# Source Data-absent Unsupervised Domain Adaptation through Hypothesis Transfer and Labeling Transfer

Jian Liang, Dapeng Hu, Yunbo Wang, Ran He, *Senior Member, IEEE,* Jiashi Feng, *Member, IEEE*

**Abstract**—Unsupervised domain adaptation (UDA) aims to transfer knowledge from a related but different well-labeled source domain to a new unlabeled target domain. Most existing UDA methods require access to the source data, and thus are not applicable when the data are confidential and not shareable due to privacy concerns. This paper aims to tackle a realistic setting with only a classification model available trained over, instead of accessing to, the source data. To effectively utilize the source model for adaptation, we propose a novel approach called Source HypOthesis Transfer (SHOT), which learns the feature extraction module for the target domain by fitting the target data features to the frozen source classification module (representing classification hypothesis). Specifically, SHOT exploits both information maximization and self-supervised learning for the feature extraction module learning to ensure the target features are implicitly aligned with the features of unseen source data via the same hypothesis. Furthermore, we propose a new labeling transfer strategy, which separates the target data into two splits based on the confidence of predictions (labeling information), and then employ semi-supervised learning to improve the accuracy of less-confident predictions in the target domain. We denote labeling transfer as SHOT++ if the predictions are obtained by SHOT. Extensive experiments on both digit classification and object recognition tasks show that SHOT and SHOT++ achieve results surpassing or comparable to the state-of-the-arts, demonstrating the effectiveness of our approaches for various visual domain adaptation problems. Code will be available at https://github.com/tim-learn/SHOT-plus.

**Index Terms**—Unsupervised domain adaptation, transfer learning, self-supervised learning, semi-supervised learning, model reuse.

## 1 Introduction

Deep neural networks have achieved remarkable success in a variety of applications across different fields but at the expense of laborious large-scale training data annotation. To avoid expensive data labeling, transfer learning [1], [2], [3] is developed to extract the knowledge from one or more source tasks which is then applied to a target task. As a typical example, unsupervised domain adaptation (UDA) tackles the problem setting where the learning task in the source domain is sufficiently similar or the same as that in the target domain but labeled data are only available in the source domain during training. Recently, UDA methods have been widely applied to boost performance of many tasks like object recognition [2], [4], [5], [6], semantic segmentation [3], [7], [8], [9], sentiment classification [10], [11], object detection [12], [13], and person re-identification [14], [15]. Existing UDA methods mainly follow two paradigms to mitigate the gap between source and target domains. The first paradigm matches the statistical moments of different feature distributions at different orders to minimize the distributional divergence between domains [16], [17], [18]. For example, the widely used Maximum Mean Discrepancy (MMD) [19] measure minimizes the distance between weighted sums of all moments from the source and target domains. The second paradigm applies adversarial learning [20] with an additional domain classifier to minimize the Proxy $\mathcal{A}$-distance [21] between the domains. All these methods require to access the source data during learning to adapt the model to the target domain.

However, nowadays the data often involves user private information, e.g., those on personal phones or from hospital records. Recently, several data protection frameworks have been proclaimed by the European Union (EU) and some governments, among which the General Data Protection Regulation (GDPR), as a typical example, highlights the safety issue of data transfer. Accordingly, it may violate the data privacy policy for previous UDA methods to access the source data during learning to adapt. To alleviate this issue in the transfer learning field, Hypothesis Transfer Learning (HTL) [22] explore to retain prior knowledge in a form of hypotheses instead of training data inherited from previous tasks. Likewise, in this paper, we introduce a *realistic but challenging* source data-absent UDA setting [23] with only a well-trained source model provided as supervision. Different from HTL, here we do not have any labeled data in the target domain for the UDA problem. Our introduced setting also differs from vanilla UDA in that the source model instead of the source data is provided to the target domain for adaptation, making the cross-domain feature-level distribution matching challenging.

To address this UDA setting, we propose a novel approach called *Source HypOthesis Transfer* (SHOT). SHOT follows common deep UDA methods [4], [24] to utilize

- *J. Liang and R. He are with National Laboratory of Pattern Recognition (NLPR), Institute of Automation, Chinese Academy of Sciences (CASIA), Beijing, China, 100190. E-mail: liangjian92@gmail.com, rhe@nlpr.ia.ac.cn.*
- *D. Hu and J. Feng are with the Department of Electrical and Computer Engineering, National University of Singapore (NUS), Singapore, 119077. E-mail: dapeng.hu@u.nus.edu, elefjia@nus.edu.sg.*
- *Y. Wang is with Wangxuan Institute of Computer Technology, Peking University, Beijing, China, 100871. Email: wangyunbo09@gmail.com.*

*Manuscript received October 22, 2020; revised May 28, 2021.*

an identical network architecture for different domains, consisting of a feature encoding module and a classification module (hypothesis). Like [5], [25], SHOT aims to learn a target-specific feature encoding module to generate target data representations that are well aligned with source data representations, but without accessing the source data or the target data labels. Intuitively, if the learned target data representations are aligned with the source ones, their classification results from the fixed source classifier (hypothesis) would be highly confident for a certain class, i.e., the classification outputs being close to one-hot vectors. We are then motivated to make SHOT adapt the feature encoding module by fine-tuning the source feature encoding module while freezing the source hypothesis, to maximize the mutual information between intermediate feature representations and outputs of the classifier, since information maximization [26], [27] can encourage the classifier to assign disparate one-hot outputs to different target feature representations.

Though target feature representations are encouraged to fit the source hypothesis via information maximization, some semantically wrong matching between target feature representations and source hypothesis may still occur, leading to wrong labels assigned to the target data. To alleviate this, we propose to fully exploit the knowledge in the unlabeled target domain by developing two new self-supervised learning schemes. First, considering pseudo labels generated by the source classifier for the target data may be noisy, we propose to attain per-class prototype representations for the target domain itself and apply the nearest prototype classifier to obtain more accurate pseudo labels as direct supervision. Secondly, inspired by RotNet [28] that predicts the absolute rotation of a rotated image, we come up with a relative rotation prediction task to capture the image-specific self-supervision more precisely, i.e. requiring the model to estimate the relative rotation between one original image and its rotated version. The two self-supervisions are used to help discard irrelevant semantic information by exploiting the data distribution of the target domain, thus helping learn feature representations that better fit the source hypothesis. In this way, we obtain a target-specific feature encoding module with the source hypothesis as the shared classifier module across domains.

Since some low-confident predictions generated with the proposed *hypothesis transfer* strategy are possibly inaccurate, we further put forward a *labeling transfer* strategy as a following step, forming a complete two-stage framework called SHOT++ for UDA problems. Particularly, we sort the the confidence of the adapted predictions after SHOT and discover an adaptive threshold to automatically divide the whole target data into two splits, i.e., 'easy' split with high confidence and 'hard' split with low confidence. Empirically, these predictions of samples in the 'easy' split are reliable. Thus, we employ a popular semi-supervised learning algorithm, MixMatch [29], to enable the reliable labeling information from the 'easy' split to flow to the 'hard' split in the target domain itself. It is worth noting that such a labeling transfer strategy can also be applied to the original source model, or even a black-box predictor without knowing the network architecture.

Experimental results on multiple benchmark datasets clearly demonstrate the proposed SHOT and SHOT++ obtain competitive results with the state-of-the-art, or outperform the state-of-the-art for three different UDA cases, i.e., closed-set [30], partial-set [31], multi-source [18] problems. The superior results over prior arts in a semi-supervised domain adaptation (SSDA) scenario [32] further verify the versatility of the proposed methods. The main contributions of this work are summarized as follows.

- We propose a novel framework, Source HypOthesis Transfer (SHOT), for unsupervised domain adaptation with only the source model provided, which is appealing for privacy protection without access to the source data.
- SHOT exploits information maximization to learn a target-specific feature encoding module, which provides an implicit perspective on feature alignment.
- SHOT further exploits the knowledge in the unlabeled target domain by developing two new kinds of self-supervisions as auxiliary tasks, which further improves the adaptation performance.
- We further propose a new labeling transfer strategy by exploiting the confidence of predictions and enforcing the labeling information to flow from 'easy' samples to 'hard' samples, even allowing adaptation with a *black-box* source model.
- Experiments on several benchmarks demonstrate our methods yield results comparable to or outperforming the state-of-the-arts for three unsupervised domain adaptation scenarios and even semi-supervised domain adaptation.

This paper extends our earlier work [23] in the following aspects. Within the hypothesis transfer framework developed in [23], we additionally propose one more self-supervision objective to predict the relative rotation, which facilitates learning semantically meaningful representations in the target domain. We also propose a new strategy named labeling transfer that only requires the labeling predictions in the target domain. Different from [23], it even allows adaptation with a *black-box* source model. Besides, it can be incorporated with the hypothesis transfer framework, yielding better adaptation results. We also expand the experimental evaluation by adding more datasets for each UDA scenario (e.g., PACS [33] for multi-source UDA) and extending our methods further to semi-supervised domain adaptation. Finally, we provide a more detailed model analysis to evaluate the proposed approaches, including training stability, parameter sensitivity and qualitative study.

## 2 Related Work

### 2.1 Unsupervised Domain Adaptation

As a typical example of transfer learning [1], unsupervised domain adaptation (UDA) aims to exploit the knowledge in a different but related labeled dataset to help learn a discriminative model for the unlabeled dataset. Early UDA methods [34], [35] assume the *covariate shift* with the identical conditional distributions across domains and approximate the target empirical risk by estimating the weight of each source instance and re-weighting the source empirical risk. Later, most UDA methods resort to domain-invariant feature transformation [36], [37], [38] or feature

space alignment [16], [39], [40] to pursue distribution alignment. However, the transferability of these shallow methods is restricted by task-specific structures [41].

Recently, deep neural networks are well explored to learn transferable representations for domain adaptation, in various visual applications like object recognition [2], [39], [42] and semantic segmentation [3], [8], [9], [43]. Based on the relationship of label spaces between source and target domains, UDA scenarios can be categorized into four cases, i.e., closed-set [30], partial-set [31], open-set [44], and universal [45]. Among them, the closed-set UDA has received the most research attention, where the source and target label spaces are assumed to be identical. Existing deep closed-set UDA methods can be roughly divided into three distinct categories: discrepancy-based, reconstruction-based, and adversarial-based. Discrepancy-based approaches minimize a divergence criterion that measures the distance between the source and target data distributions, and some favoring choices include maximum mean discrepancy (MMD) [4], high-order central moment discrepancy [17], contrastive domain discrepancy [46], and the Wasserstein metric [47]. Reconstruction-based approaches like [48] utilize reconstruction as an auxiliary task to pursue shared representations for both domains. In addition, some other reconstruction-based methods [49], [50] further seek domain-specific reconstruction and cycle consistency to improve the adaptation performance. Inspired by generative adversarial nets [20], adversarial-based approaches determine the distance between different data distributions based on binary classification performance, which in effect corresponds to the Proxy $\mathcal{A}$-distance or $\mathcal{H}$-divergence in the seminal theoretical framework [21]. Different from marginal distribution alignment using one binary domain classifier in [24], following methods encourage joint distribution alignment by considering multiple class-wise domain classifiers [51] or a semantic multi-output classifier [52], [53] instead of a feature-conditional domain discriminator [42], respectively. There are also some other studies investigating batch normalization [54], [55] and adversarial dropout [56], [57] within the network architecture to ensure feature invariance. Despite their efficacy, all these methods assume the target user's access to the source domain, which is not unpractical since the source data may be private and confidential.

