

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

Aadarsh Sahoo¹ Rameswar Panda¹ Rogerio Feris¹ Kate Saenko^{1,2} Abir Das³

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

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

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 regularization 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 for partial domain adaptation demonstrate the superiority of our proposed framework over state-of-the-art methods. Project page: <https://cvir.github.io/projects/slm>.

1. Introduction

Deep neural networks usually have recently shown impressive performance on many visual tasks by leveraging large collections of labeled data. However, they usually do not generalize well to domains that are not distributed identically to the training data. Domain adaptation [10, 46] 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 [11, 13, 25, 26, 27]. Since

large-scale labelled datasets are readily accessible as source domain data, a more realistic scenario is partial domain adaptation (PDA), which assumes that target label space is a subset of source label space, that has received increasing research attention recently [2, 8, 9, 17].

Several methods have been proposed to solve partial domain adaptation by reweighting the source domain samples [2, 8, 9, 17, 50, 54]. 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 the 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 the 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 the 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 partial domain adaptation. 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 [12]. As these decision functions are discrete and non-differentiable, we rely on Gumbel Softmax sampling [20] to learn the policy jointly with network parameters through standard back-propagation, without resorting to complex reinforcement learning settings, as in [8, 9]. 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 do-

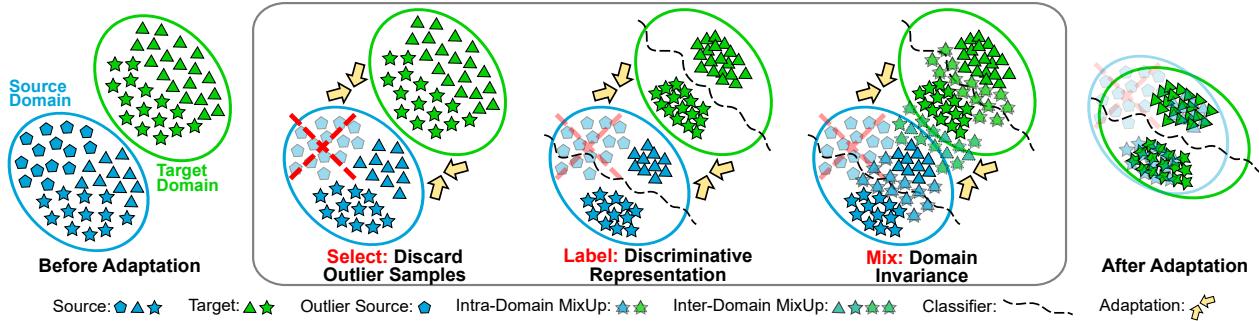


Figure 1: A conceptual overview of our approach. Our proposed approach adopts three unique modules namely Select, Label and Mix in a unified framework to mitigate domain shift and generalize the model to an unlabelled target domain with a label space which is a subset of that of the labelled source domain. 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.

main to enhance the discriminability of the latent space. Finally, the mix module utilizes both intra- and inter-domain mixup regularizations [53] to generate convex combinations of pairs of training samples and their 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 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 samples and learns discriminative invariant features by labeling and mixing samples. Experiments on four datasets illustrate the effectiveness of our proposed framework in achieving new state-of-the-art performance for partial domain adaptation (e.g., our approach outperforms DRCN [22] by 18.9% on the challenging VisDA-2017 [36] benchmark). To summarize, 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 back-propagation through Gumbel Softmax sampling.
- We conduct extensive experiments on four datasets, including Office31 [38], Office-Home [44], ImageNet-Caltech, and VisDA-2017 [36] to demonstrate superiority of our approach over state-of-the-art methods.

2. Related Works

Unsupervised Domain Adaptation. Unsupervised domain adaptation which aims to leverage labeled source

domain data to learn to classify unlabeled target domain data has been studied from multiple perspectives (see reviews [10, 46]). Various strategies have been developed for unsupervised domain adaptation, including methods for reducing cross-domain divergence [13, 25, 40, 42], adding domain discriminators for adversarial training [7, 11, 12, 25, 26, 27, 35, 43] and image-to-image translation techniques [16, 18, 33]. 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 [3, 4, 54] with weighting, or use residual correction blocks [22, 24], or use source examples based on their similarities to target domain [5]. Most relevant to our approach is the work in [8, 9] 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 proposed 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 [19, 31, 39, 55]. Majority of the methods directly choose hard pseudo-labels with high prediction confidence. The works in [56, 57] use class-balanced confidence regularizers to generate soft pseudo-labels for unsupervised domain adaptation that share same label space across domains. Our

