al. 2020) to select the first 25 categories (in alphabetic order) in each domain as target classes. ImageNet-Caltech is a challenging dataset that consists of two subsets, ImageNet1K (I) (Russakovsky et al. 2015) and Caltech256 (C) (Griffin, Holub, and Perona 2007). While source domain contains 1,000 and 256 classes for ImageNet and Caltech respectively, each target domain contains only 84 classes that are common across both domains. VisDA-2017 is a large-scale challenging dataset with 12 categories across 2 domains: photo-realistic images or real images (R), and synthetic 2D renderings of 3D models (S). We select the first 6 categories (in alphabetical order) in each of the domain as the target categories(Li et al. 2020). More details about the datasets are included in the supplementary material.

Office31							
Method	**A → W**	**D → W**	**W → D**	**A → D**	**D → A**	**W → A**	**Average**
ResNet-50	$76.5_{\pm 0.3}$	$99.2_{\pm 0.2}$	$97.7_{\pm 0.1}$	$87.5_{\pm 0.2}$	$87.2_{\pm 0.1}$	$84.1_{\pm 0.3}$	88.7
DANN	$62.8_{\pm 0.6}$	$71.6_{\pm 0.4}$	$65.6_{\pm 0.5}$	$65.1_{\pm 0.7}$	$78.9_{\pm 0.3}$	$79.2_{\pm 0.4}$	70.5
CORAL	$52.1_{\pm 0.5}$	$65.2_{\pm 0.2}$	$64.1_{\pm 0.7}$	$58.0_{\pm 0.5}$	$73.1_{\pm 0.4}$	$77.9_{\pm 0.3}$	65.1
ADDA	$75.7_{\pm 0.2}$	$95.4_{\pm 0.2}$	$\underline{99.9}_{\pm 0.1}$	$83.4_{\pm 0.2}$	$83.6_{\pm 0.1}$	$84.3_{\pm 0.1}$	87.0
RTN	75.3	97.1	98.3	66.9	85.6	85.7	84.8
CDAN+E	$80.5_{\pm 1.2}$	$99.0_{\pm 0.0}$	$98.1_{\pm 0.0}$	$77.1_{\pm 0.9}$	$93.6_{\pm 0.1}$	$91.7_{\pm 0.0}$	90.0
JDDA	$73.5_{\pm 0.6}$	$93.1_{\pm 0.3}$	$89.3_{\pm 0.2}$	$76.4_{\pm 0.4}$	$77.6_{\pm 0.1}$	$82.8_{\pm 0.2}$	82.1
CAN	$84.4_{\pm 0.0}$	$92.0_{\pm 1.4}$	$94.7_{\pm 1.7}$	$84.9_{\pm 0.9}$	$85.6_{\pm 1.0}$	$86.4_{\pm 0.8}$	88.0
PADA	$86.3_{\pm 0.4}$	$\underline{99.3}_{\pm 0.1}$	$\mathbf{100.0}_{\pm 0.0}$	$90.4_{\pm 0.1}$	$91.3_{\pm 0.2}$	$92.6_{\pm 0.1}$	93.3
SAN	$93.9_{\pm 0.5}$	$\underline{99.3}_{\pm 0.5}$	$99.4_{\pm 0.1}$	$94.3_{\pm 0.3}$	$94.2_{\pm 0.4}$	$88.7_{\pm 0.4}$	95.0
IWAN	$89.2_{\pm 0.4}$	$\underline{99.3}_{\pm 0.3}$	$99.4_{\pm 0.2}$	$90.5_{\pm 0.4}$	$\underline{95.6}_{\pm 0.3}$	$94.3_{\pm 0.3}$	94.7
ETN	$93.4_{\pm 0.3}$	$\underline{99.3}_{\pm 0.1}$	$99.2_{\pm 0.2}$	$95.5_{\pm 0.4}$	$95.4_{\pm 0.1}$	$91.7_{\pm 0.2}$	95.8
DRCN	88.1	**100.0**	**100.0**	86.0	$\underline{95.6}$	$\underline{95.8}$	94.3
RTNet	$95.1_{\pm 0.3}$	$\mathbf{100.0}_{\pm 0.0}$	$\mathbf{100.0}_{\pm 0.0}$	$97.8_{\pm 0.1}$	$93.9_{\pm 0.1}$	$94.1_{\pm 0.1}$	96.8
$\text{RTNet}_{\text{adv}}$	$96.2_{\pm 0.3}$	$\mathbf{100.0}_{\pm 0.0}$	$\mathbf{100.0}_{\pm 0.0}$	$97.6_{\pm 0.1}$	$92.3_{\pm 0.1}$	$95.4_{\pm 0.1}$	96.9
BA^3US	$\underline{99.0}_{\pm 0.3}$	$\mathbf{100.0}_{\pm 0.0}$	$98.7_{\pm 0.0}$	$\mathbf{99.4}_{\pm 0.0}$	$94.8_{\pm 0.1}$	$95.0_{\pm 0.1}$	$\underline{97.8}$
SLM (Ours)	$\mathbf{99.8}_{\pm 0.2}$	$\mathbf{100.0}_{\pm 0.0}$	$99.8_{\pm 0.3}$	$\underline{98.7}_{\pm 0.0}$	$\mathbf{96.1}_{\pm 0.1}$	$\mathbf{95.9}_{\pm 0.0}$	**98.4**

Table 1: **Performance on Office31.** Numbers show the accuracy (%) of different methods on partial domain adaptation setting. We highlight the **best** and $\underline{\text{second best}}$ method on each transfer task. While the upper section shows the results of some popular unsupervised domain adaptation approaches, the lower section shows results of existing partial domain adaptation methods.

Baselines. We compare our approach with several methods that fall into two main categories: (1) popular UDA methods (e.g., DANN (Ganin et al. 2016), CORAL (Sun and Saenko 2016)) including recent methods like CAN (Kang et al. 2019) and SPL (Wang and Breckon 2020) which have shown state-of-the-art performance on UDA setup, (2) existing partial domain adaptation methods including PADA (Cao et al. 2018b), SAN (Cao et al. 2018a), ETN (Cao et al. 2019), and DRCN (Li et al. 2020). We also compare with the recent state-of-the-art methods, RTNet (Chen et al. 2020) that uses reinforcement learning for source dataset selection, and BA^3US (Liang et al. 2020) which uses source samples to augment the target domain in partial domain adaptation. We directly quote the numbers reported in published papers (Chen et al. 2020; Li et al. 2020; Liang et al. 2020) and use the same backbone network in our approach to make a fair comparison with different baselines.