### 2.2 Hypothesis Transfer Learning

The concept of hypothesis transfer learning (HTL) is first presented by Kuzborskij and Orabona [22], also with a formal theory. Before it, there are a number of transfer learning works [58], [59], [60] that assume no explicit access to the source data and are empirically successful. Generally, HTL is an attractive and efficient framework that assumes access to a given number of source hypotheses and a small set of training samples from the target domain. However, like the famous fine-tuning strategy [61], HTL always requires at least a small set of labeled data in the target domain, limiting its applicability to the semi-supervised DA scenario. Inspired by HTL, several recent works [62], [63] assume absence of the source data and utilize the encoded information as source supervision for the UDA problem. In particular, besides target features, [62] requires predictions of target data, and [63] requires the mean and variance per-class calculated on source features. Both methods adopt a shallow framework like HTL, which are restricted to the original feature structure. By contrast, our work fully exploits the end-to-end feature learning module, allowing more flexibility during adaptation. There are also two concurrent deep UDA methods [64], [65] that attempt not to access the source data during the adaptation process. Our approach differs from [64] as we do not need any additional components like a data generator or classifier within the training algorithm; [65] introduces the first federated DA setting where knowledge is transferred from the decentralized nodes to a new node without any supervision itself and proposes an adversarial-based solution to protect user privacy, but it may fail to tackle the vanilla UDA setting with only one source domain available.

### 2.3 Self-supervised Learning

Self-supervised learning [66] offers great feasibility for effectively utilizing unlabeled data by generating and predicting labels from these data. The self-supervised task is also known as pretext task. A typical workflow[1] is to train a model on one or multiple pretext tasks with unlabeled images and then fine-tune the trained model on a variety of practical downstream tasks. In addition, pretext tasks can also be jointly trained with supervised learning tasks on labeled data with shared weights like in [67], [68]. Generally, self-supervised methods involve two aspects: pretext task and loss function. Some popular image-specific self-supervision tasks include example colorization [69], relative position prediction [70], rotation prediction [28], solving jigsaw puzzles [71]; on the other hand, contrastive losses [72], [73] and clustering losses [74], [75] focus on the similarity of sample pairs in the representation space, which always provide better performance. Some recent studies [76], [77], [78] explore self-supervision for UDA problems and find it beneficial to accomplishing domain alignment. By contrast, this paper elegantly designs two different kinds of self-supervisions for UDA problems.

### 2.4 Semi-supervised Learning

When the domain shift does not exist, the UDA problem naturally becomes a well-studied semi-supervised learning problem. Many ideas originally proposed for semi-supervised learning thus can also be employed to achieve or compensate domain alignment within UDA methods. Pseudo-labeling [79] is a simple heuristic widely used in practice, which produces 'pseudo-labels' for unlabeled data using the prediction function itself during the course of training. Among UDA methods, [80] directly incorporates pseudo-labeling as a regularization term, and [42] leverages pseudo labels in the adaptation module to achieve multi-modal distribution alignment. Entropy minimization [81] is a popular strategy that encourages the network to make 'confident' (low-entropy) predictions for all unlabeled data, which has been exploited in many previous UDA methods [41], [82]. Other favored semi-supervised techniques like tri-training and virtual adversarial training have been used in

1. https://cutt.ly/DfN3rFU

frameworks [83], [84], respectively. Recently, [85] directly employs MixMatch [29] and obtains promising results in the VisDA-2019 challenge. Different from prior works that treat the whole target domain as an unlabeled dataset, we focus on intra-domain semi-supervised learning where the labeled dataset consists of confident target data samples and the unlabeled dataset consists of remaining samples.

## 3 METHOD

We aim to address the UDA problem with only a pre-trained source model, not requiring to access the source data. In particular, we consider the $K$-way visual classification task. For a vanilla UDA task, we are given $n_s$ labeled samples $\{x_s^i, y_s^i\}_{i=1}^{n_s}$ from the source domain $\mathcal{D}_s$ where $x_s^i \in \mathcal{X}_s$, $y_s^i \in \mathcal{Y}_s$, and also $n_t$ unlabeled samples $\{x_t^i\}_{i=1}^{n_t}$ from the target domain $\mathcal{D}_t$ where $x_t^i \in \mathcal{X}_t$. The goal of UDA is to predict the labels $\{y_t^i\}_{i=1}^{n_t}$ in the target domain, where $y_t^i \in \mathcal{Y}_t$, and the source task $\mathcal{X}_s \to \mathcal{Y}_s$ is assumed to be the same with the target task $\mathcal{X}_t \to \mathcal{Y}_t$. In this work, we aim to learn a target function $f_t : \mathcal{X}_t \to \mathcal{Y}_t$ and infer $\{y_t^i\}_{i=1}^{n_t}$, with only $\{x_t^i\}_{i=1}^{n_t}$ and the source function $f_s : \mathcal{X}_s \to \mathcal{Y}_s$ available.

We address the above source data-absent UDA problem through the following steps. First, we train the classification model, consisting of a feature encoding module and a hypothesis module, from the source data and then transfer the source model to the target domain without accessing the source data. Then, we present a novel framework, Source HypOthesis Transfer (SHOT), to learn the target-specific feature encoding module using self-supervised learning and semi-supervised learning, with the source hypothesis fixed. Finally, using the predictions for the target domain, we further employ a semi-supervised learning algorithm to enforce labeling information propagation from confidently labeled target samples to the remaining target samples with low confidences. Applying such a labeling transfer strategy to SHOT yields SHOT++. Likewise, applying the labeling transfer strategy to 'Source-model-only' yields 'Source-model-only++', which can even deal with a black-box source model. In the following, we elaborate on each step in details.

### 3.1 Source Model Generation

We consider learning a deep source classification model $f_s : \mathcal{X}_s \to \mathcal{Y}_s$ by minimizing the following cross-entropy loss,

$$\mathcal{L}_{src}(f_s; \mathcal{X}_s, \mathcal{Y}_s) = \mathbb{E}_{(x_s,y_s)\in\mathcal{X}_s\times\mathcal{Y}_s} \sum_{k=1}^{K} -q_k \log \delta_k(f_s(x_s)), \tag{1}$$

where $\delta_k(a) = \frac{\exp(a_k)}{\sum_i \exp(a_i)}$ denotes the $k$-th element in the soft-max output of a $K$-dimensional vector $a$, and $q$ denotes a one-hot encoding of $y_s$ where $q_k$ is '1' for the correct class and '0' for the rest. To further lift the discriminability of the source model and facilitate the following target data alignment, we adopt the label smoothing technique for model training as it encourages learned feature representations to form tight and evenly separated clusters [86], which is useful for adaptation. Therefore, the source objective function is changed to

$$\mathcal{L}_{src}^{ls}(f_s; \mathcal{X}_s, \mathcal{Y}_s) = \mathbb{E}_{(x_s,y_s)\in\mathcal{X}_s\times\mathcal{Y}_s} \sum_{k=1}^{K} -q_k^{ls} \log \delta_k(f_s(x_s)), \tag{2}$$

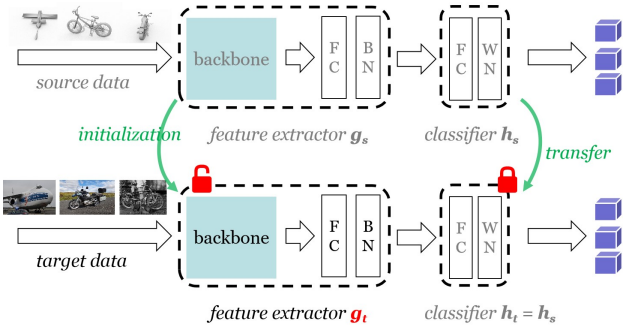


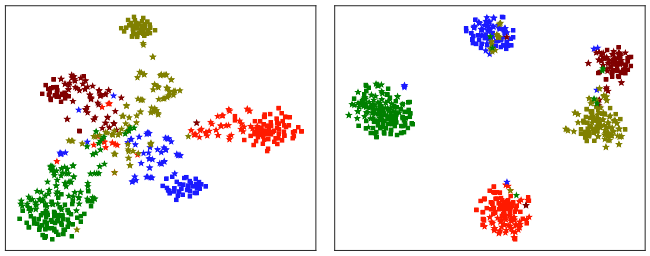

Fig. 1. The pipeline of hypothesis transfer with information maximization. The source model consists of a feature encoding module and a classifier module (hypothesis). SHOT keeps the hypothesis frozen and utilizes the feature encoding module as initialization for target domain learning.

where $q_k^{ls} = (1-\alpha)q_k + \alpha/K$ is the smoothed label and $\alpha$ is the smoothing parameter which is empirically set to 0.1.

### 3.2 Hypothesis Transfer with Information Maximization

As shown in Fig. 1, the source model parameterized by a deep neural network consists of two modules: the feature encoding module $g_s : \mathcal{X}_s \to \mathbb{R}^d$ and the classifier module $h_s : \mathbb{R}^d \to \mathbb{R}^K$, i.e., $f_s(x) = h_s(g_s(x))$, where $d$ is the dimension of the input feature. Most previous UDA methods align different domains by matching the data distributions in the feature space $\mathbb{R}^d$ using MMD [4] or domain adversarial alignment [24]. However, both strategies assume the source and target domains share the same feature encoder and need to access the source data during adaptation. This is not applicable in the tackled UDA setting here. By contrast, Adversarial Discriminative Domain Adaptation (ADDA) [5] relaxes the parameter-sharing constraint and is a new adversarial framework, which learns different mapping functions for the two domains. Also, Decision-boundary Iterative Refinement Training with a Teacher (DIRT-T) [84] first trains a parameter-sharing UDA framework as initialization and then fine-tunes the whole network by minimizing the cluster assumption violation via entropy minimization and virtual adversarial training. Both methods suggest that learning a domain-specific feature encoding module for $\mathcal{D}_t$ is practicable and even works better than the parameter-sharing mechanism, which has also been proven effective in Domain-Specific Batch Normalization (DSBN) [25].

We therefore develop a new framework termed Source HypOthesis Transfer (SHOT) by learning the domain-specific feature encoding module for the target data while fixing the source classifier module (hypothesis), as the source hypothesis encodes the distribution information of the unseen source data. Namely, SHOT utilizes the *same* classifier module $h_t = h_s$ for different domain-specific feature encoding modules. It aims to learn the optimal target feature encoding module $g_t : \mathcal{X}_t \to \mathbb{R}^d$ such that the output target features can fit the source feature distribution well and can be accurately classified by the source hypothesis directly. Note that SHOT merely utilizes the source data for just once to generate the source hypothesis, and does not need to access the source data any more, unlike prior methods (e.g., ADDA, DIRT-T, and DSBN).



(a) Source model only (b) SHOT-IM

Fig. 2. The t-SNE visualizations for a 5-way classification task. Solid '$\circ$' denotes unseen source data and '$\star$' denotes target data. Different colors represent different classes. Best viewed in colors.

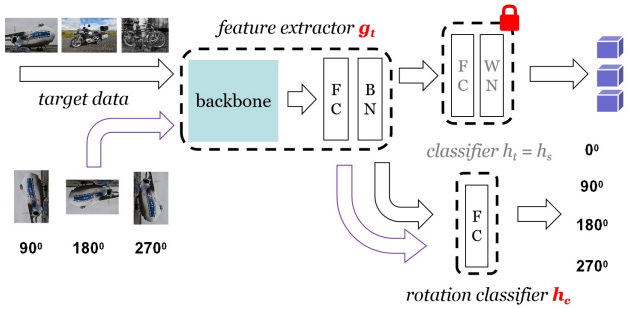



Fig. 3. The pipeline of hypothesis transfer with self-supervised learning. Besides the common target model, we impose a rotation classifier $h_c$ after the feature encoding module $g_t$. $h_c$ is parameterized by a linear classifier, which aims to predict the *relative* rotation of a target sample.

Essentially, we expect to learn the optimal target feature encoder $g_t$ so that the target data distribution $p\left(g_t(x_t)\right)$ matches the source data distribution $p\left(g_s(x_s)\right)$ well. However, feature-level alignment does not work at all since it is impossible to estimate the distribution of $p\left(g_s(x_s)\right)$ without access to the source data. We view the challenging problem from another perspective: *if there is no domain gap, what kind of outputs should be generated over the unlabeled target data?* We argue the ideal outputs of target features should be similar to those of source features with the classifier shared for both domains. Since we train the source feature encoding module $g_s$ and classifier module $h_s$ via a supervised learning loss, the output of each source feature is fairly similar to one of the one-hot encodings. Therefore, we expect that the output of each target feature through $h_t = h_s$ is also similar to one of the one-hot encodings. Such an output alignment requirement is a necessary condition for feature alignment.