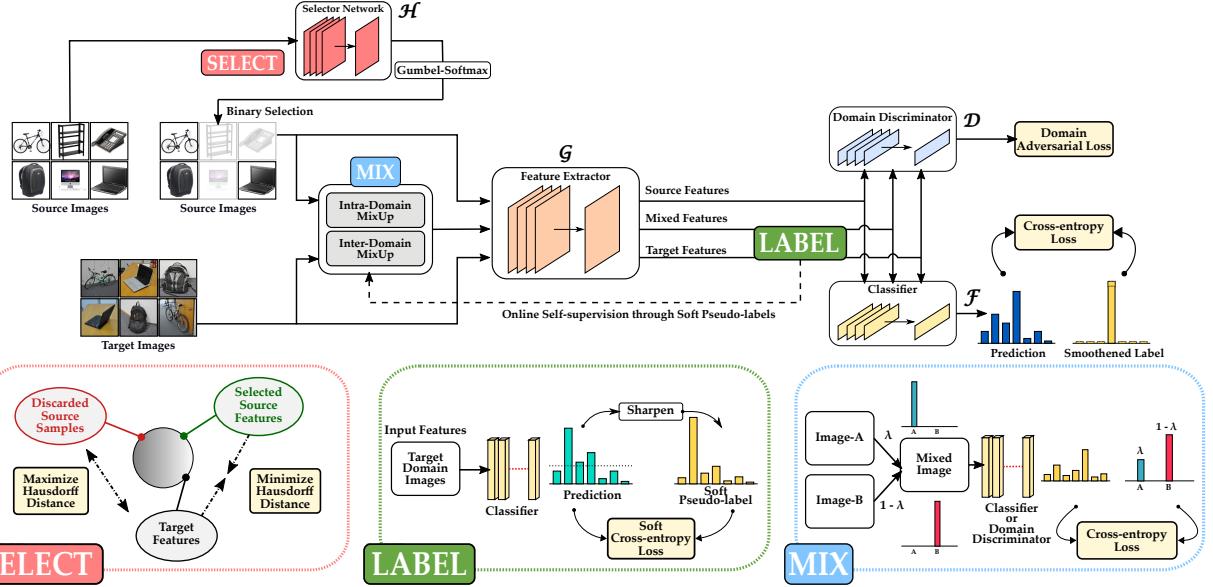


Figure 2: Illustration of our proposed framework. Our framework consists of a feature extractor \mathcal{G} which maps the images to a common latent feature space, a classifier network \mathcal{F} to provide class-wise predictions, a domain discriminator \mathcal{D} to reduce domain discrepancy, and 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 3 for more details. Best viewed in color.

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 [53] or its variants [1, 45] 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 [48, 49, 51] or to smoothen the predictions [30]. 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.

3. Methodology

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 target 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 partial domain adaptation, 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} .

3.1. Approach Overview

Figure 2 illustrates an overview of our proposed 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 $\mathcal{F}(\mathcal{G}(\cdot))$ 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 [20]. On the other hand, label module utilizes predictions of $\mathcal{F}(\mathcal{G}(\cdot))$ 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

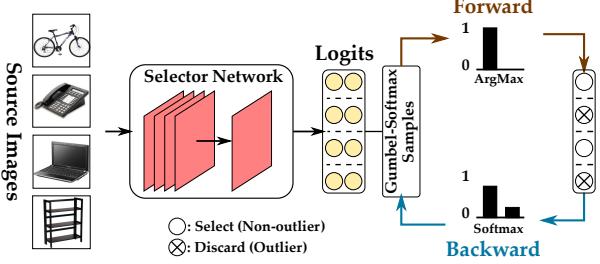


Figure 3: **Learning with Gumbel Softmax Sampling.** Gumbel-Softmax trick used for enabling gradient-based optimization for discrete output space. Best viewed in color.

the components are trained jointly and during testing, we evaluate performance using classification accuracy of the network $\mathcal{F}(\mathcal{G}(\cdot))$ on target domain data \mathcal{D}_{target} . The individual modules are discussed below.

3.2. Select Module

This module stands in the core of our framework with an aim to get rid of the outlier source samples in the source domain in order to minimize negative transfer in partial domain adaptation. 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} performs robust selection by providing 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 [20, 29] 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. As shown in Figure 3, the selector network \mathcal{H} takes a batch of source 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 $U \sim \text{Uniform}[0, 1]$ 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. 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}_{source}^b and target domain images \mathcal{D}_{target}^b , each of size b , the selector results in two subsets of source samples $\mathcal{D}_{sel} = \{x \in \mathcal{D}_{source}^b : \mathcal{H}(x) = 1\}$ and $\mathcal{D}_{dis} = \{x \in \mathcal{D}_{source}^b : \mathcal{H}(x) = 0\}$. The idea is to pull the *selected* source samples \mathcal{D}_{sel} & target samples \mathcal{D}_{target}^b closer while pushing *discarded* source samples \mathcal{D}_{dis} & \mathcal{D}_{target}^b apart in the output latent feature space of \mathcal{G} . To achieve this, we formulate the loss function as follows:

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

where $d_H(X, Y)$ represents the average Hausdorff distance between the set of features X and Y . $\mathcal{L}_{reg} = \lambda_{reg1} \sum_{x \in \mathcal{D}_{source}^b} \mathcal{H}(x) \log(\mathcal{H}(x)) + \lambda_{reg2} \{\sum_{\hat{p}} l_{ent}(\hat{p}) - l_{ent}(\hat{p}_m)\}$, 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 $\mathcal{F}(\mathcal{G}(\cdot))$ for \mathcal{D}_{target} . Note that only \mathcal{D}_{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 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.

3.3. 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 [56], 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 $\mathbf{x}_k^t \in \mathcal{D}_{target}$, the soft-pseudo-label \hat{y}_k^t is computed as follows:

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

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

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

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

3.4. 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 [53] on the selected source samples and the target samples for discovering ingrained structures in establishing domain invariance. Given \mathcal{D}_{sel} from select module and \mathcal{D}_{target}^b 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:

$$\begin{aligned} \mathcal{D}_{inter_mix}^b &= \{(\lambda \mathbf{x}_i^s + (1-\lambda) \mathbf{x}_j^t, \lambda y_i + (1-\lambda) \hat{y}_j^t)\} \\ \mathcal{D}_{intra_mix_s}^b &= \{(\lambda \mathbf{x}_i^s + (1-\lambda) \mathbf{x}_j^s, \lambda y_i + (1-\lambda) y_j)\} \\ \mathcal{D}_{intra_mix_t}^b &= \{(\lambda \mathbf{x}_i^t + (1-\lambda) \mathbf{x}_j^t, \lambda \hat{y}_i^t + (1-\lambda) \hat{y}_j^t)\} \\ \mathcal{D}_{mix}^b &= \mathcal{D}_{inter_mix}^b \cup \mathcal{D}_{intra_mix_s}^b \cup \mathcal{D}_{intra_mix_t}^b \end{aligned} \quad (5)$$

where $(\mathbf{x}_{i/j}^s, y_{i/j}) \in \mathcal{D}_{sel}$, while $\mathbf{x}_{i/j}^t \in \mathcal{D}_{target}^b$ with $\hat{y}_{i/j}^t$ 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.