Implementation Details. We use ResNet-50 (He et al. 2016) as the backbone network for the feature extractor while we use ResNet-18 for the selector network, initialized with ImageNet (Russakovsky et al. 2015) pretrained weights. In Eqn. 2 we set $\lambda_s, \lambda_{reg1}$ and λ_{reg2} as $0.01, 10.0$ and 0.1, respectively. We use a margin value of 100.0 in all our experiments. We use gradient reversal layer (GRL) for adversarially training the discriminator. We set $\tau = 1.0$ in Eqn. 1, $\alpha = 0.1$ in Eqn. 3, and $\lambda = 0.0$ for the GRL as initial values and gradually anneal τ and α down to 0 while increase λ to 1.0 during the training, as in (Jang, Gu, and Poole 2016). Additionally, we use label-smoothing for all the losses for the feature extractor involving source domain images as in (Liang, Hu, and Feng 2020; Müller, Kornblith, and Hinton 2019), with $\epsilon = 0.2$. We use SGD for optimiza-

Office-Home													
Method	**Ar → Cl**	**Ar → Pr**	**Ar → Rw**	**Cl → Ar**	**Cl → Pr**	**Cl → Rw**	**Pr → Ar**	**Pr → Cl**	**Pr → Rw**	**Rw → Ar**	**Rw → Cl**	**Rw → Pr**	**Average**
ResNet-50	$47.2_{\pm 0.2}$	$66.8_{\pm 0.3}$	$76.9_{\pm 0.5}$	$57.6_{\pm 0.2}$	$58.4_{\pm 0.1}$	$62.5_{\pm 0.3}$	$59.4_{\pm 0.3}$	$40.6_{\pm 0.2}$	$75.9_{\pm 0.3}$	$65.6_{\pm 0.1}$	$49.1_{\pm 0.2}$	$75.8_{\pm 0.4}$	61.3
DANN	$43.2_{\pm 0.5}$	$61.9_{\pm 0.2}$	$72.1_{\pm 0.4}$	$52.3_{\pm 0.4}$	$53.5_{\pm 0.2}$	$57.9_{\pm 0.1}$	$47.2_{\pm 0.3}$	$35.4_{\pm 0.1}$	$70.1_{\pm 0.3}$	$61.3_{\pm 0.2}$	$37.0_{\pm 0.2}$	$71.7_{\pm 0.3}$	55.3
CORAL	$38.2_{\pm 0.1}$	$55.6_{\pm 0.3}$	$65.9_{\pm 0.2}$	$48.4_{\pm 0.4}$	$52.5_{\pm 0.1}$	$51.3_{\pm 0.2}$	$48.9_{\pm 0.3}$	$32.6_{\pm 0.1}$	$67.1_{\pm 0.2}$	$63.8_{\pm 0.4}$	$35.9_{\pm 0.2}$	$69.8_{\pm 0.1}$	52.5
ADDA	45.2	68.8	79.2	64.6	60.0	68.3	57.6	38.9	77.5	70.3	45.2	78.3	62.8
RTN	49.4	64.3	76.2	47.6	51.7	57.7	50.4	41.5	75.5	70.2	51.8	74.8	59.3
CDAN+E	47.5	65.9	75.7	57.1	54.1	63.4	59.6	44.3	72.4	66.0	49.9	72.8	60.7
JDDA	$45.8_{\pm 0.4}$	$63.9_{\pm 0.2}$	$74.1_{\pm 0.3}$	$51.8_{\pm 0.2}$	$55.2_{\pm 0.3}$	$60.3_{\pm 0.2}$	$53.7_{\pm 0.2}$	$38.3_{\pm 0.1}$	$72.6_{\pm 0.2}$	$62.5_{\pm 0.1}$	$43.3_{\pm 0.3}$	$71.3_{\pm 0.1}$	57.7
SPL	$46.4_{\pm 0.0}$	$70.5_{\pm 0.6}$	$77.2_{\pm 0.0}$	$61.0_{\pm 0.0}$	$65.2_{\pm 0.0}$	$73.2_{\pm 0.0}$	$64.3_{\pm 0.0}$	$44.7_{\pm 0.0}$	$79.1_{\pm 0.0}$	$69.5_{\pm 0.0}$	$58.0_{\pm 0.0}$	$79.8_{\pm 0.0}$	65.7
PADA	$53.2_{\pm 0.2}$	$69.5_{\pm 0.1}$	$78.6_{\pm 0.1}$	$61.7_{\pm 0.2}$	$62.7_{\pm 0.3}$	$60.9_{\pm 0.1}$	$56.4_{\pm 0.5}$	$44.6_{\pm 0.2}$	$79.3_{\pm 0.1}$	$74.2_{\pm 0.1}$	$55.1_{\pm 0.3}$	$77.4_{\pm 0.2}$	64.5
SAN	44.4	68.7	74.6	67.5	65.0	77.8	59.8	44.7	80.1	72.2	50.2	78.7	65.3
IWAN	53.9	54.5	78.1	61.3	48.0	63.3	54.2	52.0	81.3	76.5	56.8	82.9	63.6
ETN	$60.4_{\pm 0.3}$	$76.5_{\pm 0.2}$	$77.2_{\pm 0.3}$	$64.3_{\pm 0.1}$	$67.5_{\pm 0.3}$	$75.8_{\pm 0.2}$	$69.3_{\pm 0.1}$	$54.2_{\pm 0.1}$	$83.7_{\pm 0.2}$	$75.6_{\pm 0.3}$	$56.7_{\pm 0.2}$	$84.5_{\pm 0.3}$	70.5
SAFN	$58.9_{\pm 0.5}$	$76.3_{\pm 0.3}$	$81.4_{\pm 0.3}$	$70.4_{\pm 0.5}$	$\underline{73.0}_{\pm 1.4}$	$77.8_{\pm 0.5}$	$72.4_{\pm 0.3}$	$55.3_{\pm 0.5}$	$80.4_{\pm 0.8}$	$75.8_{\pm 0.4}$	$\underline{60.4}_{\pm 0.8}$	$79.9_{\pm 0.2}$	71.8
DRCN	54.0	76.4	83.0	62.1	64.5	71.0	70.8	49.8	80.5	77.5	59.1	79.9	69.0
RTNet	$\underline{62.7}_{\pm 0.1}$	$79.3_{\pm 0.2}$	$81.2_{\pm 0.1}$	$65.1_{\pm 0.1}$	$68.4_{\pm 0.3}$	$76.5_{\pm 0.1}$	$70.8_{\pm 0.2}$	$55.3_{\pm 0.1}$	$85.2_{\pm 0.3}$	$76.9_{\pm 0.2}$	$59.1_{\pm 0.2}$	$83.4_{\pm 0.3}$	72.0
$\text{RTNet}_{\text{adv}}$	$\mathbf{63.2}_{\pm 0.1}$	$80.1_{\pm 0.2}$	$80.7_{\pm 0.1}$	$66.7_{\pm 0.1}$	$69.3_{\pm 0.2}$	$77.2_{\pm 0.2}$	$71.6_{\pm 0.3}$	$53.9_{\pm 0.3}$	$84.6_{\pm 0.1}$	$77.4_{\pm 0.2}$	$57.9_{\pm 0.3}$	$\underline{85.5}_{\pm 0.1}$	$\underline{72.3}$
BA^3US	$60.6_{\pm 0.5}$	$\underline{83.2}_{\pm 0.1}$	$\underline{88.4}_{\pm 0.2}$	$\underline{71.8}_{\pm 0.2}$	$72.8_{\pm 1.1}$	$\mathbf{83.4}_{\pm 0.6}$	$\mathbf{75.5}_{\pm 0.2}$	$\mathbf{61.6}_{\pm 0.4}$	$\underline{86.5}_{\pm 0.2}$	$\underline{79.3}_{\pm 0.7}$	$\mathbf{62.8}_{\pm 0.5}$	$\mathbf{86.1}_{\pm 0.3}$	**76.0**
SLM (Ours)	$61.1_{\pm 0.7}$	$\mathbf{84.0}_{\pm 0.7}$	$\mathbf{91.4}_{\pm 0.3}$	$\mathbf{76.5}_{\pm 0.1}$	$\mathbf{75.0}_{\pm 1.2}$	$\underline{81.8}_{\pm 1.2}$	$\underline{74.6}_{\pm 0.7}$	$\underline{55.6}_{\pm 0.7}$	$\mathbf{87.8}_{\pm 0.8}$	$\mathbf{82.3}_{\pm 0.5}$	$57.8_{\pm 0.8}$	$83.5_{\pm 0.6}$	**76.0**