For this purpose, we adopt the information maximization (IM) loss [26], [27], [87] to make the classification outputs of target features individually certain and globally diverse. In practice, we minimize the following $\mathcal{L}_{ent}$ and $\mathcal{L}_{div}$ that together constitute the IM loss ($\beta = 1$):

$$\begin{aligned}
\mathcal{L}_{im}(f_t; \mathcal{X}_t) &= \mathcal{L}_{ent}(f_t; \mathcal{X}_t) + \beta \mathcal{L}_{div}(f_t; \mathcal{X}_t) \\
\mathcal{L}_{ent}(f_t; \mathcal{X}_t) &= -\mathbb{E}_{x \in \mathcal{X}_t} \sum\nolimits_{k=1}^{K} \delta_k(f_t(x)) \log \delta_k(f_t(x)), \\
\mathcal{L}_{div}(f_t; \mathcal{X}_t) &= \sum\nolimits_{k=1}^{K} \hat{p}_k \log \hat{p}_k \\
&= D_{KL}(\hat{p}, \frac{1}{K}\mathbf{1}_K) - \log K,
\end{aligned} \tag{3}$$

where $f_t(x) = h_t(g_t(x))$ is the $K$-dimensional output of each target sample, $\mathbf{1}_K$ is a $K$-dimensional vector with all ones, and $\hat{p} = \mathbb{E}_{x \in \mathcal{X}_t}[\delta(f_t(x))]$ is the mean output embedding of the whole target domain. The IM loss would work better than conditional entropy minimization [81] widely used in prior UDA methods [32], [88] since IM can circumvent the trivial solution where all unlabeled data have the same one-hot encoding via the fair diversity-promoting objective $\mathcal{L}_{div}$. For convenience, we denote SHOT with the information maximization loss as SHOT-IM.

### 3.3 Hypothesis Transfer with Self-supervised Learning

Fig. 2 shows the t-SNE visualizations of features for a 5-way classification task learned by SHOT-IM and the 'source model only' method. Intuitively, the target feature representations are distributed in a mess for the 'source model only' method in Fig. 2(a), and using the IM loss indeed helps align the target data with the unseen source data well. However, the target data may be matched to the wrong source hypothesis to some extent in Fig. 2(b).

We argue that the harmful effects result from the inaccurate original network outputs. For instance, a target sample from the second class with the normalized network output [0.4, 0.3, 0.1, 0.1, 0.1] may be forced to have an expected output [1.0, 0.0, 0.0, 0.0, 0.0]. Motivated by [74], [89], self-supervised learning helps focus on semantically meaningful features, which is in line with domain invariant learning. Therefore, we try to learn structure-aware and semantic representations in the unlabeled target domain to alleviate such effects. Specifically, we develop two new self-supervision objectives to be jointly trained with the main unsupervised task in Eq. (3) in a similar manner to prior methods [68], [77]. We first exploit self-supervision from the perspective of the loss function and design a novel self-supervised pseudo-labeling strategy. Different from pseudo-labeling [79] where pseudo labels conventionally generated by source hypotheses are still noisy due to domain shift, our self-supervised version considers the structure of the target domain (i.e. the target-specific prototypes) and is able to provide accurate pseudo labels. The detailed learning procedure is provided in the following.

- We first attain prototype representation (centriods) for each class in the target domain, similar to weighted k-means clustering,

$$c_k^{(0)} = \frac{\sum_{x \in \mathcal{X}_t} \delta_k(\hat{f}_t(x))\ \hat{g}_t(x)}{\sum_{x \in \mathcal{X}_t} \delta_k(\hat{f}_t(x))}, \tag{4}$$

  where $\delta_k(\cdot)$ denotes the $k$-th element in the softmax output and $\hat{f}_t = \hat{g}_t \circ h_t$ denotes the previously learned target hypothesis. These centroids can robustly and more reliably characterize the distribution of different categories within the target domain.
- We then obtain new pseudo labels via the nearest centroid classifier:

$$\hat{y}_t = \arg\min_k D_f(\hat{g}_t(x), c_k^{(0)}), \tag{5}$$

  where $D_f(a, b)$ measures the distance between $a$ and $b$. We use the cosine distance by default.

- Finally, we compute the target centroids based on the new pseudo labels:

$$c_k^{(1)} = \frac{\sum_{x \in \mathcal{X}_t} \mathbb{1}(\hat{y}_t = k)\ \hat{g}_t(x)}{\sum_{x \in \mathcal{X}_t} \mathbb{1}(\hat{y}_t = k)}, \quad \hat{y}_t = \arg\min_k D_f(\hat{g}_t(x), c_k^{(1)}). \tag{6}$$

We term $\hat{y}_t$ as self-supervised pseudo labels since they are generated by the centroids obtained in an unsupervised manner. Actually, this solution to pseudo labels behaves like that in Minimum Centroid Shift (MCS) [63] where target-specific centroids and pseudo labels are alternately updated via optimizing the intra-class divergence minimization loss. In contrast, we employ the cross-entropy loss and just update the centroids and labels in Eq. (6) for one round since updating once gives sufficiently good pseudo labels according to our observation in the experiment. We provide the cross-entropy loss of self-supervised pseudo-labeling below,

$$\mathcal{L}_{ssl}^1(f_t; \mathcal{X}_t, \hat{\mathcal{Y}}_t) = -\gamma_1\ \mathbb{E}_{(x,\hat{y}_t) \in \mathcal{X}_t \times \hat{\mathcal{Y}}_t} \sum_{k=1}^{K} \mathbb{1}_{[k=\hat{y}_t]} \log \delta_k(f_t(x)), \tag{7}$$

where $\gamma_1 > 0$ is a regularization parameter for the trade-off between $\mathcal{L}_{ssl}^1$ and the main task in Eq. (3).

Also, we investigate the image-specific self-supervision in the unlabeled target domain. Rotation prediction in RotNet [28] aims to recognize one of four different 2d rotation (i.e., 0°, 90°, 180°, and 270°) that is applied to the image that it gets as input, which is a simple yet effective criterion in the self-supervised learning field. It is further verified by several recent studies [89], [90] to learn semantically meaningful representations quite well, which is also desirable for domain adaptation problems. However, absolute rotation prediction is sensitive to some classification tasks. For example, in a main task aiming to distinguish digit '6' from digit '9', it is hard to determine which rotation category '9' belongs to, since '9' could also be a rotated '6' with 180 degrees or a rotated '9' with 0 degrees. To resolve this dilemma, we propose a new self-supervised learning task by predicting the relative rotation of each image pair. As shown in Fig. 3, the relative rotation predictor is represented by $h_c : \mathbb{R}^{2d} \to \{1, 2, 3, 4\}$ that takes the concatenated features of an image pair as input and maps them to one of four different rotation degrees.

For an image in the target domain $x_i \in \mathcal{X}_t$, we first randomly sample an integral number $z_i$ from $[1, 2, 3, 4]$ which corresponds to the rotation degree pool $[0°, 90°, 180°, 270°]$. Then we obtain the transformed image $x_i^{z_i} = Rot(x_i, z_i)$ by rotating $x_t$ with the associated degree $z_i$. Finally, the probability score of the $k$-th relative rotation degree predicted by $h_c$ is given by

$$\delta_k(h_c([g_t(x_i), g_t(x_i^{z_i})])), \tag{8}$$

where $\delta_k(\cdot)$ denotes the $k$-th element in the soft-max output vector, and $[\cdot, \cdot]$ denotes the feature-level concatenation function. Therefore, the self-supervised rotation prediction loss is defined as

$$\mathcal{L}_{ssl}^2(g_t, h_c; \mathcal{X}_t, \mathcal{Z}_t) = -\gamma_2\ \mathbb{E}_{(x_i,\hat{z}_i) \in \mathcal{X}_t \times \mathcal{Z}_t} \sum_{k=1}^{4} \mathbb{1}_{[k=z_i]} \log \delta_k(h_c([g_t(x_i), g_t(x_i^{z_i})])), \tag{9}$$

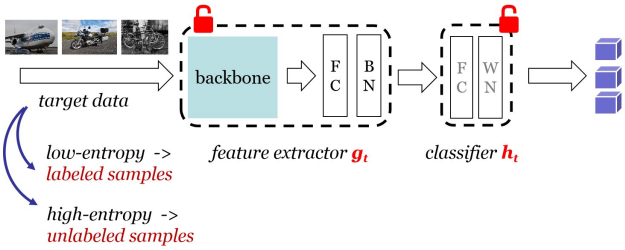



Fig. 4. The pipeline of the labeling transfer strategy with semi-supervised learning. Both the feature encoding module $g_t$ and the classification module $h_t$ are learned via the MixMatch [29] algorithm.

where $\gamma_2 > 0$ is a regularization parameter for the trade-off between $\mathcal{L}_{ssl}^2$ and the main loss, i.e. Eq. (3).

We provide an illustrative example of the complete hypothesis transfer framework in Fig. 3. To summarize, given the source model $f_s = g_s \circ h_s$ and pseudo labels $\hat{\mathcal{Y}}_t$ generated in Eq. (6) and randomly generated rotation labels $\mathcal{Z}_t$ as above, SHOT freezes the hypothesis from the source via $h_t \equiv h_s$ and learns the feature encoding module $g_t$ with the full optimization objective as

$$\mathcal{L}(g_t, h_c) = \mathcal{L}_{ent}(h_t \circ g_t; \mathcal{X}_t) + \beta \mathcal{L}_{div}(h_t \circ g_t; \mathcal{X}_t) - \gamma_1\ \mathbb{E}_{(x,\hat{y}_t) \in \mathcal{X}_t \times \hat{\mathcal{Y}}_t} \sum_{k=1}^{K} \mathbb{1}_{[k=\hat{y}_t]} \log \delta_k(f_t(x)) - \gamma_2\ \mathbb{E}_{(x_i,\hat{z}_i) \in \mathcal{X}_t \times \mathcal{Z}_t} \sum_{k=1}^{4} \mathbb{1}_{[k=z_i]} \log \delta_k(h_c([g_t(x_i), g_t(x_i^{z_i})])). \tag{10}$$

### 3.4 Labeling Transfer with Semi-supervised Learning

After we obtain the predictions for all the samples in the target domain via SHOT in Eq. (10), we can measure the confidence scores of these predictions via the entropy function $\mathbb{H}(p) = \sum_i p_i \log p_i$, where $p$ is a probability prediction vector. Observing the distribution of confidence scores, we find that there always exist some less confident (high-entropy) predictions that are possibly inaccurate. Fortunately, we can utilize the reliable labeling information from high confident predictions to improve the accuracy of these less confident ones. To this end, we propose a two-step method to enforce the information propagation from low-entropy predictions to high-entropy ones. In the first step, we divide the target domain into two splits according to the confidence scores and treat these two splits as a labeled subset and an unlabeled subset, respectively. In the second step, we readily employ a semi-supervised learning algorithm to learn the enhanced predictions for the unlabeled set here.

Regarding the choice of a semi-supervised learning algorithm in the second step, we simply adopt a popular and well-performing approach, MixMatch [29], which unifies consistency regularization, entropy minimization, and the MixUp regularization into one framework. Then the key point lies in *how to divide the target domain into two splits.* With average entropy, we first obtain the proportion of the labeled subset in the entire target domain by automatically computing

$$a = \frac{\sum_i \mathbb{1}(\xi_i < \frac{\sum_j \xi_j}{n_t})}{n_t}, \tag{11}$$

where $\xi \in \mathcal{R}^{n_t}$ denotes the entropy values of all the predictions in the target domain, where $\xi_i = \mathbb{H}(\delta(f_t(x_i))), i \in$

$[1, \cdots, n_t]$. Then for each class $k \in [0, K]$, we put the index with entropy values among the top $t_k$ smallest into the index pool of labeled split, where

$$t_k = \lfloor a \sum\nolimits_i \mathbb{1}(\bar{y}_i = k) \rfloor, \; k \in [1, \cdots, K], \tag{12}$$

and $\bar{y}_i \in [1, K]^{n_t}$ is the predicted label by SHOT in Eq. (10). In this manner, we get the labeled split, and the remaining samples constitute the unlabeled split. We call this strategy in Fig. 4 as labeling transfer since in this stage we only need the labeling information (predictions) while the feature encoding module $g_t$ is initialized with that learned in Eq. (10). Besides, the classification module $h_t$ is newly initialized from scratch and not frozen any more. So far, we develop a two-stage approach, called SHOT++, in which the first stage is SHOT in Eq. (10) and the second stage is the proposed labeling transfer strategy in Fig. 4.