Given the new augmented images, we utilize the new augmented images in training both the classifier \mathcal{F} and the domain discriminator \mathcal{D} as follows:

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

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 partial domain adaptation, we use the standard supervised loss on the labeled source data and domain adversarial loss as follows:

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

where \mathcal{L}_{adv} is entropy-conditioned domain adversarial loss with weights w^s and w^t for source and target domain respectively [26]. The overall loss \mathcal{L}_{total} is

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

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

4. Experiments

We conduct extensive experiments to show that our **SLM** framework outperforms many competing approaches to achieve the new state-of-the-art on several PDA benchmark datasets. We also perform comprehensive ablation experiments and feature visualizations to verify the effectiveness of different components in detail.

4.1. Experimental Setup

Datasets. We evaluate our approach using four datasets under PDA setting, namely Office31 [38], Office-Home [44], ImageNet-Caltech and VisDA-2017 [36]. Office31 contains

4,110 images of 31 classes from three distinct domains, namely Amazon (A), Webcam (W) and DSLR (D). Following [9], we select 10 classes shared by Office31 and Caltech256 [14] as target categories. Office-Home is a challenging dataset that contains images from four domains: Artistic images (Ar), Clipart images (Cl), Product images (Pr) and Real-World images (Rw). We follow [9] 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) [37] and Caltech256 (C) [14]. 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: one consists photo-realistic images or real images (R), and the other comprises of 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[22]. More details are included in the Appendix.

Baselines. We compare our approach with several methods that fall into two main categories: (1) popular UDA methods (e.g., DANN [12], CORAL [42]) including recent methods like CAN [21] and SPL [47] which have shown state-of-the-art performance on UDA setup, (2) existing partial domain adaptation methods including PADA [4], SAN [3], ETN [5], and DRCN [22]. We also compare with the recent state-of-the-art methods, RTNNet [9] that uses reinforcement learning for source dataset selection, and BA³US [24] which uses source samples to augment the target domain in partial domain adaptation.

Implementation Details. We use ResNet-50 [15] as the backbone network for the feature extractor while we use ResNet-18 for the selector network, initialized with ImageNet [37] pretrained weights. All the weights are shared by both the source and target domain images except that of the BatchNorm layers, for which we use Domain-Specific Batch Normalization [6]. In Eqn. 2 we set λ_s , λ_{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 [20]. Additionally, we use label-smoothing for all the losses for the feature extractor involving source domain images as in [23, 32], with $\epsilon = 0.2$. We use SGD for optimization 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-

Method	Office31						Average
	A → W	D → W	W → D	A → D	D → A	W → A	
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	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	99.3 \pm 0.1	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	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	99.3 \pm 0.3	99.4 \pm 0.2	90.5 \pm 0.4	95.6 \pm 0.3	94.3 \pm 0.3	94.7
ETN	93.4 \pm 0.3	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	95.6	95.8	94.3
RTNNet	95.1 \pm 0.3	100.0 \pm 0.0	100.0 \pm 0.0	97.8 \pm 0.1	93.9 \pm 0.1	94.1 \pm 0.1	96.8
RTNNet _{adv}	96.2 \pm 0.3	100.0 \pm 0.0	100.0 \pm 0.0	97.6 \pm 0.1	92.3 \pm 0.1	95.4 \pm 0.1	96.9
BA ³ US	99.0 \pm 0.3	100.0 \pm 0.0	98.7 \pm 0.0	99.4 \pm 0.0	94.8 \pm 0.1	95.0 \pm 0.1	97.8
SLM (Ours)	99.8 \pm 0.2	100.0 \pm 0.0	99.8 \pm 0.3	98.7 \pm 0.0	96.1 \pm 0.1	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 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. **SLM** achieves the best performance on 4 out of 6 transfer tasks including the best average performance among all compared methods.

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. All the codes were implemented using PyTorch [34].

4.2. Results and Analysis

Table 1 shows the results of our proposed method and other competing approaches on the Office31 dataset. We have the following key observations. (1) As expected, the popular UDA methods including the recent CAN [21], 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, BA³US [24] 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 BA³US by **1.3%**. This shows that our method improves generalization ability of source classifier in target domain while reducing negative transfer.