Table 2: **Performance on Office-Home.** We highlight the **best** and second best method on each task.

tion with momentum=0.9 while a weight decay of 1e-3 and 5e-4 for the selector network and the other networks respectively. We use an initial learning rate of 5e-3 for the selector and the classifier, while 5e-4 for the rest of the networks and decay it following a cosine annealing strategy. We use a batch size of 64 for Office31 and VisDA-2017 while a batch size of 128 is used for Office-Home and ImageNet-Caltech. We report average classification accuracy and standard deviation over 3 random trials. More implementation details and source codes can be found in supplementary material. We will make our source codes publicly available.

Results and Analysis. Table 1 shows the results of our method and other competing approaches on Office31 dataset. We have the following key observations. (1) As expected, the popular UDA methods including the recent CAN (Kang et al. 2019), fail to outperform the simple no adaptation model (ResNet-50), which implies that they suffer from negative transfer due to the presence of outlier source samples in partial domain adaptation. (2) Overall, our **SLM** framework outperforms all the existing PDA methods by achieving the best results on **4 out of 6** transfer tasks. Among PDA methods, BA^3US (Liang et al. 2020) is the most competitive. However, **SLM** still outperforms it (97.8% vs 98.4%) due to our two novel components working in concert with the removal of outliers: enhancing discriminability of the latent space via iterative pseudo-labeling of target domain samples and learning domain-invariance through mixup regularizations. (3) Our approach performed remarkably well on transfer tasks where the number of source domain images is very small compared to the target domain, e.g., on D→A, **SLM** outperforms BA^3US by $\mathbf{1.3\%}$.

On the challenging Office-Home dataset, our approach obtains the best performance on **6 out of 12** transfer tasks, with an average accuracy of 76.0% on this dataset (Table 2). Table 3 summarizes the results on ImageNet-Caltech and VisDA-2017 datasets. Our approach achieves new state-of-the-art result, outperforming the next competitive method by a margin of about $\mathbf{2.8\%}$ and $\mathbf{18.9\%}$ on ImageNet-Caltech and VisDA-2017 respectively. Especially for task S → R on VisDA-2017, our approach significantly outperforms SAFN (Xu et al. 2019b) and DRCN (Li et al. 2020) by an increase of $\mathbf{24.1\%}$ and $\mathbf{33.5\%}$ respectively. Note that on the most challenging VisDA-2017 dataset, our approach is still able to distill more positive knowledge from the synthetic to the real domain despite significant domain gap across them. In summary, our **SLM** framework outperforms the existing PDA methods on all four datasets, showing the effectiveness of our approach in not only identifying the most relevant source classes but also learning more transferable features for partial domain adaptation.

	ImageNet-Caltech			VisDA-2017		
Method	**I → C**	**C → I**	**Average**	**R → S**	**S → R**	**Average**
ResNet-50	$69.7_{\pm 0.8}$	$71.3_{\pm 0.7}$	70.5	64.3	45.3	54.8
DAN	71.6	66.5	69.0	68.4	47.6	58.0
DANN	68.7	52.9	60.8	73.8	51.0	62.4
ADDA	$71.8_{\pm 0.5}$	$69.3_{\pm 0.4}$	70.6	–	–	–
RTN	72.2	68.3	70.3	72.9	50.0	61.5
CDAN+E	$72.5_{\pm 0.1}$	$72.0_{\pm 0.1}$	72.2	–	–	–
PADA	$75.0_{\pm 0.4}$	$70.5_{\pm 0.4}$	72.8	$\underline{76.5}$	53.5	65.0
SAN	$77.8_{\pm 0.4}$	$75.3_{\pm 0.4}$	76.5	69.7	49.9	
IWAN	$78.1_{\pm 0.4}$	$73.3_{\pm 0.5}$	75.7	71.3	48.6	
ETN	$\mathbf{83.2}_{\pm 0.2}$	$74.9_{\pm 0.4}$	$\underline{79.1}$	–	–	–
SAFN	–	–	–	–	$\underline{67.7}_{\pm 0.5}$	–
DRCN	75.3	$\underline{78.9}$	77.1	73.2	58.2	$\underline{65.7}$
SLM (Ours)	$\underline{82.3}_{\pm 0.1}$	$\mathbf{81.4}_{\pm 0.6}$	**81.9**	$\mathbf{77.5}_{\pm 0.8}$	$\mathbf{91.7}_{\pm 0.8}$	**84.6**

Table 3: **Performance on ImageNet-Caltech and VisDA-2017.**

Ablation Studies. We perform the following experiments to test the effectiveness of the proposed modules including the effect of number of target classes on different datasets.

Effectiveness of Individual Modules. We conduct experiments to investigate the importance of our three unique

Modules			Dataset		
Select	Label	Mix	Office-31	Office-Home	VisDA-2017
✗	✗	✗	89.3	57.7	57.0
✓	✗	✗	94.9	65.8	68.7
✓	✓	✗	96.0	73.2	81.0
✓	✓	✓	98.4	76.0	84.6

Table 4: **Effectiveness of Different Modules.**

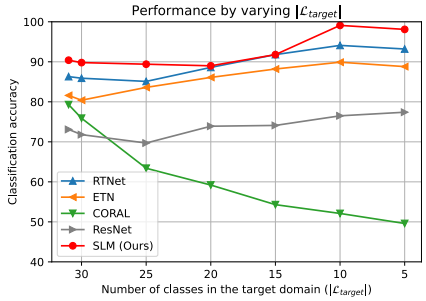

Figure 4: Performance by varying the number of target classes on A→W task from Office31 dataset. Best viewed in color.

modules on three datasets. E.g. On Office-Home, as seen from Table 9, while the Select only module improves the vanilla performance by 8%, addition of Label and Mix modules progressively improves the result to obtain the best performance of 76.0%. This corroborates the fact that both discriminability and invariance of the latent space plays a crucial role in partial domain adaptation in addition to the removal of source domain outlier samples.

Comparison with Varying Number of Target Classes. We compare different methods by varying the number of target classes. Figure 4 shows that our **SLM** framework consistently obtains the best results indicating its advantage in alleviating negative transfer by removing outlier source samples. Moreover, our approach outperforms all the compared methods even in the case of completely shared space (A31 → W31), which shows that it does not discard relevant samples incorrectly when there are no outlier classes.

Effectiveness of Hausdorff Distance. We investigate the effect of Hausdorff distance (Eqn. 2) in selector network training and find that removing it lowers down the performance from 76.0% to 73.7% on Office-Home dataset, showing its importance in guiding the selector network to discard the outlier source samples for reduction in negative transfer.