### 3.5 Extension to Multi-source Domain Adaptation

We also provide an extension of the proposed SHOT approach for multi-source domain adaptation (MSDA) [18]. Different from vanilla UDA (one source and one target), there are multiple sources in the MSDA task. For simplicity, we run SHOT and SHOT-IM on each source-target pair and then sum up the probabilistic scores obtained from each pair. Finally, we get the predictions of samples in the target domain via the *argmax* operation. As for labeling transfer, we split the target domain into two pieces for each pair, and learn the independent prediction scores.

### 3.6 Extension to Partial-set Domain Adaptation

We also provide an extension of the proposed SHOT approach for partial-set domain adaptation (PDA) [91]. PDA differs from vanilla UDA in that the target label space is a subset of that of the source label space. Looking at the diversity-promoting term $\mathcal{L}_{div}$ in Eq. (10), it encourages the target domain to own a uniform label distribution. Though seemingly reasonable for solving closed-set UDA, it is not suitable for PDA. In reality, the target domain only contains some classes of all the classes in the source domain, making the label distribution sparse. Hence, we drop the second term $\mathcal{L}_{div}$ for PDA by letting $\beta = 0$.

Besides, within the self-supervised pseudo-labeling strategy, we usually need to obtain $K$ centroids in the target domain. However, for the PDA task, there are some tiny centroids which should be considered as empty like in k-means clustering. Therefore, SHOT discards tiny centroids with size smaller than $T_c$ in Eq. (6) for PDA problems.

### 3.7 Extension to Semi-supervised Domain Adaptation

We further extend the proposed SHOT approach for semi-supervised domain adaptation (SSDA) [32]. SSDA differs from UDA in that some labeled data exist in the target domain. Therefore, we adopt the supervised training loss in Eq. (2) for labeled target data and the complete loss in Eq. (10) for unlabeled target data. Besides, we also consider the labeled target data when computing the target-specific centriods. As for labeling transfer, we split the unlabeled target domain into two pieces and then add the labeled data into the labeled split.

### 3.8 Network Architecture

Here we discuss some architecture choices for the neural network model to parameterize both the feature encoding module and the hypothesis. First, we need to look back at the expected network outputs for cross-entropy loss in Eq. (1). If $y_s = k$, then maximizing $f_s^{(k)}(x_s) = \frac{\exp(w_k^\top g_s(x_s))}{\sum_i \exp(w_i^\top g_s(x_s))}$ means minimizing the distance between $g_s(x_s)$ and $w_k$, where $w_k$ is the $k$-th weight vector in the last FC layer. Ideally, all the samples from the $k$-th class would have a feature embedding near to $w_k$. If unlabeled target samples are given the correct pseudo labels, it is easily understandable that source feature embeddings are similar to target ones via the pseudo-labeling term in Eq. (7). The intuition behind is quite similar to previous studies [37], [92] where a simplified MMD is exploited for multi-modal domain confusion. Since the weight norm matters in the inner distance within the soft-max output, we adopt weight normalization (WN) [93] to keep the norm of each weight vector $w_i$ the same in the FC classifier layer. Besides, as indicated in prior studies, batch normalization (BN) [94] can reduce the internal dataset shift since different domains share the same mean (zero) and variance which can be considered as first-order and second-order moments. Based on these considerations, we form the frameworks of SHOT and SHOT++ as shown in Figs. 1∼4.

## 4 Experiments

### 4.1 Setup

To testify their versatility, we evaluate our methods in three unsupervised DA scenarios (i.e. closed-set, partial-set, multi-source), and one semi-supervised DA scenario over several popular visual benchmarks as introduced below.

**Digits** is a widely used DA benchmark that focuses on digit recognition. We follow the protocol of [42] and utilize three representative subsets: SVHN (**S**), MNIST (**M**), and USPS (**U**). We train our model using the training sets of each domain and report the recognition results on the standard test set of the target domain.

**Office** [30] is a standard DA benchmark which contains three domains, i.e., Amazon (**A**), DSLR (**D**), and Webcam (**W**), and each domain includes 31 object classes in the office environment. Gong et al. [95] further extract 10 shared categories between Office and Caltech-256 (**C**) to form a new benchmark named **Office-Caltech**. Both **Office** and **Office-Caltech** are considered small-sized.

**Office-Home** [96] is a challenging medium-sized benchmark, which consists of four distinct domains, i.e., Artistic images (**Ar**), Clip Art (**Cl**), Product images (**Pr**), and Real-World images (**Rw**). There are totally 65 everyday object categories in each domain.

**VisDA-C** [97] is a challenging large-scale benchmark that mainly focuses on the 12-class synthesis-to-real object recognition task. The source domain contains 152 thousand synthetic (**S**) images generated by rendering 3D models while the target domain has 55 thousand real (**R**) object images sampled from Microsoft COCO.

**PACS** [33] is a popular benchmark for multi-source domain adaptation. It contains four different domains, i.e.,

Art painting (**A**), Cartoon (**C**), Photo (**P**), and Sketch (**S**). There are totally 7 common categories in each domain.

**Baseline methods.** For vanilla unsupervised DA in digit recognition, we compare SHOT with ADDA [5], ADR [56], CDAN [42], CyCADA [8], CAT [98], SWD [99] and STAR [100]; for object recognition, we compare ours with DANN [24], DAN [4], SAFN [82], BSP [101], MDD [102], TransNorm [55], DSBN [25], BNM [103] and GVB-GD [104]. For partial-set DA tasks, we compare ours with IWAN [105], SAN [91], ETN [106], DRCN [107], RTN$et_{adv}$ [108], BA$^3$US [109], and TSCDA [110]. For multi-source UDA, we compare ours with DCTN [111], MCD [6], WBN [112], M$^3$SDA-$\beta$ [18], Meta-MCD [118], SImAl [119], and CMSS [113]. For SSDA, we mainly compare our methods with MME [32] and UODA [114]. Note that results are directly cited from published papers if we follow the same setting. 'Source-model-only' (also called 'src-only') denotes using the entire model learned from the source domain for target label prediction. 'labeled-data-only' denotes using labeled target data only when learning the feature extractor $g_t$. 'Target-supervised' denotes training with the target data itself. For datasets without train-validation splits, we divide the target domain into three parts (**0.6**/**0.2**/**0.2**) as training, validation, and testing sets. Then we train the network via the training set and the validation set and finally report the accuracy on the testing set. SHOT-IM is a special case of SHOT, where both self-supervised losses are ignored by letting $\gamma_1 = \gamma_2 = 0$ in Eq. (10).

### 4.2 Implementation Details

**Network architecture.** For the digit recognition task, we use the same architectures with CDAN [42], namely, the classical LeNet-5 [115] network for USPS↔MNIST and a variant of LeNet for SVHN→MNIST. More network details can be found in *Appendix A* of [23]. For the object recognition task, we employ the pre-trained ResNet-50 or ResNet-101 [116] models as the backbone, like [18], [42], [82], [98]. Following [24], we replace the original FC layer with a bottleneck layer (256 units) and a task-specific FC classifier layer in Fig. 1. Precisely, a BN layer is put after FC inside the bottleneck layer and a weight normalization layer is utilized in the task-specific FC layer.

**Network hyper-parameters.** We train the whole network through back-propagation, and the newly added layers are trained with a learning rate 10 times that of the pre-trained layers (backbone shown in Fig. 1). Concretely, we adopt mini-batch SGD with momentum 0.9, weight decay 1e$^{-3}$ and learning rate $\eta_0 = 1e^{-2}$ for the new layers and those layers learned from scratch for all experiments except $\eta_0 = 1e^{-3}$ for **VisDA-C**. We further adopt the same learning rate scheduler $\eta = \eta_0 \cdot (1 + 10 \cdot p)^{-0.75}$ as [24], [42], where $p$ is the training progress changing from 0 to 1. Besides, we set the batch size to 64 for all the tasks. We utilize $\gamma_1 = 0.3, \gamma_2 = 0.6$ for all experiments except $\gamma_1 = 0.1, \gamma_2 = 0.2$ for **Digits** in Table 1 and SSDA in Table 7. Concerning the labeling transfer strategy, only 'source-model-only++' for object recognition does not use the learned source model as initialization.

For **Digits**, we train the best source hypothesis using the test set of the source dataset as validation. For other datasets without train-validation splits, we randomly specify a **0.9**/**0.1** split in the source dataset and generate the best source hypothesis based on the validation split. The maximum number of epochs for **Digits**, **Office**, **Office-Home**, **VisDA-C** and **Office-Caltech** is empirically set as 30, 100, 50, 10, and 100, respectively. For learning in the target domain, we update the pseudo-labels epoch by epoch, and the maximum number of epochs is empirically set as 15. Regarding the second step in Section 3.4, we adopt the same learning setting as that of training SHOT and the default parameters $\alpha = 0.75$ within MixMatch [29]. We utilize $\alpha = 0.1$ only for **Digits**. Besides, we randomly run our methods for three times with different random seeds $\{2019, 2020, 2021\}$ via **PyTorch**, and report the mean accuracy. Note that we do not use any target augmentation such as the ten-crop ensemble [42] for evaluation.

TABLE 1
Classification accuracies (%) on **Digits** dataset for *vanilla closed-set UDA*. S: SVHN, M:MNIST, U: USPS. (Best value is in **red color**)

| Method (Source → Target) | S → M | U → M | M → U | Avg. |
|---|---|---|---|---|
| Source only [8] | 67.1±0.6 | 69.6±3.8 | 82.2±0.8 | 73.0 |
| ADDA [5] | 76.0±1.8 | 90.1±0.8 | 89.4±0.2 | 85.2 |
| ADR [56] | 95.0±1.9 | 93.1±1.3 | 93.2±2.5 | 93.8 |
| CyCADA [8] | 90.4±0.4 | 96.5±0.1 | 95.6±0.4 | 94.2 |
| CDAN [42] | 89.2 | **98.0** | 95.6 | 94.3 |
| rRevGrad+CAT [98] | 98.8±0.0 | 96.0±0.9 | 94.0±0.7 | 96.3 |
| SWD [99] | 98.9±0.1 | 97.1±0.1 | 98.1±0.1 | 98.0 |
| STAR [100] | 98.8±0.1 | 97.7±0.1 | 97.8±0.1 | 98.1 |
| Source-model-only | 71.2±0.7 | 88.0±2.5 | 78.4±2.3 | 79.2 |
| SHOT-IM | 98.5±0.8 | 97.6±0.1 | 97.8±0.4 | 97.9 |
| SHOT | **99.0**±0.1 | 97.6±0.2 | 97.8±0.3 | 98.1 |
| Source-model-only++ | 88.7±3.0 | 94.5±0.9 | 89.8±2.0 | 91.0 |
| SHOT-IM++ | 98.5±0.8 | 97.7±0.1 | **98.4**±0.4 | 98.2 |
| SHOT++ | 98.9±0.1 | 97.8±0.1 | **98.4**±0.1 | **98.4** |
| Target-supervised | 99.2±0.1 | 99.2±0.1 | 96.8±0.2 | 98.4 |

TABLE 2
Classification accuracies (%) on small-sized **Office** dataset for *vanilla closed-set UDA* (ResNet-50).