On the challenging Office-Home dataset, our proposed approach obtains very competitive performance, with an average accuracy of 76.0% on this dataset (Table 2). Our method obtains the best on **6 out of 12** transfer tasks. Table 3 summarizes the results on ImageNet-Caltech and VisDA-2017 datasets. Our approach once again achieves

Method	Office-Home													Average
	Ar→Cl	Ar→Pr	Ar→Rw	Cl→Ar	Cl→Pr	Cl→Rw	Pr→Ar	Pr→Cl	Pr→Rw	Rw→Ar	Rw→Cl	Rw→Pr		
ResNet-50	47.2	66.8	76.9	57.6	58.4	62.5	59.4	40.6	75.9	65.6	49.1	75.8	61.3	
DANN	43.2	61.9	72.1	52.3	53.5	57.9	47.2	35.4	70.1	61.3	37.0	71.7	55.3	
CORAL	38.2	55.6	65.9	48.4	52.5	51.3	48.9	32.6	67.1	63.8	35.9	69.8	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	63.9	74.1	51.8	55.2	60.3	53.7	38.3	72.6	62.5	43.3	71.3	57.7	
SPL	46.4	70.5	77.2	61.0	65.2	73.2	64.3	44.7	79.1	69.5	58.0	79.8	65.7	
PADA	53.2	69.5	78.6	61.7	62.7	60.9	56.4	44.6	79.3	74.2	55.1	77.4	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	76.5	77.2	64.3	67.5	75.8	69.3	54.2	83.7	75.6	56.7	84.5	70.5	
SAFN	58.9	76.3	81.4	70.4	73.0	77.8	72.4	55.3	80.4	75.8	60.4	79.9	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	62.7	79.3	81.2	65.1	68.4	76.5	70.8	55.3	85.2	76.9	59.1	83.4	72.0	
RTNet _{adv}	63.2	80.1	80.7	66.7	69.3	77.2	71.6	53.9	84.6	77.4	57.9	85.5	72.3	
BA ³ US	60.6	83.2	88.4	71.8	72.8	83.4	75.5	61.6	86.5	79.3	62.8	86.1	76.0	
SLM (Ours)	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 2: **Performance on Office-Home.** We highlight the **best** and **second best** method on each task. While the upper section shows results of unsupervised domain adaptation approaches, the lower section shows results of existing partial domain adaptation methods. **SLM** achieves the best average performance among all compared methods. See supplementary material for standard deviation of each task.

Method	ImageNet-Caltech			VisDA-2017			Average
	I→C	C→I	Average	R→S	S→R	Average	
ResNet-50	69.7±0.8	71.3±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±0.5	69.3±0.4	70.6	—	—	—	
RTN	72.2	68.3	70.3	72.9	50.0	61.5	
CDAN+E	72.5±0.1	72.0±0.1	72.2	—	—	—	
PADA	75.0±0.4	70.5±0.4	72.8	76.5	53.5	65.0	
SAN	77.8±0.4	75.3±0.4	76.5	69.7	49.9		
IWAN	78.1±0.4	73.3±0.5	75.7	71.3	48.6		
ETN	83.2 ±0.2	74.9±0.4	79.1	—	—	—	
SAFN	—	—	—	—	67.7±0.5	—	
DRCN	75.3	78.9	77.1	73.2	58.2	65.7	
SLM (Ours)	82.3±0.1	81.4 ±0.6	81.9	77.5 ±0.8	91.7 ±0.8	84.6	

Table 3: **Performance on ImageNet-Caltech and VisDA-2017.** Our **SLM** performs the best on both datasets.

the best performance, outperforming the next competitive method by a margin of about **2.8%** and **18.9%** on ImageNet-Caltech and VisDA-2017 datasets respectively. Especially for task S→R on VisDA-2017 dataset, our approach significantly outperforms SAFN [50] and DRCN [22] by an increase of **24.1%** and **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, **SLM** 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.

4.3. 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 on three datasets. E.g. On Office-Home, as seen from Table 4, 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 removal of source domain outlier samples.

Effectiveness of Discrete Selection. The adverse effects of negative transfer motivated us to adopt a *strong* form of discrete selection of relevant source samples instead of a *weak* form of filtering using soft-weights as adopted in many prior works [4, 5]. In Table 5, we replaced the weighting module in PADA [4] with our stronger Select module (PADA w/ SEL) and obtained an average accuracy of 94.9% on Office31 which is 1.6% higher than the original PADA method, showing the superior selection of our Select module. We also adopted the class-weighting (γ) scheme from PADA [4] and used thresholding on top of it (**SLM** w/ W+T) to replace the Select module in **SLM** and obtained 94.0% average accuracy. The 4.4% drop shows the ineffectiveness of filtering outliers solely based on the target predictions and the importance of having a dedicated Select module which takes decisions as a function of the source samples.

Comparison with Varying Number of Target Classes. We compare different methods by varying the target label space. In Figure 4, **SLM** consistently obtains the best results indicating its advantage in alleviating negative transfer by removing outlier source samples. **SLM** outperforms all the

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	Rw → Pr	Rw → Cl	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	77.1	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 4: **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.

Office31									
Method	A → W	D → W	W → D	A → D	D → A	W → A	Average		
PADA	86.3	99.3	100.0	90.4	91.3	92.6	93.3		
PADA w/ SEL	91.8	99.3	96.6	93.8	94.2	93.5	94.9		
SLM w/ W+T	90.8	99.7	98.7	93.0	91.8	90.2	94.0		
SLM	99.8	100.0	99.8	98.7	96.1	95.9	98.4		

Table 5: Effectiveness of Discrete Selection.

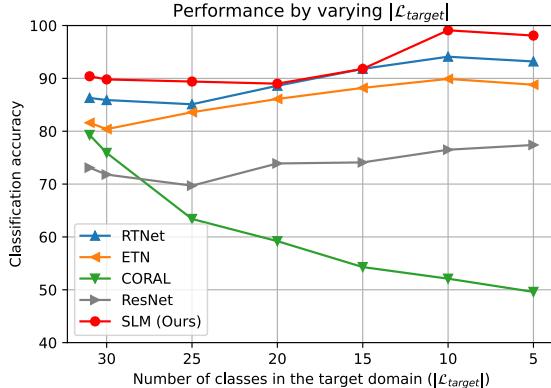


Figure 4: Performance by varying the number of target classes on A → W task from Office31 dataset. Best viewed in color.