Distance between Domains. Following (Chen et al. 2020), we compute the Wasserstein distance between the probability distribution of the target samples (T) with that of the selected (S_{sel}) and discarded samples (S_{dis}) by the selector network. Table 5 shows that $\texttt{dist}(S_{sel}, T)$ is smaller than $\texttt{dist}(S_{all}, T)$, while $\texttt{dist}(S_{dis}, T)$ is greater than $\texttt{dist}(S_{all}, T)$ on two randomly sampled adaptation tasks from Office31 and Office-Home datasets. This results indicate that the samples selected by our selector network is closer to the target domain while the discarded samples are very dissimilar to the target domain.

Effectiveness of Soft Pseudo-Labels. We also test the effectiveness of soft pseudo-labels by replacing them with

Distance	A → D	W → A	Cl → Pr	Rw → Pr
$\texttt{dist}(S_{sel}, T)$	0.999	0.893	0.819	0.947
$\texttt{dist}(S_{dis}, T)$	1.013	1.144	1.418	1.008

Table 5: **Wasserstein Distance between Domains.** The values are normalized by assuming $\texttt{dist}(S_{all}, T)$ to be equal to 1.000, where S_{all} represents all source samples for the corresponding tasks.

Figure 5: **t-SNE Visualizations for A→W task from Office-31.** Blue and red dots represent source and target data respectively.

hard pseudo-labels for the target samples and observe that the average performance decreases from 76.0% to 72.0% on Office-Home dataset. This confirms that soft pseudo-labels are critical in attenuating the unwanted deviations caused by the false and noisy hard pseudo-labels.

Effectiveness of Different Mixup. With mixup regularizations working for both discriminator and classifier, the average performance on Office-Home dataset is 76.0%. By removing mixup regularization from the training of domain discriminator, the performance decreases to 73.6%. Similarly, by removing mixup regularization from the classifier training, the average performance becomes 73.9%. This corroborates the fact that our Mix strategy not only helps to explore intrinsic structures across domains, but also helps to stabilize the domain discriminator.

Feature Visualizations. We use t-SNE (Maaten and Hinton 2008) to visualize the features learned using different components of our **SLM** framework. We choose an UDA setup (similar to DANN (Ganin et al. 2016)) as a vanilla method and add different modules one-by-one to visualize their individual contribution in learning discriminative features for partial domain adaptation. As seen from Figure 5, the feature space for vanilla setup lacks dicriminability for both source and target features. The discriminability improves for both source as well as target features as we add "Select" and "Label" to the Vanilla setup. The best results are obtained when all three modules "Select", "Label" and "Mix" i.e., **SLM** are added and trained jointly in an end-to-end manner. More visualizations including additional results with different backbones are included in supplementary material.

Conclusion

In this paper, we propose an end-to-end framework for learning discriminative invariant feature representation while preventing negative transfer in partial domain adaptation. While our select module facilitates the identification of relevant source samples for adaptation, the label module enhances the discriminability of the latent space by utilizing pseudo-labels for the target domain samples. The mix module uses mixup regularizations jointly with the other two strategies to enforce domain invariance in latent space. We demonstrate the effectiveness of our approach on four standard datasets, outperforming several competing methods.

References

Bucci, S.; D'Innocente, A.; and Tommasi, T. 2019. Tackling partial domain adaptation with self-supervision. In *International Conference on Image Analysis and Processing*, 70–81. Springer.

Cao, Z.; Long, M.; Wang, J.; and Jordan, M. I. 2018a. Partial transfer learning with selective adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2724–2732.

Cao, Z.; Ma, L.; Long, M.; and Wang, J. 2018b. Partial adversarial domain adaptation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 135–150.

Cao, Z.; You, K.; Long, M.; Wang, J.; and Yang, Q. 2019. Learning to transfer examples for partial domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2985–2994.

Chang, W.-G.; You, T.; Seo, S.; Kwak, S.; and Han, B. 2019. Domain-specific batch normalization for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 7354–7362.

Chen, C.; Chen, Z.; Jiang, B.; and Jin, X. 2019a. Joint domain alignment and discriminative feature learning for unsupervised deep domain adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 3296–3303.

Chen, J.; Wu, X.; Duan, L.; and Gao, S. 2019b. Domain Adversarial Reinforcement Learning for Partial Domain Adaptation. *arXiv preprint arXiv:1905.04094*.

Chen, Z.; Chen, C.; Cheng, Z.; Jiang, B.; Fang, K.; and Jin, X. 2020. Selective transfer with reinforced transfer network for partial domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 12706–12714.

Csurka, G. 2017. Domain adaptation for visual applications: A comprehensive survey. *arXiv preprint arXiv:1702.05374*.

Ganin, Y.; and Lempitsky, V. 2015. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*, 1180–1189. PMLR.

Ganin, Y.; Ustinova, E.; Ajakan, H.; Germain, P.; Larochelle, H.; Laviolette, F.; Marchand, M.; and Lempitsky, V. 2016. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1): 2096–2030.

Gretton, A.; Borgwardt, K. M.; Rasch, M. J.; Schölkopf, B.; and Smola, A. 2012. A kernel two-sample test. *The Journal of Machine Learning Research*, 13(1): 723–773.

Griffin, G.; Holub, A.; and Perona, P. 2007. Caltech-256 object category dataset.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

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

Hu, J.; Tuo, H.; Wang, C.; Qiao, L.; Zhong, H.; and Jing, Z. 2019. Multi-Weight Partial Domain Adaptation. In *BMVC*, 5.

Hu, L.; Kan, M.; Shan, S.; and Chen, X. 2018. Duplex generative adversarial network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1498–1507.

Inoue, N.; Furuta, R.; Yamasaki, T.; and Aizawa, K. 2018. Cross-domain weakly-supervised object detection through progressive domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 5001–5009.

Jang, E.; Gu, S.; and Poole, B. 2016. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*.

Kang, G.; Jiang, L.; Yang, Y.; and Hauptmann, A. G. 2019. Contrastive adaptation network for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4893–4902.

Li, S.; Liu, C. H.; Lin, Q.; Wen, Q.; Su, L.; Huang, G.; and Ding, Z. 2020. Deep residual correction network for partial domain adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Liang, J.; Hu, D.; and Feng, J. 2020. Do We Really Need to Access the Source Data? Source Hypothesis Transfer for Unsupervised Domain Adaptation. *arXiv preprint arXiv:2002.08546*.

Liang, J.; Wang, Y.; Hu, D.; He, R.; and Feng, J. 2020. A balanced and uncertainty-aware approach for partial domain adaptation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*, 123–140. Springer.

Long, M.; Cao, Y.; Wang, J.; and Jordan, M. 2015. Learning Transferable Features with Deep Adaptation Networks. In *International conference on machine learning*, 97–105. PMLR.

Long, M.; Cao, Z.; Wang, J.; and Jordan, M. I. 2018. Conditional Adversarial Domain Adaptation. In *Neural Information Processing Systems*, 1640–1650.

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

Maaten, L. v. d.; and Hinton, G. 2008. Visualizing data using t-SNE. *Journal of machine learning research*, 9(Nov): 2579–2605.