| Method (Source→Target) | A→D | A→W | D→A | D→W | W→A | W→D | Avg. |
|---|---|---|---|---|---|---|---|
| ResNet-50 [116] | 68.9 | 68.4 | 62.5 | 96.7 | 60.7 | 99.3 | 76.1 |
| DAN [4] | 78.6 | 80.5 | 63.6 | 97.1 | 62.8 | 99.6 | 80.4 |
| DANN [24] | 79.7 | 82.0 | 68.2 | 96.9 | 67.4 | 99.1 | 82.2 |
| SAFN+ENT [82] | 90.7 | 90.1 | 73.0 | 98.6 | 70.2 | 99.8 | 87.1 |
| rRevGrad+CAT [98] | 90.8 | 94.4 | 72.2 | 98.0 | 70.2 | **100.** | 87.6 |
| CDAN [42] | 92.9 | 94.1 | 71.0 | 98.6 | 69.3 | **100.** | 87.7 |
| DSBN+MSTN [25] | 92.2 | 92.7 | 71.7 | **99.0** | 74.4 | **100.** | 88.3 |
| CDAN+BSP [101] | 93.0 | 93.3 | 73.6 | 98.2 | 72.6 | **100.** | 88.5 |
| CDAN+BNM [103] | 92.9 | 92.8 | 73.5 | 98.8 | 73.8 | **100.** | 88.6 |
| MDD [102] | 93.5 | 94.5 | 74.6 | 98.4 | 72.2 | **100.** | 88.9 |
| CDAN+TransNorm [55] | 94.0 | **95.7** | 73.4 | 98.7 | 74.2 | **100.** | **89.3** |
| GVB-GD [104] | **95.0** | 94.8 | 73.4 | 98.7 | 73.7 | **100.** | **89.3** |
| Source-model-only | 80.2 | 76.9 | 60.3 | 95.4 | 63.6 | 98.9 | 79.2 |
| SHOT-IM | 90.2 | 91.1 | 72.4 | 98.3 | 71.8 | 99.9 | 87.3 |
| SHOT | 93.9 | 90.1 | 75.3 | 98.7 | 75.0 | 99.9 | 88.8 |
| Source-model-only++ | 88.5 | 87.3 | 69.0 | 97.7 | 70.8 | 99.0 | 85.4 |
| SHOT-IM++ | 90.9 | 91.9 | 73.5 | 98.6 | 72.5 | 99.7 | 87.8 |
| SHOT++ | 94.3 | 90.4 | **76.2** | 98.7 | **75.8** | 99.9 | 89.2 |
| Target-supervised | 98.0 | 98.7 | 86.0 | 98.7 | 86.0 | 98.0 | 94.3 |

TABLE 3
Classification accuracies (%) on medium-sized **Office-Home** dataset for *vanilla closed-set UDA* (ResNet-50).

| Method (Source→Target) | Ar→Cl | Ar→Pr | Ar→Re | Cl→Ar | Cl→Pr | Cl→Re | Pr→Ar | Pr→Cl | Pr→Re | Re→Ar | Re→Cl | Re→Pr | Avg. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ResNet-50 [116] | 34.9 | 50.0 | 58.0 | 37.4 | 41.9 | 46.2 | 38.5 | 31.2 | 60.4 | 53.9 | 41.2 | 59.9 | 46.1 |
| DANN [24] | 45.6 | 59.3 | 70.1 | 47.0 | 58.5 | 60.9 | 46.1 | 43.7 | 68.5 | 63.2 | 51.8 | 76.8 | 57.6 |
| DAN [4] | 43.6 | 57.0 | 67.9 | 45.8 | 56.5 | 60.4 | 44.0 | 43.6 | 67.7 | 63.1 | 51.5 | 74.3 | 56.3 |
| CDAN [42] | 50.7 | 70.6 | 76.0 | 57.6 | 70.0 | 70.0 | 57.4 | 50.9 | 77.3 | 70.9 | 56.7 | 81.6 | 65.8 |
| CDAN+BSP [101] | 52.0 | 68.6 | 76.1 | 58.0 | 70.3 | 70.2 | 58.6 | 50.2 | 77.6 | 72.2 | 59.3 | 81.9 | 66.3 |
| SAFN [82] | 52.0 | 71.7 | 76.3 | 64.2 | 69.9 | 71.9 | 63.7 | 51.4 | 77.1 | 70.9 | 57.1 | 81.5 | 67.3 |
| CDAN+TransNorm [55] | 50.2 | 71.4 | 77.4 | 59.3 | 72.7 | 73.1 | 61.0 | 53.1 | 79.5 | 71.9 | 59.0 | 82.9 | 67.6 |
| MDD [102] | 54.9 | 73.7 | 77.8 | 60.0 | 71.4 | 71.8 | 61.2 | 53.6 | 78.1 | 72.5 | 60.2 | 82.3 | 68.1 |
| CDAN+BNM [103] | 56.2 | 73.7 | 79.0 | 63.1 | 73.6 | 74.0 | 62.4 | 54.8 | 80.7 | 72.4 | 58.9 | 83.5 | 69.4 |
| GVB-GD [104] | 57.0 | 74.7 | 79.8 | 64.6 | 74.1 | 74.6 | 65.2 | 55.1 | 81.0 | **74.6** | 59.7 | 84.3 | 70.4 |
| Source-model-only | 44.5 | 67.6 | 74.7 | 52.5 | 62.8 | 64.9 | 53.1 | 40.5 | 73.3 | 65.3 | 45.1 | 77.9 | 60.2 |
| SHOT-IM | 55.9 | 76.8 | 80.6 | 66.7 | 73.7 | 75.4 | 65.4 | 54.9 | 80.9 | 73.2 | 58.5 | 83.5 | 70.5 |
| SHOT | 57.7 | 79.1 | 81.5 | 67.6 | 77.9 | 77.8 | 68.1 | 55.8 | 82.0 | 72.8 | 59.7 | 84.4 | 72.0 |
| Source-model-only++ | 50.2 | 75.9 | 79.7 | 62.6 | 74.3 | 74.8 | 59.5 | 44.4 | 79.9 | 69.4 | 45.4 | 83.1 | 66.6 |
| SHOT-IM++ | 56.9 | 77.7 | 81.5 | 67.6 | 74.9 | 76.9 | 66.1 | 55.9 | 81.7 | 73.8 | 59.3 | 84.4 | 71.4 |
| SHOT++ | **57.9** | **79.7** | **82.5** | **68.5** | **79.6** | **79.3** | **68.5** | **57.0** | **83.0** | 73.7 | **60.7** | **84.9** | **73.0** |
| Target-supervised | 77.9 | 91.4 | 84.4 | 74.5 | 91.4 | 84.4 | 74.5 | 77.9 | 84.4 | 74.5 | 77.9 | 91.4 | 82.0 |

TABLE 4
Classification accuracies (%) on large-scale **VisDA-C** dataset for *vanilla closed-set UDA* (ResNet-101).

| Method (Synthesis → Real) | plane | bcycl | bus | car | horse | knife | mcycl | person | plant | sktbrd | train | truck | Per-class |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ResNet-101 [116] | 55.1 | 53.3 | 61.9 | 59.1 | 80.6 | 17.9 | 79.7 | 31.2 | 81.0 | 26.5 | 73.5 | 8.5 | 52.4 |
| DANN [24] | 81.9 | 77.7 | 82.8 | 44.3 | 81.2 | 29.5 | 65.1 | 28.6 | 51.9 | 54.6 | 82.8 | 7.8 | 57.4 |
| DAN [4] | 87.1 | 63.0 | 76.5 | 42.0 | 90.3 | 42.9 | 85.9 | 53.1 | 49.7 | 36.3 | 85.8 | 20.7 | 61.1 |
| ADR [56] | 94.2 | 48.5 | 84.0 | 72.9 | 90.1 | 74.2 | 92.6 | 72.5 | 80.8 | 61.8 | 82.2 | 28.8 | 73.5 |
| CDAN+BSP [101] | 92.4 | 61.0 | 81.0 | 57.5 | 89.0 | 80.6 | 90.1 | 77.0 | 84.2 | 77.9 | 82.1 | 38.4 | 75.9 |
| SAFN [82] | 93.6 | 61.3 | 84.1 | 70.6 | 94.1 | 79.0 | 91.8 | 79.6 | 89.9 | 55.6 | 89.0 | 24.4 | 76.1 |
| SWD [99] | 90.8 | 82.5 | 81.7 | 70.5 | 91.7 | 69.5 | 86.3 | 77.5 | 87.4 | 63.6 | 85.6 | 29.2 | 76.4 |
| DSBN+MSTN [25] | 94.7 | 86.7 | 76.0 | 72.0 | 95.2 | 75.1 | 87.9 | 81.3 | 91.1 | 68.9 | 88.3 | 45.5 | 80.2 |
| DTA [57] | 93.7 | 82.2 | 85.6 | 83.8 | 93.0 | 81.0 | 90.7 | 82.1 | 95.1 | 78.1 | 86.4 | 32.1 | 81.5 |
| STAR [100] | 95.0 | 84.0 | 84.6 | 73.0 | 91.6 | 91.8 | 85.9 | 78.4 | 94.4 | 84.7 | 87.0 | 42.2 | 82.7 |
| CAN [46] | 97.0 | 87.2 | 82.5 | 74.3 | 97.8 | 96.2 | 90.8 | 80.7 | 96.6 | **96.3** | 87.5 | **59.9** | 87.2 |
| Source-model-only | 64.1 | 24.9 | 53.0 | 66.5 | 67.9 | 9.1 | 84.5 | 21.1 | 62.8 | 29.8 | 83.5 | 9.3 | 48.0 |
| SHOT-IM | 93.7 | 86.4 | 78.7 | 50.6 | 91.0 | 93.6 | 79.0 | 78.3 | 89.3 | 85.4 | 88.0 | 51.1 | 80.4 |
| SHOT | 95.8 | 88.2 | 87.2 | 73.7 | 95.2 | 96.4 | 87.9 | **84.5** | 92.5 | 89.3 | 85.7 | 49.1 | 85.5 |
| Source-model-only++ | 73.0 | 12.9 | 76.1 | **90.3** | 93.7 | 1.5 | **94.9** | 40.9 | 84.6 | 75.1 | 91.2 | 4.9 | 61.6 |
| SHOT-IM++ | 96.7 | 87.6 | 89.2 | 71.4 | 96.3 | 98.5 | 91.9 | 79.9 | 95.5 | 86.4 | 93.5 | 32.9 | 85.0 |
| SHOT++ | **97.7** | **88.4** | **90.2** | 86.3 | **97.9** | **98.6** | **92.9** | 84.1 | **97.1** | 92.2 | **93.6** | 28.8 | **87.3** |
| Target-supervised | 97.0 | 86.6 | 84.3 | 88.7 | 96.3 | 94.4 | 92.0 | 89.4 | 95.5 | 91.8 | 90.7 | 68.7 | 89.6 |

### 4.3 Results of Digit Recognition (Vanilla Closed-set)

For digit recognition, we evaluate our methods on three popular closed-set unsupervised domain adaptation tasks, i.e., SVHN→MNIST, USPS→MNIST, and MNIST→USPS. The classification accuracies of our methods and prior work are reported in Table 1. Obviously, SHOT obtains the best mean accuracy for each task and also outperforms prior work in terms of the average accuracy. Compared with the baseline method source-model-only, SHOT-IM always achieves better results, and SHOT performs better than SHOT-IM due to the contribution of self-supervised learning in the target domain. Taking into consideration the labeling transfer strategy, all three methods are able to obtain enhanced classification results, indicating the effectiveness of intra-domain semi-supervised learning. It is also worth noting that SHOT++ even offers superior performance to the target-supervised result in MNIST→USPS. This may be because MNIST is much larger than USPS, which alleviates the domain shift well.

### 4.4 Results of Object Recognition (Vanilla Closed-set)

Next, we evaluate our methods on object recognition benchmarks including **Office**, **Office-Home** and **VisDA-C** under the vanilla closed-set DA setting. As shown in Table 2, SHOT performs the best for two challenging tasks, D→A and W→A, and obtains an average accuracy 88.8% that is competitive to two state-of-the-art methods, MDD [102] and BNM [103]. Similar to the observations in Table 1, the labeling transfer strategy is beneficial to cross-domain object recognition, and SHOT++ obtains the same mean accuracy as previous state-of-the-art methods, TransNorm [55] and GVB-GD [104]. This may be because SHOT needs a relatively large target domain to learn the target-specific module $g_t$ while $D$ and $W$ as the target domain are not big enough. Generally, SHOT obtains competitive performance even with no direct access to the source domain data.