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 Different Mixup. We examine the effect of mixup regularization on both domain discriminator and classifier on Office-Home dataset. 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, it 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. We also explored CutMix [52] as an alternative mixing strategy, which can result with new images with information from both the domains are also expected to work well resonating with our motivation. In Table 6, we replace MixUp with CutMix (**SLM** w/ CutMix) in the Mix module and obtain an average accuracy of 98.0% on Office31 almost similar to that using MixUp (98.4%). We also tried adding CutMix in addition to MixUp (**SLM** w/ MixUp+Cutmix) and obtain a similar value of 98.4%, with a slight improvement in W→D task.

Office31							
Method	A → W	D → W	W → D	A → D	D → A	W → A	Average
SLM w/ CutMix	99.6	100.0	100.0	97.4	95.7	95.5	98.0
SLM w/ MixUp	99.8	100.0	99.8	98.7	96.1	95.9	98.4
SLM w/ MixUp+CutMix	99.8	100.0	100.0	98.7	96.1	95.9	98.4

Table 6: Performance using CutMix.

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

Table 7: **Wasserstein Distance between Domains.** Table shows values for two randomly sampled tasks from **Office-31** and **Office-Home**. The values are normalized by assuming the distance for dist(S_{all}, T) to be equal to 1.000, where S_{all} represents all source samples for the corresponding tasks.

Distance between Domains. Following [9], we compute the Wasserstein distance between the probability distribution of target samples (T) with that of selected (S_{sel}) and discarded samples (S_{dis}) by the selector network. Table 7 shows that dist(S_{sel}, T) is smaller than dist(S_{all}, T), while dist(S_{dis}, T) is greater than dist(S_{all}, T) on two randomly sampled adaptation tasks from Office31 and Office-Home. The results affirm that samples selected by our selector network are closer to target domain while the discarded samples are very dissimilar to the target domain.

Additional Ablation Analysis. We provide additional ablation analyses including effect of Hausdorff distance, soft pseudo-labels, results with different backbones, feature visualizations using t-SNE [28], etc. in the Appendix.

5. 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.

Acknowledgement. This work is supported by the SERB Grant SRG/2019/001205 and the Defense Advanced Research Projects Agency (DARPA) under Contract No. FA8750-19-C-1001. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Defense Advanced Research Projects Agency.

References

- [1] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. In *Advances in Neural Information Processing Systems*, pages 5049–5059, 2019.
- [2] Silvia Bucci, Antonio D’Innocente, and Tatiana Tommasi. Tackling partial domain adaptation with self-supervision. In *International Conference on Image Analysis and Processing*, pages 70–81. Springer, 2019.
- [3] Zhangjie Cao, Mingsheng Long, Jianmin Wang, and Michael I Jordan. Partial transfer learning with selective adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2724–2732, 2018.
- [4] Zhangjie Cao, Lijia Ma, Mingsheng Long, and Jianmin Wang. Partial adversarial domain adaptation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 135–150, 2018.
- [5] Zhangjie Cao, Kaichao You, Mingsheng Long, Jianmin Wang, and Qiang Yang. Learning to transfer examples for partial domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2985–2994, 2019.
- [6] Woong-Gi Chang, Tackgeun You, Seonguk Seo, Suha Kwak, and Bohyung Han. Domain-specific batch normalization for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7354–7362, 2019.
- [7] Chao Chen, Zhihong Chen, Boyuan Jiang, and Xinyu Jin. Joint domain alignment and discriminative feature learning for unsupervised deep domain adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3296–3303, 2019.
- [8] Jin Chen, Xinxiao Wu, Lixin Duan, and Shenghua Gao. Domain adversarial reinforcement learning for partial domain adaptation. *arXiv preprint arXiv:1905.04094*, 2019.
- [9] Zhihong Chen, Chao Chen, Zhaowei Cheng, Boyuan Jiang, Ke Fang, and Xinyu Jin. Selective transfer with reinforced transfer network for partial domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12706–12714, 2020.
- [10] Gabriela Csurka. Domain adaptation for visual applications: A comprehensive survey. *arXiv preprint arXiv:1702.05374*, 2017.
- [11] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*, pages 1180–1189. PMLR, 2015.
- [12] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030, 2016.
- [13] Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *The Journal of Machine Learning Research*, 13(1):723–773, 2012.
- [14] Gregory Griffin, Alex Holub, and Pietro Perona. Caltech-256 object category dataset. 2007.
- [15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [16] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *International conference on machine learning*, pages 1989–1998. PMLR, 2018.
- [17] Jian Hu, Hongya Tuo, Chao Wang, Lingfeng Qiao, Haowen Zhong, and Zhongliang Jing. Multi-weight partial domain adaptation. In *BMVC*, page 5, 2019.
- [18] Lanqing Hu, Meina Kan, Shiguang Shan, and Xilin Chen. Duplex generative adversarial network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1498–1507, 2018.
- [19] Naoto Inoue, Ryosuke Furuta, Toshihiko Yamasaki, and Kiyoharu Aizawa. Cross-domain weakly-supervised object detection through progressive domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5001–5009, 2018.
- [20] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.
- [21] Guoliang Kang, Lu Jiang, Yi Yang, and Alexander G Hauptmann. Contrastive adaptation network for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4893–4902, 2019.
- [22] Shuang Li, Chi Harold Liu, Qiuxia Lin, Qi Wen, Limin Su, Gao Huang, and Zhengming Ding. Deep residual correction network for partial domain adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.
- [23] Jian Liang, Dapeng Hu, and Jiashi Feng. Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. *arXiv preprint arXiv:2002.08546*, 2020.
- [24] Jian Liang, Yunbo Wang, Dapeng Hu, Ran He, and Jiashi Feng. 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*, pages 123–140. Springer, 2020.
- [25] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning Transferable Features with Deep Adaptation Networks. In *International conference on machine learning*, pages 97–105. PMLR, 2015.
- [26] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Conditional Adversarial Domain Adaptation. In *Neural Information Processing Systems*, pages 1640–1650, 2018.
- [27] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Unsupervised domain adaptation with residual transfer networks. In *Advances in neural information processing systems*, pages 136–144, 2016.