Maddison, C. J.; Mnih, A.; and Teh, Y. W. 2016. The concrete distribution: A continuous relaxation of discrete random variables. *arXiv preprint arXiv:1611.00712*.

Mao, X.; Ma, Y.; Yang, Z.; Chen, Y.; and Li, Q. 2019. Virtual mixup training for unsupervised domain adaptation. *arXiv preprint arXiv:1905.04215*.

Mei, K.; Zhu, C.; Zou, J.; and Zhang, S. 2020. Instance Adaptive Self-Training for Unsupervised Domain Adaptation. *arXiv preprint arXiv:2008.12197*.

Müller, R.; Kornblith, S.; and Hinton, G. E. 2019. When does label smoothing help? In *Advances in Neural Information Processing Systems*, 4694–4703.

Murez, Z.; Kolouri, S.; Kriegman, D.; Ramamoorthi, R.; and Kim, K. 2018. Image to image translation for domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4500–4509.

Pei, Z.; Cao, Z.; Long, M.; and Wang, J. 2018. Multi-adversarial domain adaptation. *arXiv preprint arXiv:1809.02176*.

Peng, X.; Usman, B.; Kaushik, N.; Hoffman, J.; Wang, D.; and Saenko, K. 2017. Visda: The visual domain adaptation challenge. *arXiv preprint arXiv:1710.06924*.

Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. 2015. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3): 211–252.

Saenko, K.; Kulis, B.; Fritz, M.; and Darrell, T. 2010. Adapting visual category models to new domains. In *European conference on computer vision*, 213–226. Springer.

Saito, K.; Ushiku, Y.; and Harada, T. 2017. Asymmetric tri-training for unsupervised domain adaptation. *arXiv preprint arXiv:1702.08400*.

Shen, J.; Qu, Y.; Zhang, W.; and Yu, Y. 2017. Wasserstein distance guided representation learning for domain adaptation. *arXiv preprint arXiv:1707.01217*.

Simonyan, K.; and Zisserman, A. 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *International Conference on Learning Representations*.

Sun, B.; and Saenko, K. 2016. Deep coral: Correlation alignment for deep domain adaptation. In *European conference on computer vision*, 443–450. Springer.

Tzeng, E.; Hoffman, J.; Saenko, K.; and Darrell, T. 2017. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 7167–7176.

Venkateswara, H.; Eusebio, J.; Chakraborty, S.; and Panchanathan, S. 2017. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 5018–5027.

Wang, M.; and Deng, W. 2018. Deep visual domain adaptation: A survey. *Neurocomputing*, 312: 135–153.

Wang, Q.; and Breckon, T. 2020. Unsupervised domain adaptation via structured prediction based selective pseudo-labeling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 6243–6250.

Wu, Y.; Inkpen, D.; and El-Roby, A. 2020. Dual mixup regularized learning for adversarial domain adaptation. In *European Conference on Computer Vision*, 540–555. Springer.

Xu, M.; Zhang, J.; Ni, B.; Li, T.; Wang, C.; Tian, Q.; and Zhang, W. 2019a. Adversarial domain adaptation with domain mixup. *arXiv preprint arXiv:1912.01805*.

Xu, R.; Li, G.; Yang, J.; and Lin, L. 2019b. Larger norm more transferable: An adaptive feature norm approach for unsupervised domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, 1426–1435.

Yan, S.; Song, H.; Li, N.; Zou, L.; and Ren, L. 2020. Improve Unsupervised Domain Adaptation with Mixup Training. *arXiv preprint arXiv:2001.00677*.

Zhang, H.; Cisse, M.; Dauphin, Y. N.; and Lopez-Paz, D. 2017. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*.

Zhang, J.; Ding, Z.; Li, W.; and Ogunbona, P. 2018. Importance weighted adversarial nets for partial domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 8156–8164.

Zhang, Y.; Deng, B.; Jia, K.; and Zhang, L. 2020. Label Propagation with Augmented Anchors: A Simple Semi-Supervised Learning baseline for Unsupervised Domain Adaptation. In *European Conference on Computer Vision*, 781–797. Springer.

Zou, Y.; Yu, Z.; Liu, X.; Kumar, B.; and Wang, J. 2019. Confidence regularized self-training. In *Proceedings of the IEEE International Conference on Computer Vision*, 5982–5991.

Zou, Y.; Yu, Z.; Vijaya Kumar, B.; and Wang, J. 2018. Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In *Proceedings of the European conference on computer vision (ECCV)*, 289–305.

Supplementary Material

Section	Content
A	Dataset Details
B	Implementation Details
C	Additional Experimental Results
D	Qualitative Results
E	Limitations and Broader Impact

Table 6: Overview of Supplementary Material.

Dataset Details

We evaluate the performance of our approach on several benchmark datasets for partial domain adaptation, namely Office31 (Saenko et al. 2010), Office-Home (Venkateswara et al. 2017), ImageNet-Caltech and VisDA-2017 (Peng et al. 2017). The following are the detailed descriptions of the above datasets:

Office31. This dataset contains 4,110 images distributed among 31 different classes and collected from three different domains: Amazon (A), Webcam (W) and DSLR (D), resulting in 6 transfer tasks. The dataset is imbalanced across domains with 2,817 images belonging to Amazon, 795 images to Webcam, and 498 images to DSLR, making Amazon a larger domain as compared to Webcam and DSLR. For all our experiments, we select the 10 classes shared by Office31 and Caltech256 (Griffin, Holub, and Perona 2007) as the target categories and obtain the following label spaces:
$\mathcal{L}_{source} = \{0, 1, 2, ..., 30\}$.
$\mathcal{L}_{target} = \{0, 1, 5, 10, 11, 12, 15, 16, 17, 22\}$.
Number of Outlier Classes = 21.
Figure 6 shows few randomly sampled images from this dataset. The dataset is publicly available to download at: https://people.eecs.berkeley.edu/~jhoffman/domainadapt/#datasets_code.

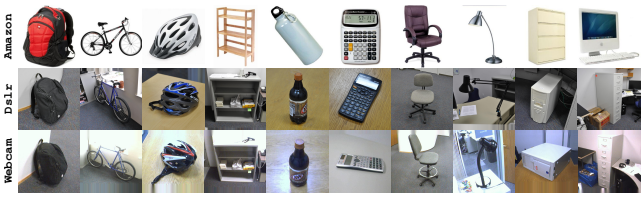

Figure 6: **Sampled Images from Office31 Dataset**. Each row from top to bottom corresponds to the domains Amazon, Dslr and Webcam, respectively. The images in the same column belong to the same class. Best viewed in color.

Office-Home. This dataset contains 15,588 images distributed among 65 different classes and collected from four different domains: Art (Ar), Clipart (Cl), Product (Pr), and RealWorld (Rw), resulting in 12 transfer tasks. The dataset is split across domains with 2427 images belonging to Art, 4365 images to Clipart, 4439 images to Product, and 4347 images to RealWorld. We select the first 25 categories (in alphabetic order) in each domain as the target classes and obtain the following label spaces:
$\mathcal{L}_{source} = \{0, 1, 2, ..., 64\}$.
$\mathcal{L}_{target} = \{0, 1, 2, ..., 24\}$.
Number of Outlier Classes = 40.
Figure 7 displays a gallery of sample images for this dataset. The dataset is publicly available to download at: http://hemanthdv.org/OfficeHome-Dataset/.