As expected, on the medium-sized **Office-Home** dataset, our method SHOT++ significantly outperforms previously published state-of-the-art approaches, advancing the aver-

TABLE 5
Classification accuracies (%) on **Office-Caltech** (ResNet-101) and **Office-Home** (ResNet-50) and **PACS** (ResNet-18) for *multi-source UDA*.

| (**Office-Caltech**) | →A | →C | →D | →W | Avg. | (**Office-Home**) | →Ar | →Cl | →Pr | →Re | Avg. | (**PACS**) | →A | →C | →P | →S | Avg. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ResNet-101 [116] | 88.7 | 85.4 | 98.2 | 99.1 | 92.9 | ResNet-50 [116] | 65.3 | 49.6 | 79.7 | 75.4 | 67.5 | ResNet-18 [116] | 74.9 | 72.1 | 94.5 | 64.7 | 76.6 |
| DAN [4] | 91.6 | 89.2 | 99.1 | 99.5 | 94.8 | M$^3$SDA-$\beta$ [18] | 67.2 | 58.6 | 79.1 | 81.2 | 71.5 | DANN [24] | 81.9 | 77.5 | 91.8 | 74.6 | 81.5 |
| DCTN [111] | 92.7 | 90.2 | 99.0 | 99.4 | 95.3 | Meta-DANN [118] | 70.6 | 59.1 | 80.2 | 82.8 | 73.2 | Meta-DANN [118] | 87.3 | 84.9 | 96.9 | 73.2 | 85.6 |
| MCD [6] | 92.1 | 91.5 | 99.1 | 99.5 | 95.6 | MCD [6] | 69.8 | 59.8 | 80.9 | 82.7 | 73.3 | Meta-MCD [118] | 87.4 | 86.2 | 97.1 | 78.3 | 87.2 |
| M$^3$SDA-$\beta$ [18] | 94.5 | 92.2 | 99.2 | 99.5 | 96.4 | Meta-MCD [118] | 70.2 | 60.5 | 81.2 | 83.4 | 73.8 | M$^3$SDA-$\beta$ [18] | 89.3 | 89.9 | 97.3 | 76.7 | 88.3 |
| CMSS [113] | 96.0 | 93.7 | 99.3 | 99.6 | 97.2 | SImpAl [119] | 72.1 | 62.0 | 80.3 | 81.8 | 74.1 | CMSS [113] | 88.6 | 90.4 | 96.9 | 82.0 | 89.5 |
| Source-model-only | 95.4 | 93.6 | 98.9 | 98.4 | 96.6 | Source-model-only | 67.3 | 51.2 | 78.7 | 81.4 | 69.6 | Source-model-only | 63.6 | 51.7 | 94.4 | 47.4 | 64.3 |
| SHOT-IM | 96.3 | 95.5 | 99.6 | 99.8 | 97.8 | SHOT-IM | 72.1 | 60.3 | 82.4 | 82.9 | 74.4 | SHOT-IM | 89.6 | 87.9 | 98.6 | 62.5 | 84.7 |
| SHOT | 96.2 | 96.2 | 98.5 | 99.8 | 97.7 | SHOT | 73.0 | 60.4 | 83.9 | 83.3 | 75.2 | SHOT | 90.7 | 88.1 | 98.5 | 75.4 | 88.2 |
| Source-only++ | 96.3 | 95.5 | 99.6 | 99.8 | 97.8 | Source-only++ | 70.3 | 51.6 | 83.0 | 83.5 | 72.1 | Source-only++ | 72.1 | 44.1 | 98.1 | 41.5 | 63.9 |
| SHOT-IM++ | 96.5 | 96.5 | 99.2 | 99.9 | 98.0 | SHOT-IM++ | 72.3 | 60.4 | 82.5 | 83.2 | 74.6 | SHOT-IM++ | 91.5 | 89.8 | 98.9 | 64.0 | 86.0 |
| SHOT++ | 96.2 | 96.5 | 99.4 | 100. | 98.0 | SHOT++ | 73.1 | 61.3 | 84.3 | 84.0 | 75.7 | SHOT++ | 92.3 | 89.7 | 98.8 | 75.5 | 89.1 |
| Target-supervised | 96.7 | 95.3 | 99.0 | 98.9 | 97.5 | Target-supervised | 74.5 | 77.9 | 91.4 | 84.4 | 82.0 | Target-supervised | 92.7 | 92.9 | 98.4 | 93.8 | 94.5 |

TABLE 6
Classification accuracies (%) on **Office-Home** and **VisDA-C** for *partial-set UDA* (ResNet-50).

| Methods | **Office-Home (65→25)** | | | | | | | | | | | | | **VisDA-C (12→6)** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Source→Target | Ar→Cl | Ar→Pr | Ar→Re | Cl→Ar | Cl→Pr | Cl→Re | Pr→Ar | Pr→Cl | Pr→Re | Re→Ar | Re→Cl | Re→Pr | Avg. | R→S | S→R | Avg. |
| ResNet-50 [116] | 46.3 | 67.5 | 75.9 | 59.1 | 59.9 | 62.7 | 58.2 | 41.8 | 74.9 | 67.4 | 48.2 | 74.2 | 61.3 | 64.3 | 45.3 | 54.8 |
| IWAN [105] | 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 | 71.3 | 48.6 | 60.0 |
| SAN [91] | 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 | 69.7 | 49.9 | 59.8 |
| DRCN [107] | 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 | 73.2 | 58.2 | 65.7 |
| ETN [106] | 59.2 | 77.0 | 79.5 | 62.9 | 65.7 | 75.0 | 68.3 | 55.4 | 84.4 | 75.7 | 57.7 | 84.5 | 70.5 | - | - | - |
| SAFN [82] | 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 | - | - | - |
| RTNet$_{adv}$ [108] | 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 [109] | 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 | - | - | - |
| TSCDA [110] | 63.6 | 82.5 | 89.6 | 73.7 | 73.9 | 81.4 | 75.4 | 61.6 | 87.9 | 83.6 | 67.2 | 88.8 | 77.4 | - | - | - |
| Source-model-only | 44.9 | 70.5 | 81.0 | 55.4 | 60.2 | 66.2 | 61.5 | 40.3 | 76.5 | 70.6 | 47.8 | 77.2 | 62.7 | 60.9 | 46.6 | 53.8 |
| SHOT-IM | 59.1 | 83.9 | 88.5 | 72.7 | 73.5 | 78.4 | 75.9 | 59.9 | 90.3 | 81.3 | 68.6 | 88.7 | 76.7 | 69.2 | 68.8 | 69.0 |
| SHOT | 64.6 | 85.1 | 92.9 | 78.4 | 76.8 | 86.9 | 79.0 | 65.7 | 89.0 | 81.1 | 67.7 | 86.4 | 79.5 | 73.1 | 74.2 | 73.6 |
| Source-model-only++ | 50.3 | 77.1 | 86.6 | 66.2 | 67.6 | 75.7 | 69.2 | 46.4 | 83.6 | 76.2 | 51.3 | 82.4 | 69.4 | 67.7 | 65.8 | 66.8 |
| SHOT-IM++ | 59.6 | 84.5 | 89.0 | 73.7 | 74.2 | 79.3 | 77.0 | 60.7 | 91.0 | 81.8 | 69.4 | 89.3 | 77.5 | 70.0 | 75.7 | 72.9 |
| SHOT++ | 65.0 | 85.8 | 93.4 | 78.8 | 77.4 | 87.3 | 79.3 | 66.0 | 89.6 | 81.3 | 68.1 | 86.8 | 79.9 | 75.3 | 78.6 | 77.0 |
| Target-supervised | 81.0 | 91.5 | 85.8 | 80.0 | 91.5 | 85.8 | 80.0 | 81.0 | 85.8 | 80.0 | 81.0 | 91.5 | 84.6 | 98.8 | 89.9 | 94.3 |

age accuracy from 70.4% in GVB-GD [104] to 73.0% in Table 3. Besides, SHOT++ performs the best among 11 out of 12 separate tasks. For the transfer task Re→Ar, SHOT++ gets the third-best result 73.7% that is slightly lower than the best result 74.6% of GVB-GD. Generally, the hypothesis transfer strategy works well enough, seen from the outperforming results of SHOT over prior methods, and the labeling transfer strategy further lifts the avg. accuracy by nearly 1 point.

For the large-scale synthesis-to-real **VisDA-C** dataset, we follow the protocol in prior works [56], [82] and employ the most favoring backbone ResNet-101 [116]. As shown in Table 4, SHOT++ achieves the best per-class accuracy and wins among 8 out of 12 tasks. Even when ignoring the second stage, namely, labeling transfer, SHOT can still obtain a promising per-class result 85.5%, higher than the prior state-of-the-art 82.7% in STAR [100]. Carefully comparing SHOT with prior work, we find that SHOT performs well even for the most challenging class 'truck'. Besides, using the intra-domain semi-supervised learning stage via MixMatch, the per-class results are improved but the accuracy of the hard class 'truck' decreases. This may be because large error in the labeled split affects the final results.

### 4.5 Results of Object Recognition beyond Vanilla UDA

**Results of object recognition for MSDA.** For the multi-source UDA setting, we adopt the protocol in [113] on **Office-Caltech** and **PACS** and the prototcol in [118] on **Office-Home**. For the three datasets, we specify a target subset and use other three subsets as three source domains, forming a multi-source UDA task. Likewise, SHOT does not access the source data but provided with multiple source models instead. The results of ours and previously published state-of-the-arts are shown in Table 5. It is clear that SHOT achieves better results than CMSS [113] or SImpAl [119] in 3 out of 4 tasks on **Office-Caltech**, 3 of the 4 tasks on **PACS**, and 2 of the 4 tasks on **PACS**, respectively. With the incorporation of labeling transfer, SHOT++ wins SHOT for all these transfer tasks on the three datasets. Besides, the gap between SHOT and SHOT-IM is relatively small on **Office-Caltech** since the predictions learned by SHOT-IM are already good enough. On **PACS**, SHOT++ achieves competitive performance with that in CMSS [113].

**Results of object recognition for PDA.** For the partial-set UDA setting, we follow the protocol in [107] on **Office-Home** and **VISDA-C**. In particular, there are totally 25 classes (the first 25 in the alphabetical order) out of 65 classes in the target domain for **Office-Home**, while the first 6 classes in the alphabetical order out of 12 classes are included in the target domain for **VISDA-C**. Results of our methods and previous state-of-the-art PDA methods [107], [108], [109], [110] are shown in Table 6. As explained in Section 3.6, $\beta = 0$ is utilized in all of our methods here.