- [28] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.
- [29] Chris J Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. *arXiv preprint arXiv:1611.00712*, 2016.
- [30] Xudong Mao, Yun Ma, Zhenguo Yang, Yangbin Chen, and Qing Li. Virtual mixup training for unsupervised domain adaptation. *arXiv preprint arXiv:1905.04215*, 2019.
- [31] Ke Mei, Chuang Zhu, Jiaqi Zou, and Shanghang Zhang. Instance adaptive self-training for unsupervised domain adaptation. *arXiv preprint arXiv:2008.12197*, 2020.
- [32] Rafael Müller, Simon Kornblith, and Geoffrey E Hinton. When does label smoothing help? In *Advances in Neural Information Processing Systems*, pages 4694–4703, 2019.
- [33] Zak Murez, Soheil Kolouri, David Kriegman, Ravi Ramamoorthi, and Kyungnam Kim. Image to image translation for domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4500–4509, 2018.
- [34] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*, pages 8026–8037, 2019.
- [35] Zhongyi Pei, Zhangjie Cao, Mingsheng Long, and Jianmin Wang. Multi-adversarial domain adaptation. *arXiv preprint arXiv:1809.02176*, 2018.
- [36] Xingchao Peng, Ben Usman, Neela Kaushik, Judy Hoffman, Dequan Wang, and Kate Saenko. Visda: The visual domain adaptation challenge. *arXiv preprint arXiv:1710.06924*, 2017.
- [37] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.
- [38] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *European conference on computer vision*, pages 213–226. Springer, 2010.
- [39] Kuniaki Saito, Yoshitaka Ushiku, and Tatsuya Harada. Asymmetric tri-training for unsupervised domain adaptation. *arXiv preprint arXiv:1702.08400*, 2017.
- [40] Jian Shen, Yanru Qu, Weinan Zhang, and Yong Yu. Wasserstein distance guided representation learning for domain adaptation. *arXiv preprint arXiv:1707.01217*, 2017.
- [41] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.
- [42] Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *European conference on computer vision*, pages 443–450. Springer, 2016.
- [43] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7167–7176, 2017.
- [44] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5018–5027, 2017.
- [45] Vikas Verma, Alex Lamb, Christopher Beckham, Amir Nafaji, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. Manifold mixup: Better representations by interpolating hidden states. In *International Conference on Machine Learning*, pages 6438–6447. PMLR, 2019.
- [46] Mei Wang and Weihong Deng. Deep visual domain adaptation: A survey. *Neurocomputing*, 312:135–153, 2018.
- [47] Qian Wang and Toby Breckon. Unsupervised domain adaptation via structured prediction based selective pseudo-labeling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 6243–6250, 2020.
- [48] Yuan Wu, Diana Inkpen, and Ahmed El-Roby. Dual mixup regularized learning for adversarial domain adaptation. In *European Conference on Computer Vision*, pages 540–555. Springer, 2020.
- [49] Minghao Xu, Jian Zhang, Bingbing Ni, Teng Li, Chengjie Wang, Qi Tian, and Wenjun Zhang. Adversarial domain adaptation with domain mixup. *arXiv:1912.01805*, 2019.
- [50] Ruijia Xu, Guanbin Li, Jihan Yang, and Liang Lin. Larger norm more transferable: An adaptive feature norm approach for unsupervised domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1426–1435, 2019.
- [51] Shen Yan, Huan Song, Nanxiang Li, Lincan Zou, and Liu Ren. Improve unsupervised domain adaptation with mixup training. *arXiv preprint arXiv:2001.00677*, 2020.
- [52] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6023–6032, 2019.
- [53] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.
- [54] Jing Zhang, Zewei Ding, Wanqing Li, and Philip Ogunbona. Importance weighted adversarial nets for partial domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8156–8164, 2018.
- [55] Yabin Zhang, Bin Deng, Kui Jia, and Lei Zhang. Label propagation with augmented anchors: A simple semi-supervised learning baseline for unsupervised domain adaptation. In *European Conference on Computer Vision*, pages 781–797. Springer, 2020.
- [56] Yang Zou, Zhiding Yu, Xiaofeng Liu, BVK Kumar, and Jin-song Wang. Confidence regularized self-training. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5982–5991, 2019.
- [57] Yang Zou, Zhiding Yu, BVK Vijaya Kumar, and Jinsong Wang. Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In *Proceedings of the European conference on computer vision (ECCV)*, pages 289–305, 2018.