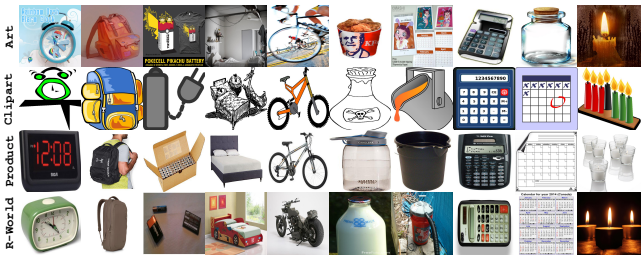

Figure 7: **Sampled Images from Office-Home Dataset**. Each row from top to bottom corresponds to the domains Art, Clipart, Product and RealWorld, respectively. The images in the same column belong to the same class. Best viewed in color.

ImageNet-Caltech. This large-scale dataset consists of two datasets (ImageNet1K (Russakovsky et al. 2015) (I) & Caltech256 (Griffin, Holub, and Perona 2007) (C)) as two separate domains and consist of over 14 million images combined. 2 transfer tasks are formed for this dataset. While source domain contains 1,000 and 256 classes for ImageNet and Caltech respectively, each target domain contains only 84 classes that are common across both domains. As it is a general practice to use ImageNet pretrained weights for network initialization, we use the validation set images when using ImageNet as the target domain. Number of Outlier Classes = 172 for C→I, 916 for I→C. Figure 8 displays a gallery of sample images for this dataset. The datasets are publicly available to download at: http://www.image-net.org/
http://www.vision.caltech.edu/Image_Datasets/Caltech256/.

VisDA-2017. This dataset contains 280,157 images distributed among 12 different classes and two domains. The dataset contains three sets of images: training, validation and testing. The training set contains 152,397 synthetic (S) images, the validation set contains 55,388 real-world (R) images, while the test set contains 72,372 real-world images. For the experiments, the training set is considered as the Synthetic (S) domain, while the validation set as the Real (R) domain, following (Li et al. 2020). This results in 2 transfer tasks. The first 6 categories (in alphabetical order) are selected in each of the domains as the target classes, and the following label spaces are obtained:

Figure 8: **Sampled Images from ImageNet-Caltech Dataset**. The top row corresponds to the ImageNet domain, while the bottom row to the Caltech domain. The images in the same column belong to the same class. Best viewed in color.

Figure 9: **Sampled Images from VisDA-2017 Dataset**. The top row corresponds to the Synthetic domain, while the bottom row to the Real domain. The images in the same column belong to the same class. Best viewed in color.

$\mathcal{L}_{source} = \{0, 1, 2, ..., 11\}$.
$\mathcal{L}_{target} = \{0, 1, 2, ..., 5\}$.
Number of Outlier Classes = 6.
Figure 9 displays a gallery of sample images for this dataset. The dataset is publicly available to download at: http://ai.bu.edu/visda-2017/#download.

Implementation Details

The training pipeline pseudo-code for **SLM** is shown in Algorithm 1. Following are the detailed description of the implementation we follow for various components of the framework:

Algorithm 1: The training pipeline for **SLM**

Data: source data $\mathcal{D}_{source}$ and target data $\mathcal{D}_{target}$.
Networks: Selector Network $\mathcal{H}(.)$, Feature Extractor $\mathcal{G}(.)$, Classifier $\mathcal{F}(.)$, and Domain Discriminator $\mathcal{D}(.)$.

Initialize networks $\mathcal{G}(.)$, $\mathcal{F}(.)$, $\mathcal{H}(.)$, and $\mathcal{D}(.)$ in **SLM** .
for $itrn = 1 \rightarrow num\_itrn$ **do**
  Obtain the mini-batches $\mathcal{D}^b_{source}$ and $\mathcal{D}^b_{target}$.
  # "Select" Module
  Obtain the binary decisions from $\mathcal{H}(\mathcal{D}^b_{source})$ and use them to obtain $\mathcal{D}^b_{sel}$ and $\mathcal{D}^b_{dis}$.
  # "Label" Module
  Obtain soft pseudo-labels $\hat{y}^b$ from $\mathcal{F}(\mathcal{G}(\mathcal{D}_{target}))$ for $\mathcal{D}_{target}$.
  # "Mix" Module
  Obtain $\mathcal{D}^b_{inter\_mix}$, $\mathcal{D}^b_{intra\_mix\_s}$, and $\mathcal{D}^b_{intra\_mix\_t}$.
  Compute $\mathcal{L}_{sup}$, $\mathcal{L}_{adv}$, $\mathcal{L}_{select}$, $\mathcal{L}_{label}$, and $\mathcal{L}_{mix}$.
  Compute the gradients and backpropagate for optimization using gradient descent.
end for

Feature Extractor ($\mathcal{G}$). We use ResNet-50 (He et al. 2016) backbone for the feature extractor. The overall structure of ResNet-50 is `Initial Layers, Layer-1, Layer-2, Layer-3, Layer-4, AvgPool, Fc`. The model is initialized with ImageNet (Russakovsky et al. 2015) pretrained weights. Additionally, we add a bottleneck layer of width 256 just after the `AvgPool` layer to obtain the features and replace all the BatchNorm layers with Domain-Specific Batch-Normalization (Chang et al. 2019) layers. All the layers till `Layer-3` are frozen and only the rest of the layers are fine-tuned.

Selector Network ($\mathcal{H}$). We use a ResNet-18 (He et al. 2016) network with the `Fc` layer replaced with a binary-length fully connected layer as the selector network in our framework. The network is initialized with ImageNet pretrained weights and all the layers are trained while optimization.

Classifier ($\mathcal{F}$). The final `Fc` layer of ResNet-50 described above is replaced with a task-specific fully-connected layer to form the classifier network of our framework.

Domain Discriminator ($\mathcal{D}$). A three-layer fully-connected network is used as the domain discriminator network. It takes the 256-length features obtained from the feature extractor as input. The adversarial training is incorporated using a gradient reversal layer (GRL).

Hyperparameters. All the networks are optimised using mini-batch stochastic gradient descent with a momentum of 0.9. A batch size of 64 is used for Office31 and VisDA-2017 while a batch size of 128 is used for Office-Home and ImageNet-Caltech. For feature extractor an initial learning rate of 5e-5 for the convolutional layers while an initial learning rate of 5e-4 for all the fully-connected layers is used. For the selector network and the domain discriminator an initial learning rate of 5e-3 and 5e-4 are used respectively. The learning rates are decayed following a cosine-annealing strategy as the training progresses. The best models are captured by obtaining the performance on a validation set. We do NOT follow the ten-crop technique (Cao et al. 2018b, 2019), to improve the performance in the inference phase. We obtain the best hyperparameters using grid search. All the experiments were averaged over three runs, which used random seed values of 1, 2, and 3 respectively.