TABLE 7
Classification accuracies (%) on **Office-Home** dataset for *semi-supervised DA* (VGG16 on one-shot setting).

| SSDA (Source→Target) | Ar→Cl | Ar→Pr | Ar→Re | Cl→Ar | Cl→Pr | Cl→Re | Pr→Ar | Pr→Cl | Pr→Re | Re→Ar | Re→Cl | Re→Pr | Avg. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| S+T [32] | 37.5 | 63.6 | 69.5 | 51.4 | 65.9 | 64.5 | 52.0 | 37.0 | 71.6 | 61.2 | 39.5 | 75.3 | 57.4 |
| DANN [24] | 44.4 | 64.3 | 68.9 | 52.3 | 65.3 | 64.2 | 51.3 | 45.9 | 72.7 | 62.7 | 52.0 | 75.7 | 60.0 |
| PAC [90] | 43.5 | 69.8 | 69.5 | 45.3 | 69.6 | 65.3 | 55.3 | 54.7 | 73.1 | 64.6 | 56.4 | 78.8 | 62.2 |
| MME [32] | 45.8 | 68.6 | 72.2 | 57.5 | 71.3 | 68.0 | 56.0 | 46.2 | 74.4 | 65.1 | 49.1 | 78.7 | 62.7 |
| ELP [120] | 46.1 | 69.0 | 72.4 | 57.4 | 71.6 | 68.2 | 56.3 | 46.7 | 75.3 | 65.5 | 49.2 | 79.7 | 63.1 |
| UODA [114] | 43.3 | 72.5 | 73.3 | 59.3 | 72.1 | 70.5 | 58.8 | 45.5 | 75.4 | 66.1 | 49.6 | 79.8 | 63.9 |
| labeled-data-only | 40.7 | 66.7 | 69.2 | 52.9 | 67.6 | 65.1 | 52.4 | 38.1 | 70.7 | 61.4 | 42.9 | 75.5 | 58.6 |
| SHOT-IM | 47.5 | 72.4 | 74.1 | 59.4 | 73.3 | 71.2 | 57.9 | 45.2 | 76.5 | 64.5 | 49.6 | 80.6 | 64.4 |
| SHOT | 49.1 | 73.9 | 74.9 | 59.4 | 75.0 | 72.9 | 58.0 | 47.0 | 77.1 | 65.0 | 50.7 | 80.8 | 65.3 |
| labeled-data-only++ | 41.8 | 71.7 | 71.9 | 58.2 | 74.3 | 69.9 | 55.9 | 39.2 | 75.0 | 63.7 | 43.8 | 78.9 | 62.0 |
| SHOT-IM++ | 48.1 | 73.6 | 75.3 | 60.5 | 74.6 | 72.1 | 58.9 | 45.6 | 76.7 | 64.8 | 50.2 | 81.4 | 65.2 |
| SHOT++ | 49.7 | 75.0 | 76.0 | 60.4 | 76.1 | 73.6 | 59.8 | 47.5 | 77.6 | 65.4 | 51.1 | 81.7 | 66.1 |
| Target-supervised | 75.8 | 88.3 | 81.6 | 66.4 | 88.3 | 81.6 | 66.4 | 75.8 | 81.6 | 66.4 | 75.8 | 88.3 | 78.0 |

TABLE 8
Accuracies for **ImageNet→Caltech**. Methods † utilize the training set of ImageNet besides pre-trained ResNet-50 model.

| Methods† | DRCN [107] | SAN [91] | IWAN [105] | ETN [106] |
|---|---|---|---|---|
| Accuracy | 75.3 | 77.8 | 78.1 | 83.2 ± 0.2 |
| Methods | Source-only | SHOT-IM | SHOT ($\gamma_2 = 0$) | SHOT |
| Accuracy | 69.7 | 81.8 ± 0.4 | 83.1 ± 0.1 | 83.3 ± 0.3 |

TABLE 9
Classification accuracies (%) on large-scale **VisDA-C** dataset for *vanilla closed-set UDA* (ResNet-50).

| Method | Per-class | Method | Per-class |
|---|---|---|---|
| ResNet-50 [116] | 52.4 | CDAN [42] | 70.0 |
| DANN [24] | 57.4 | CDAN+TransNorm [55] | 71.4 |
| DAN [4] | 61.6 | MDD [102] | 74.6 |
| MCD [6] | 69.2 | GVB-GD [104] | 75.3 |
| Dis-tune [117] | 70.4 | DTA [57] | 76.2 |
| Source-model-only | 43.5 | Source-model-only++ | 52.2 |
| SHOT-IM ($\beta = 0$) | 64.6 | SHOT-IM++ ($\beta = 0$) | 67.9 |
| SHOT-IM | 73.9 | SHOT-IM++ | 75.6 |
| SHOT ($\gamma_1 = 0$) | 74.6 | SHOT++ ($\gamma_1 = 0$) | 76.3 |
| SHOT ($\gamma_2 = 0$) | 74.8 | SHOT++ ($\gamma_2 = 0$) | 76.5 |
| SHOT | 76.7 | SHOT++ | 77.1 |
| Target-supervised (Synthesis → **Real**) | | | 88.8 |

Compared with previous methods, SHOT obtains the best average accuracy for both datasets as before. Besides, SHOT again outperforms SHOT-IM by 2.8% and 4.6% in terms of the average accuracy on two datasets, and SHOT++ further improves the average accuracy from 79.5% to 79.9% and 73.6% to 77.0%, respectively. Generally, both the hypothesis transfer strategy and the labeling transfer strategy are proven effective for the challenging PDA problem.

**Results of object recognition for SSDA.** For the semi-supervised domain adaptation setting, we follow the protocol in [32] on **Office-Home** under the one-shot setting where one labeled example per class is available in the target domain. As shown in Table 7, SHOT outperforms UODA [114] and MME [32] in 10 out of 12 tasks and achieves the best average accuracy. Besides, SHOT is always superior to SHOT-IM, validating the effectiveness of self-supervision over the unlabeled target data. SHOT++ further improves the average accuracy from 65.3% to 66.1%, indicating the effectiveness of the labeling transfer strategy.

**Special case.** One may wonder whether SHOT works if we cannot train the source model by ourselves. To find the answer, we utilize the most popular off-the-shelf pre-trained ImageNet models ResNet-50 [116] and consider a special PDA task (**ImageNet→Caltech**) to evaluate the effectiveness of SHOT with the same basic setting as [106]. Obviously, in Table 8, SHOT achieves a slightly higher mean accuracy than prior state-of-the-art ETN [106] even without access to the source data. It shows that the proposed hypothesis transfer strategy is indeed effective even without the design of model network architectures.

### 4.6 Model Analysis and Discussions

**Ablation study on different losses.** Following previous works [102], [104], we further adopt the ResNet-50 [116] backbone to validate the effectiveness of our methods. Results are shown in Table 9. With the hypothesis transfer strategy, SHOT beats the state-of-the-art method DTA [57] by 0.5% in terms of per-class accuracy. Benefited from the labeling transfer strategy, the per-class accuracy further grows from 76.7% (SHOT) to 77.1% (SHOT++) and again ranks the best for **VisDA-C** with the ResNet-50 backbone.

In Table 9, we further fix three balancing parameters (i.e., $\beta, \gamma_1, \gamma_2$) to zero in turn and investigate the effectiveness of each component within SHOT in Eq. (10), including $\mathcal{L}_{div}$, $\mathcal{L}_{ssl}^1$, and $\mathcal{L}_{ssl}^2$. Firstly, the advantages of SHOT-IM over SHOT-IM ($\beta = 0$) validate the effectiveness of the diversity term $\mathcal{L}_{div}$. Incorporated with the labeling transfer strategy, SHOT-IM++ also obtains a better per-class accuracy than its variant SHOT-IM++ ($\beta = 0$). Secondly, SHOT ($\gamma_1 = 0$) performs worse than SHOT, indicating the effectiveness of the self-supervised pseudo labeling term in Eq. (7). Thirdly, SHOT ($\gamma_2 = 0$) performs worse than SHOT, indicating the effectiveness of the self-supervised rotation prediction term in Eq. (9). Two latter conclusions can also be drawn by comparing SHOT ($\gamma_1 = 0$) and SHOT ($\gamma_2 = 0$) with SHOT-IM. Also, it seems $\mathcal{L}_{ssl}^1$ contributes more than $\mathcal{L}_{ssl}^2$ within SHOT. Finally, the benefits of the labeling transfer strategy are also easily validated by comparing the values in the second column with those in the fourth column.

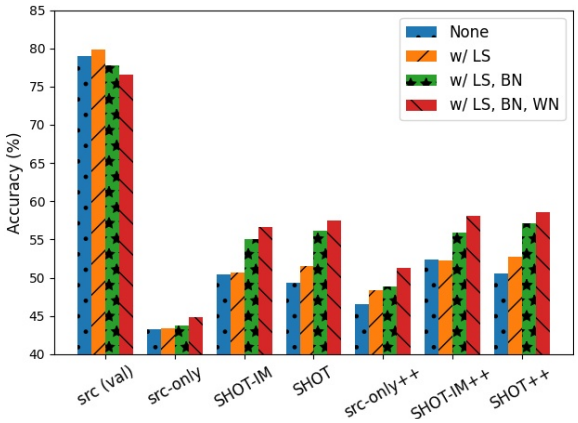
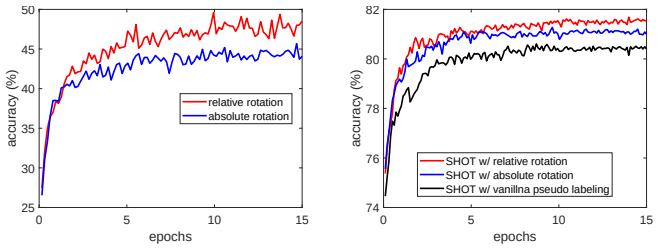



Fig. 5. Ablation study of label smoothing (LS) and batch normalization (BN) and weight normalization (WN) in the network for a 65-way classification UDA task Ar→Cl on **Office-Home**. 'src (val)' denotes the accuracy in the *source validation test*, and 'src-only' is short for source-model-only. Best viewed in colors.

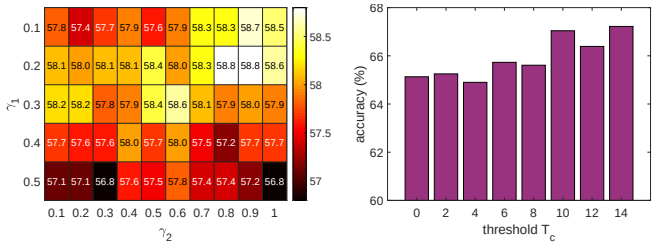
**Ablation study on network components.** As discussed in Section 3.8, we utilize label smoothing (LS), batch normalization (BN), and weight normalization (WN) during training the source model and learning the target feature encoder. We report the ablation study about network components in Fig. 5 to validate their contribution. First, using BN or using WN results in the decreasing accuracy over the source training set, which may be useful for generalization since we find in the second bin, high accuracy on the source test set corresponds to low accuracy on the training set. Second, LS improves accuracies of both the source training set and the target set, which is desirable for source model generation. Third, the higher accuracy the 'src-only' method obtains, the better results SHOT and its variants achieve. Generally, BN and WN are beneficial to domain adaptation. The improvements brought by BN are larger than those brought by WN.




(a) rotation prediction (b) target accuracy

Fig. 6. Accuracies of different variants during training for a 65-way classification UDA task Ar→Re on **Office-Home** (15 epochs).

**Discussion on loss functions.** To analyze the advantages of our proposed self-supervised loss functions in Eq. (7) and Eq. (9), we design one vanilla alternative for each function on the transfer task Ar→Re. As shown in Fig. 6(a), the proposed relative rotation prediction objective works better than the vanilla variant in terms of the rotation prediction accuracy. Besides, comparisons in terms of the semantic accuracy in Fig. 6(b) indicate that the proposed relative rotation prediction objective is also beneficial to UDA. Compared with SHOT w/ vanilla pseudo-labeling, SHOT always obtains better results along the training process, implying the superiority of the proposed self-supervised pseudo-labeling term.




(a) accuracy of Ar→Cl (UDA) (b) accuracy of Ar→Cl (PDA)

Fig. 7. Performance sensitivity of 3 parameters $\gamma_1$, $\gamma_2$, $T_c$ within SHOT.

**Parameter sensitivity.** To better understand the effects of $\gamma_1, \gamma_2$, we test their performance sensitivity in the UDA task Ar→Cl on **Office-Home** and show the results in Fig. 7(a). The accuracies around $\gamma_1 = 0.2, \gamma_2 = 0.6$ are not sensitive. Besides, we study the sensitivity of the threshold parameter $T_c$ for the PDA task Ar→Cl on **Office-Home** in Fig. 7(b). It shows that the accuracies around $T_c = 10$ are also not sensitive. Generally, the parameters within the proposed method i.e. SHOT are not sensitive.