A. Dataset Details

We evaluate the performance of our approach on several benchmark datasets for partial domain adaptation, namely Office31 [38], Office-Home [44], ImageNet-Caltech and VisDA-2017 [36]. 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 [14] as the target categories and obtain the following label spaces:

$$\mathcal{L}_{source} = \{0, 1, 2, \dots, 30\}.$$

$$\mathcal{L}_{target} = \{0, 1, 5, 10, 11, 12, 15, 16, 17, 22\}.$$

Number of Outlier Classes = 21.

Figure 5 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 5: **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:

$$\mathcal{L}_{source} = \{0, 1, 2, \dots, 64\}.$$

$$\mathcal{L}_{target} = \{0, 1, 2, \dots, 24\}.$$

Number of Outlier Classes = 40.

Figure 6 displays a gallery of sample images for this dataset. The dataset is publicly available to download at: <http://hemanthdv.org/OfficeHome-Dataset/>.



Figure 6: **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.



Figure 7: **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.

ImageNet-Caltech. This large-scale dataset consists of two datasets (ImageNet1K [37] (I) & Caltech256 [14] (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 7 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 [22]. 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:

$$\mathcal{L}_{source} = \{0, 1, 2, \dots, 11\}.$$

$$\mathcal{L}_{target} = \{0, 1, 2, \dots, 5\}.$$

Number of Outlier Classes = 6.



Figure 8: **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.

Figure 8 displays a gallery of sample images for this dataset. The dataset is publicly available to download at:
<http://ai.bu.edu/visda-2017/#download>.

B. 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:

Feature Extractor (\mathcal{G}). We use ResNet-50 [15] 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 [37] 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 [6] 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 [15] 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 [4, 5], 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.

Algorithm 1 The training pipeline for **SLM**

Data: source data \mathcal{D}_{source} and target data \mathcal{D}_{target} .

Networks: Selector Network $\mathcal{H}(\cdot)$, Feature Extractor $\mathcal{G}(\cdot)$, Classifier $\mathcal{F}(\cdot)$, and Domain Discriminator $\mathcal{D}(\cdot)$.

- 1: Initialize networks $\mathcal{G}(\cdot)$, $\mathcal{F}(\cdot)$, $\mathcal{H}(\cdot)$, and $\mathcal{D}(\cdot)$ in **SLM**.
 - 2: **for** $itrn = 1 \rightarrow num_itrn$ **do**
 - 3: Obtain the mini-batches \mathcal{D}_{source}^b and \mathcal{D}_{target}^b .
“Select” Module
 - 4: Obtain the binary decisions from $\mathcal{H}(\mathcal{D}_{source}^b)$ & use them to obtain \mathcal{D}_{sel}^b & \mathcal{D}_{dis}^b .
“Label” Module
 - 5: Obtain soft pseudo-labels \hat{y}^b from $\mathcal{F}(\mathcal{G}(\mathcal{D}_{target}))$ for \mathcal{D}_{target} .
“Mix” Module
 - 6: Obtain $\mathcal{D}_{inter_mix}^b$, $\mathcal{D}_{intra_mix_s}^b$, and $\mathcal{D}_{intra_mix_t}^b$.
 - 7: Compute \mathcal{L}_{sup} , \mathcal{L}_{adv} , \mathcal{L}_{select} , \mathcal{L}_{label} , and \mathcal{L}_{mix} .
 - 8: Compute the gradients & backpropagate for optimization using gradient descent.
 - 9: **end for**
-

C. Additional Experimental Results

Results on Office-Home Dataset. In Table 8, along with the performance accuracies, we have included the standard deviation for each adaptation task for the Office-Home dataset, as promised in Table-2 of the main paper.

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 [41] backbone for the feature extractor. We report the results on the transfer tasks from the Office31 dataset in Table 9 and compare it with the current state-of-the-art methods. Our method outperforms the previously best results by a margin of **3.0%** on average and achieves new state-of-the-art results. This confirms that our proposed

Method	Office-Home												Average
	Ar→Cl	Ar→Pr	Ar→Rw	Cl→Ar	Cl→Pr	Cl→Rw	Pr→Ar	Pr→Cl	Pr→Rw	Rw→Ar	Rw→Cl	Rw→Pr	
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	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	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	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
RTNet _{adv}	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	85.5 \pm 0.1	72.3
BA³US	60.6 \pm 0.5	83.2 \pm 0.1	88.4 \pm 0.2	71.8 \pm 0.2	72.8 \pm 1.1	83.4 \pm 0.6	75.5 \pm 0.2	61.6 \pm 0.4	86.5 \pm 0.2	79.3 \pm 0.7	62.8 \pm 0.5	86.1 \pm 0.3	76.0

Table 8: **Performance on Office-Home.** We highlight the **best** and **second best** method on each task. While the upper section shows results of unsupervised domain adaptation approaches, the lower section shows results of existing partial domain adaptation methods. **SLM** achieves the best average performance among all compared methods.

Method	Office31						Average
	A→W	D→W	W→D	A→D	D→A	W→A	
VGG-16 [41] (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 [4] (ECCV’18)	86.1 \pm 0.4	100.0 \pm 0.0	100.0 \pm 0.0	81.7 \pm 0.3	93.0 \pm 0.2	95.3 \pm 0.3	92.5
SAN [3] (CVPR’18)	83.4 \pm 0.4	99.3 \pm 0.5	100.0 \pm 0.0	90.7 \pm 0.2	87.2 \pm 0.2	91.9 \pm 0.4	92.1
IWAN [54] (CVPR’18)	82.9 \pm 0.3	79.8 \pm 0.3	88.5 \pm 0.2	91.0 \pm 0.3	89.6 \pm 0.2	93.4 \pm 0.2	87.5
ETN [5] (CVPR’19)	85.7 \pm 0.2	100.0 \pm 0.0	100.0 \pm 0.0	89.4 \pm 0.2	95.9 \pm 0.2	92.3 \pm 0.2	93.9
SLM (Ours)	92.0 \pm 0.1	99.8 \pm 0.2	99.6 \pm 0.5	98.1 \pm 0.0	96.1 \pm 0.0	96.0 \pm 0.1	96.9

Table 9: **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.