Hardware and Software Details. All the experiments were conducted using a single `NVIDIA Tesla V100-DGXS GPU` with `32 GigaBytes` of memory, equipped with a `Intel(R) Xeon(R) CPU E5-2698 v4 @ 2.20GHz`. We used PyTorch v1.4.0, Python v3.6.10 to implement the codes.

Additional Experimental Results

Effectiveness on Different Backbone Networks. To show that the proposed framework is backbone-agnostic, i.e., it provides the best performance irrespective of the architecture of the feature extractor, we conduct experiments using a VGG-16 (Simonyan and Zisserman 2015) backbone for the feature extractor. We report the results on the transfer tasks from the Office31 dataset in Table 7 and compare it with the current state-of-the-art methods. Our method outperforms the previously best results by a margin of $\mathbf{3.0}\%$ on average and achieves new state-of-the-art results. This confirms that our proposed framework for partial domain adaptation is robust with respect to the change of backbone network.

Effectiveness of Individual Modules. In Section 4.3 of the main paper, we discussed the importance of the proposed three unique modules on three datasets and provided the average accuracies for wach of them. Here, we provide the performance on the individual transfer tasks for those three datasets, namely Office-31, VisDA-2017, and Office-Home in Tables 8 and 9. Our approach with all the three modules (Select, Label and Mix) working jointly, works the best on all the three datasets.

Effectiveness of Hausdorff Distance. In Section 4.3 of the

Office31							
Method	**A → W**	**D → W**	**W → D**	**A → D**	**D → A**	**W → A**	**Average**
VGG-16 (Simonyan and Zisserman 2015) (ICLR'15)	$60.3_{\pm 0.8}$	$98.0_{\pm 0.6}$	$99.4_{\pm 0.4}$	$76.4_{\pm 0.5}$	$73.0_{\pm 0.6}$	$79.1_{\pm 0.5}$	81.0
PADA (Cao et al. 2018b) (ECCV'18)	$\underline{86.1}_{\pm 0.4}$	$\mathbf{100.0}_{\pm 0.0}$	$\mathbf{100.0}_{\pm 0.0}$	$81.7_{\pm 0.3}$	$93.0_{\pm 0.2}$	$\underline{95.3}_{\pm 0.3}$	92.5
SAN (Cao et al. 2018a) (CVPR'18)	$83.4_{\pm 0.4}$	$99.3_{\pm 0.5}$	$\mathbf{100.0}_{\pm 0.0}$	$90.7_{\pm 0.2}$	$87.2_{\pm 0.2}$	$91.9_{\pm 0.4}$	92.1
IWAN (Zhang et al. 2018) (CVPR'18)	$82.9_{\pm 0.3}$	$79.8_{\pm 0.3}$	$88.5_{\pm 0.2}$	$\underline{91.0}_{\pm 0.3}$	$89.6_{\pm 0.2}$	$93.4_{\pm 0.2}$	87.5
ETN (Cao et al. 2019) (CVPR'19)	$85.7_{\pm 0.2}$	$\mathbf{100.0}_{\pm 0.0}$	$\mathbf{100.0}_{\pm 0.0}$	$89.4_{\pm 0.2}$	$\underline{95.9}_{\pm 0.2}$	$92.3_{\pm 0.2}$	$\underline{93.9}$
SLM (Ours)	$\mathbf{92.0}_{\pm 0.1}$	$\underline{99.8}_{\pm 0.2}$	$\underline{99.6}_{\pm 0.5}$	$\mathbf{98.1}_{\pm 0.0}$	$\mathbf{96.1}_{\pm 0.0}$	$\mathbf{96.0}_{\pm 0.1}$	**96.9**

Table 7: **Performance on Office31 with VGG-16 backbone.** Numbers show the accuracy (%) of different methods on partial domain adaptation setting. We highlight the **best** and second best method on each transfer task. Our proposed framework, **SLM** achieves the best performance on 4 out of 6 transfer tasks including the best average performance among all compared methods.

Modules			Office31							VisDA-2017		
Select	**Label**	**Mix**	**A → W**	**D → W**	**W → D**	**A → D**	**D → A**	**W → A**	**Average**	**R → S**	**S → R**	**Average**
✗	✗	✗	88.0	98.3	95.8	88.8	84.5	80.2	89.3	57.7	56.4	57.0
✓	✗	✗	91.8	99.3	96.6	93.8	94.2	93.5	94.9	69.0	68.4	68.7
✓	✓	✗	92.4	99.9	99.2	94.9	95.5	93.8	96.0	77.2	84.8	81.0
✓	✓	✓	99.8	100.0	99.8	98.7	96.1	95.9	98.4	77.5	91.7	84.6

Table 8: **Effectiveness of Different Modules on Office31 and VisDA-2017 Datasets.** Our proposed approach achieves the best performance with all the modules working jointly for learning discriminative invariant features in partial domain adaptation.

Modules			Office-Home											
Select	**Label**	**Mix**	**Ar → Cl**	**Ar → Pr**	**Ar → Rw**	**Cl → Ar**	**Cl → Pr**	**Cl → Rw**	**Pr → Ar**	**Pr → Cl**	**Pr → Rw**	**Rw → Ar**	**Pr → Cl**	**Pr → Rw**	**Average**
✗	✗	✗	44.2	61.6	75.9	54.6	55.2	65.0	51.0	37.3	69.6	64.8	42.4	71.4	57.7
✓	✗	✗	50.6	72.9	79.2	65.4	67.2	71.7	60.8	46.7	77.0	71.9	49.4	77.0	65.8
✓	✓	✗	56.1	82.4	89.8	74.2	73.0	81.6	70.8	48.4	87.0	80.1	53.1	81.7	73.2
✓	✓	✓	61.1	84.0	91.4	76.5	75.0	81.8	74.6	55.6	87.8	82.3	57.8	83.5	76.0

Table 9: **Effectiveness of Different Modules on Office-Home Dataset.** Our proposed approach achieves the best performance with all the modules working jointly for learning discriminative invariant features in partial domain adaptation.

main paper, we discussed the importance of Hausdorff Distance loss in guiding the selector network to discard the outlier source samples. Here we provide the individual performance of all the transfer tasks on Office-Home dataset in Table 10, which shows that our approach with Hausdorff distance loss works the best in all cases.

Effectiveness of Soft Pseudo-Labels. As discussed in the Section 4.3, we confirmed the importance of soft pseudo-labels for our framework as it attenuates the unwanted deviations because of noisy and false hard pseudo-labels. Here, we provide the performance on each of the transfer tasks from Office-Home in Table 11.

Effectiveness of Different MixUp. We examined the effect of mixup regularization on both domain discriminator and classifier separately in Section 4.3 of the main paper. We concluded that our Mix strategy not only helps to explore intrinsic structures across domains, but also helps to stabilize the domain discriminator. Here, we provide the corresponding performance on each of the transfer tasks of Office-Home in Table 12.

Qualitative Results

Feature Visualizations. We provide some additional feature visualizations using t-SNE (Maaten and Hinton 2008) in Figure 10. Similar to Section 4.4 in the main paper, we choose an UDA setup as a vanilla method and add the proposed modules one-by-one to visualize the contribution of each of the modules in learning discriminative features for partial domain adaptation.