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 Office-Home dataset. Here, we extend the experiments to Office31 and VisDA-2017 and provide the performance on the transfer tasks in Table 10. Similar to the results on Office-Home dataset, our approach with all the three modules (Select, Label and Mix) working jointly, works the best on both datasets.

Effectiveness of Hausdorff Distance. We investigate the effect of Hausdorff distance (Eqn. 2 in the main paper) in selector network training and find that removing it lowers down performance from 76.0% to 73.7% on Office-Home dataset, showing its importance in guiding the selector to discard the outlier source samples for effective reduction in negative transfer. We provide the individual performance of all the transfer tasks on Office-Home dataset in Table 11, which shows that our approach with Hausdorff distance loss

works the best in all cases.

Effectiveness of Soft Pseudo-Labels. We also test the effectiveness of soft pseudo-labels by replacing them with 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. We provide the performance on each of the transfer tasks from Office-Home in Table 12.

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 13.

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 10: **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.

	Ar → Cl	Ar → Pr	Ar → Rw	Cl → Ar	Cl → Pr	Cl → Rw	Pr → Ar	Pr → Cl	Pr → Rw	Rw → Ar	Rw → 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 11: **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	Rw → 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 12: **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	Rw → 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 13: **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 classifier (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.

D. Qualitative Results

Feature Visualizations. We use t-SNE [28] to visualize the features learned using different components of our **SLM** framework. We choose an UDA setup (similar to DANN [12]) 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 9, the feature space for vanilla setup lacks discriminability 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.

E. Broader Impact and Limitations

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 [37]) 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. We also believe our approach of selecting relevant source data would motivate the research community to extend it to various open-world problems and would help in training more generalizable models. 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 interpretability.

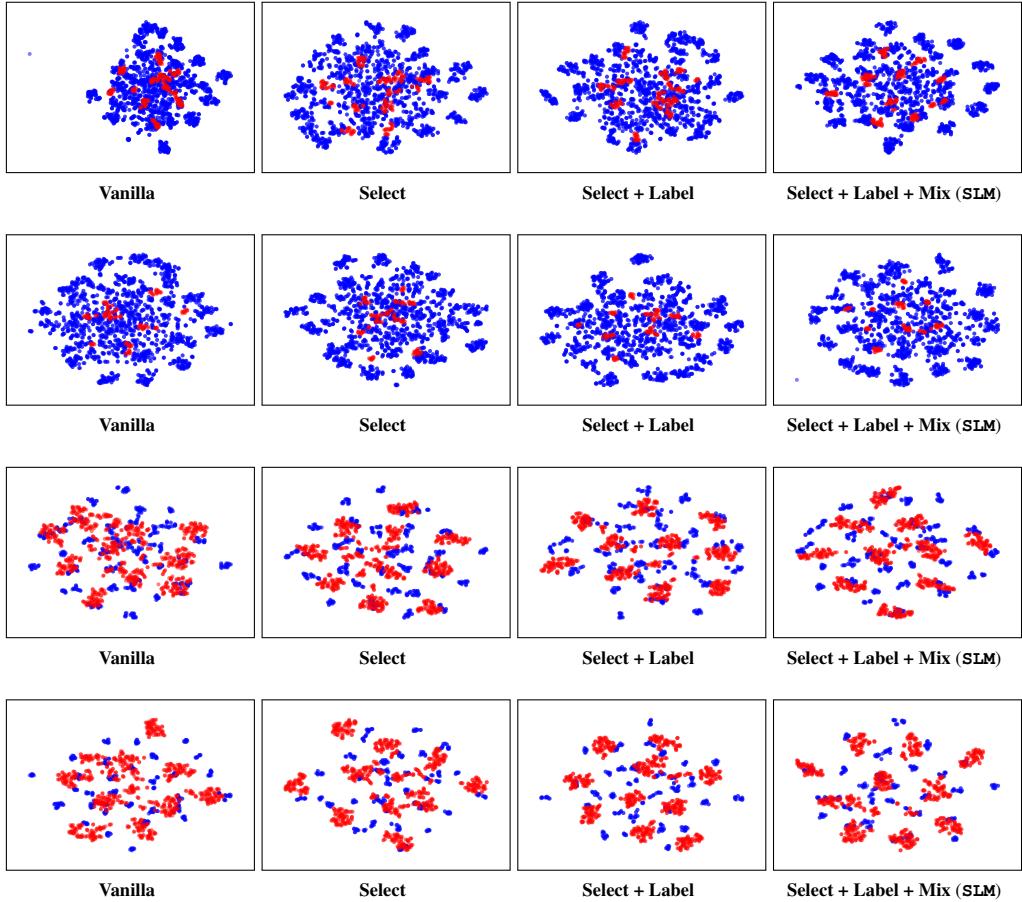


Figure 9: Feature Visualizations using t-SNE. Plots show visualization of our approach with different modules on $A \rightarrow W$, $A \rightarrow D$, $W \rightarrow A$, and $D \rightarrow A$ tasks respectively (top to down) from Office31 dataset. Blue and red dots represent source and target data respectively. 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.