Limitations and Broader Impact

Our research can help reduce burden of collecting large-scale supervised data in many real-world applications of visual classification by transferring knowledge from models trained on large broad datasets to specific datasets possessing a domain shift. This scenario is quite common as large datasets (e.g. ImageNet (Russakovsky et al. 2015)) can be used for training which contain a broader range of categories while our goal can be to transfer the knowledge to smaller datasets with a smaller number of categories. The positive impact that our work could have on society is in making technology more accessible for institutions and individuals that do not have rich resources for annotating newly collected datasets. Negative impacts of our research are difficult to predict, however, it shares many of the pitfalls associated with standard deep learning models such as susceptibility to adversarial attacks and lack of interpretablity.

	Ar → Cl	Ar → Pr	Ar → Rw	Cl → Ar	Cl → Pr	Cl → Rw	Pr → Ar	Pr → Cl	Pr → Rw	Rw → Ar	Pr → Cl	Pr → Rw	Average
W/o Hausdorff Loss	56.2	83.1	90.3	72.6	71.5	80.8	71.4	51.6	84.8	82.5	57.5	81.7	73.7
Ours (**SLM**)	61.1	84.0	91.4	76.5	75.0	81.8	74.6	55.6	87.8	82.3	57.8	83.5	76.0

Table 10: **Effectiveness of Hausdorff Triplet Loss on Office-Home Dataset.** The table shows the performance of the framework without (top-row) and with (bottom-row) the inclusion of the Hausdorff distance triplet loss. The results highlight the importance of the Hausdorff distance loss in our proposed framework.

	Ar → Cl	Ar → Pr	Ar → Rw	Cl → Ar	Cl → Pr	Cl → Rw	Pr → Ar	Pr → Cl	Pr → Rw	Rw → Ar	Pr → Cl	Pr → Rw	Average
W/ Hard Pseudo-labels	52.5	79.9	90.2	73.5	72.6	78.2	69.9	47.5	87.5	78.6	50.6	82.7	72.0
Ours (**SLM**)	61.1	84.0	91.4	76.5	75.0	81.8	74.6	55.6	87.8	82.3	57.8	83.5	76.0

Table 11: **Effectiveness of Soft Pseudo-labels on Office-Home Dataset.** Table shows the performance of the framework when we replace the soft pseudo-labels with hard pseudo-labels (top-row) for the target samples. The results justify that the soft pseudo-labels are critical for our framework and attenuate unwanted deviations caused by hard pseudo-labels.

	Ar → Cl	Ar → Pr	Ar → Rw	Cl → Ar	Cl → Pr	Cl → Rw	Pr → Ar	Pr → Cl	Pr → Rw	Rw → Ar	Pr → Cl	Pr → Rw	Average
No Domain Discriminator MixUp	56.2	81.5	90.0	74.0	71.8	80.3	72.2	50.9	86.3	79.8	58.0	82.0	73.6
No Classifier MixUp	57.8	82.9	88.5	75.1	73.6	79.3	69.0	54.9	86.6	79.8	57.6	81.2	73.9
Ours (**SLM**)	61.1	84.0	91.4	76.5	75.0	81.8	74.6	55.6	87.8	82.3	57.8	83.5	76.0

Table 12: **Effectiveness of Different MixUp on Office-Home Dataset.** The table shows the performance of the framework with the exclusion of mixup regularization from the domain discriminator (top-row) and the classsifier (middle-row). The final row shows the results of the proposed **SLM** framework, which provides the best performance confirming the importance of our Mix strategy.

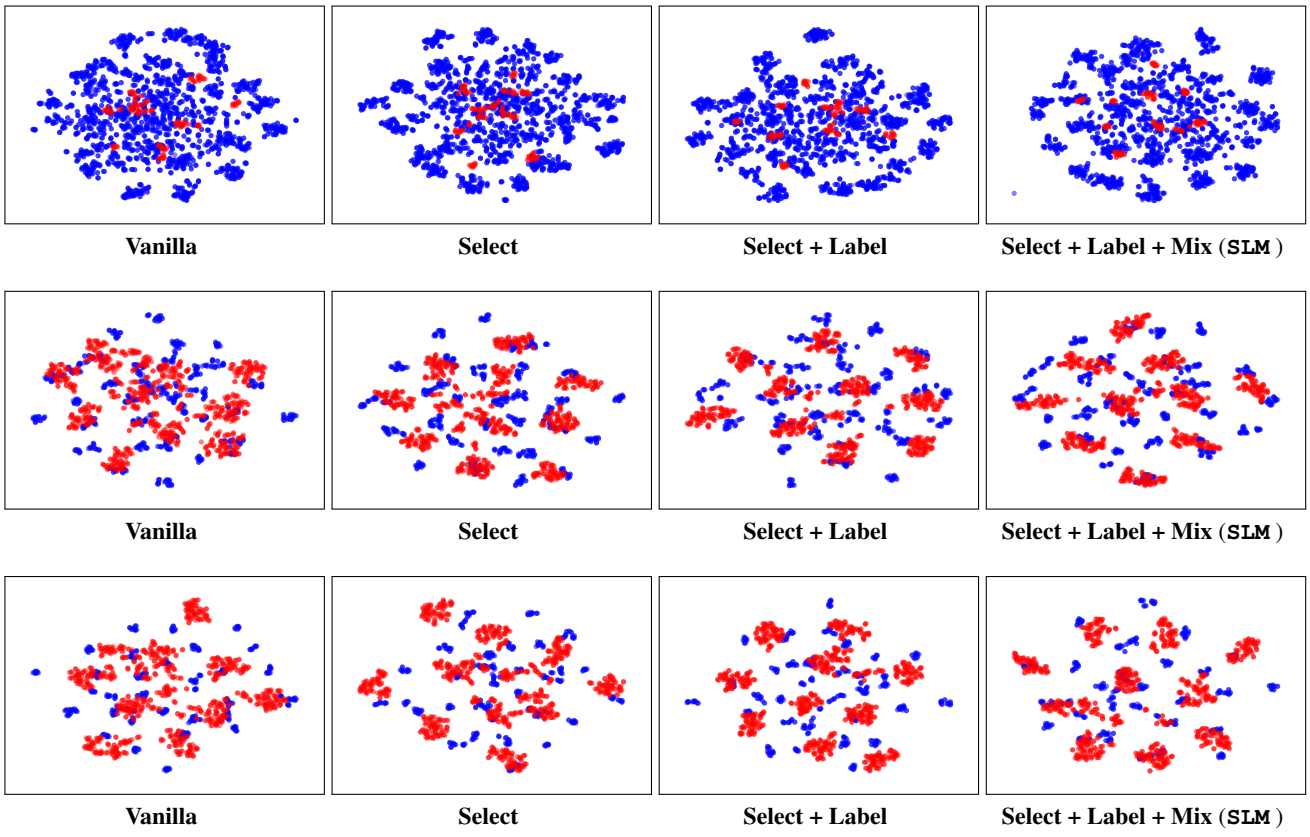

Figure 10: **Feature Visualizations using t-SNE.** Plots show visualization of our approach with different modules on A→W, W→A, and D→A tasks repectively (top to down) from Office31 dataset. Blue and red dots represent source and target data respectively. As can be seen, features for both target as well as source domain become progressively discriminative and improve from left to right by adoption of our proposed modules. Best viewed in color.