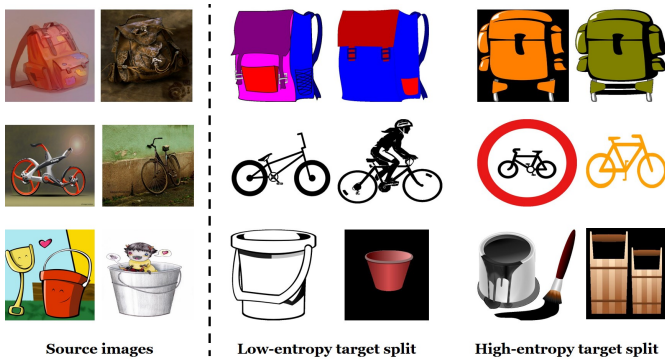


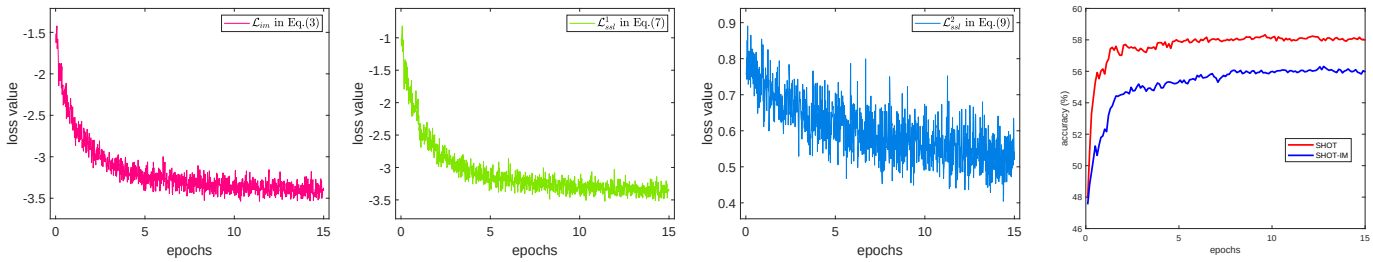

Fig. 8. Images of the source domain, the low-entropy target split, and the high-entropy target split for Ar→Cl on **Office-Home** (closed-set).

**Qualitative Study.** We randomly select some samples in the source domain, the low-entropy target split, and the high-entropy target split to provide some intuitive insights about the labeling transfer strategy. Particularly, we pick up two images from three representative classes, i.e., 'backpack', 'bike', and 'bucket', for the UDA task Ar→Cl on **Office-Home**, and show them in Fig. 8. It can be seen that the proposed strategy can well separate the easy samples from the hard samples in the target domain. Besides, the easy samples in the low-entropy target split are more trustworthy than the source samples for the hard samples in the high-entropy target split, making the proposed labeling transfer strategy understandable and effective.

**Training stability.** We investigate the accuracy and the values of three different objective functions within the optimization process in Fig. 9 on the UDA task Ar→Cl. It can be easily seen that the values of $\mathcal{L}_{im}$ and $\mathcal{L}^1_{ssl}$ quickly decrease and converge after nearly 8 epochs. The value of rotation prediction loss $\mathcal{L}^2_{ssl}$ also keeps decreasing but at a slow speed. As shown in Fig. 9(d), the accuracy varies following a very similar trend, i.e. growing up quickly and starting to

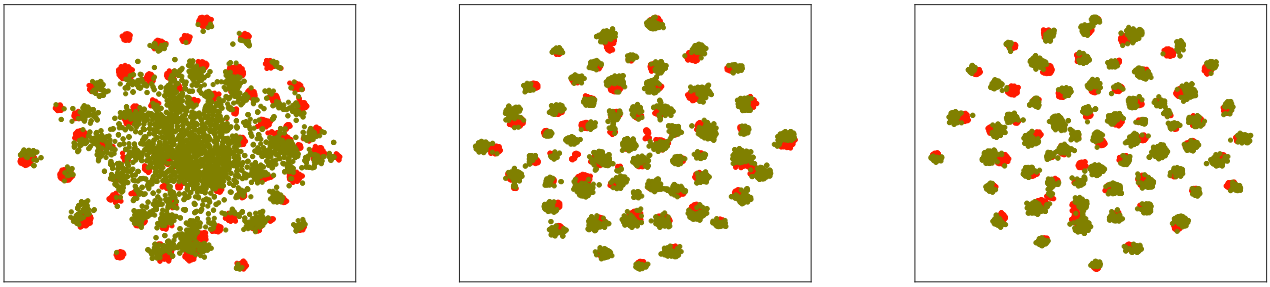
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2021.3103390, IEEE Transactions on Pattern Analysis and Machine Intelligence





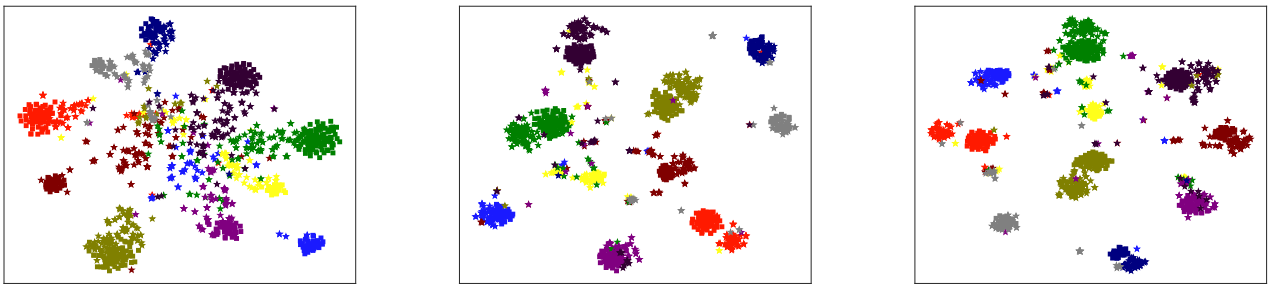

(a) value of $\mathcal{L}_{im}$ in Eq. (3) (b) value of $\mathcal{L}^1_{ssl}$ in Eq. (7) (c) value of $\mathcal{L}^2_{ssl}$ in Eq. (9) (d) Accuracy (%)

Fig. 9. Values of different loss functions and the accuracy during training for a 65-way classification UDA task Ar→Cl on **Office-Home** (15 epochs).



(a) Source-model-only (b) SHOT-IM (c) SHOT

Fig. 10. The t-SNE feature visualizations for a 65-way classification UDA task Ar→Cl on **Office-Home**. Circles in red denote unseen source data and circles in olive denote target data. Best viewed in colors.



(a) Source-model-only (b) SHOT-IM (c) SHOT

Fig. 11. The t-SNE feature visualizations for a 65-way classification UDA task Ar→Cl on **Office-Home**. For a better illustration, we choose features in the **first 10 classes** of each domain, and different color denotes different class. Best viewed in colors. [source in square, target in star]

converge after 6 epochs. Generally, the training procedure of SHOT is stable and effective.

**Feature visualization.** We provide the t-SNE visualizations [2] of the features learned by Source-model-only, SHOT-IM, and SHOT for the UDA task Ar→Cl on **Office-Home** in Fig. 10 and Fig. 11, respectively. As expected, both SHOT-IM and SHOT help align the target features with the source features in Fig. 10. Carefully looking at the semantic labels in Fig. 11, we find that SHOT outperforms SHOT-IM by semantically aligning features from different domains.

## 5 Conclusion

In this paper, we have proposed a generic representation learning framework called source hypothesis transfer (SHOT) for source data-absent unsupervised domain adaptation. SHOT merely needs the well-trained source model and offers the feasibility of unsupervised domain adaptation without access to the source data which may be private or decentralized. Specifically, SHOT learns the optimal target-specific feature learning module to fit the source hypothesis by exploiting information maximization and self-supervised learning. We further present a labeling transfer strategy and apply it to enhance SHOT to SHOT++, which exploits the intra-domain information via a semi-supervised algorithm. Experiments for both digit classification and object recognition verify that SHOT and SHOT++ can achieve results comparable to or even better than the state-of-the-art for three different unsupervised domain adaptation scenarios as well as the semi-supervised domain adaptation problem. In the future, we plan to apply the proposed methods to other visual tasks like semantic segmentation [43] and object detection [12].

## Acknowledgments

The authors would like to thank the reviewers and the associate editor for their valuable comments.

2. https://lvdmaaten.github.io/tsne/

## REFERENCES


[1] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2009.

[2] G. Csurka, "A comprehensive survey on domain adaptation for visual applications," in *Domain Adaptation in Computer Vision Applications*. Springer, 2017, pp. 1–35.

[3] M. Toldo, A. Maracani, U. Michieli, and P. Zanuttigh, "Unsupervised domain adaptation in semantic segmentation: a review," *arXiv preprint arXiv:2005.10876*, 2020.

[4] M. Long, Y. Cao, J. Wang, and M. Jordan, "Learning transferable features with deep adaptation networks," in *Proc. ICML*, 2015, pp. 97–105.

[5] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *Proc. CVPR*, 2017, pp. 7167–7176.

[6] K. Saito, K. Watanabe, Y. Ushiku, and T. Harada, "Maximum classifier discrepancy for unsupervised domain adaptation," in *Proc. CVPR*, 2018, pp. 3723–3732.

[7] Y. Zhang, P. David, and B. Gong, "Curriculum domain adaptation for semantic segmentation of urban scenes," in *Proc. ICCV*, 2017, pp. 2020–2030.

[8] J. Hoffman, E. Tzeng, T. Park, J.-Y. Zhu, P. Isola, K. Saenko, A. Efros, and T. Darrell, "Cycada: cycle-consistent adversarial domain adaptation," in *Proc. ICML*, 2018, pp. 1989–1998.

[9] Y. Zou, Z. Yu, B. Vijaya Kumar, and J. Wang, "Unsupervised domain adaptation for semantic segmentation via class-balanced self-training," in *Proc. ECCV*, 2018, pp. 289–305.

[10] X. Glorot, A. Bordes, and Y. Bengio, "Domain adaptation for large-scale sentiment classification: A deep learning approach," in *Proc. ICML*, 2011, pp. 513–520.

[11] M. Peng, Q. Zhang, Y.-g. Jiang, and X.-J. Huang, "Cross-domain sentiment classification with target domain specific information," in *Proc. ACL*, 2018, pp. 2505–2513.

[12] Y. Chen, W. Li, C. Sakaridis, D. Dai, and L. Van Gool, "Domain adaptive faster r-cnn for object detection in the wild," in *Proc. CVPR*, 2018, pp. 3339–3348.

[13] W. Li, F. Li, Y. Luo, and P. Wang, "Deep domain adaptive object detection: a survey," *arXiv preprint arXiv:2002.06797*, 2020.

[14] W. Deng, L. Zheng, Q. Ye, G. Kang, Y. Yang, and J. Jiao, "Image-image domain adaptation with preserved self-similarity and domain-dissimilarity for person re-identification," in *Proc. CVPR*, 2018, pp. 994–1003.

[15] Z. Wang, Z. Wang, Y. Zheng, Y. Wu, W. Zeng, and S. Satoh, "Beyond intra-modality: A survey of heterogeneous person re-identification," *arXiv preprint arXiv:1905.10048*, 2019.

[16] B. Sun, J. Feng, and K. Saenko, "Return of frustratingly easy domain adaptation," in *Proc. AAAI*, 2016, pp. 2058–2065.

[17] W. Zellinger, T. Grubinger, E. Lughofer, T. Natschläger, and S. Saminger-Platz, "Central moment discrepancy (cmd) for domain-invariant representation learning," in *Proc. ICLR*, 2017.

[18] X. Peng, Q. Bai, X. Xia, Z. Huang, K. Saenko, and B. Wang, "Moment matching for multi-source domain adaptation," in *Proc. ICCV*, 2019, pp. 1406–1415.

[19] A. Gretton, K. Borgwardt, M. Rasch, B. Schölkopf, and A. J. Smola, "A kernel method for the two-sample-problem," in *Proc. NeurIPS*, 2007, pp. 513–520.

[20] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. NeurIPS*, 2014, pp. 2672–2680.

[21] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan, "A theory of learning from different domains," *Machine Learning*, vol. 79, no. 1-2, pp. 151–175, 2010.

[22] I. Kuzborskij and F. Orabona, "Stability and hypothesis transfer learning," in *Proc. ICML*, 2013, pp. 942–950.

[23] J. Liang, D. Hu, and J. Feng, "Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation," in *Proc. ICML*, 2020.

[24] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *Proc. ICML*, 2015, pp. 1180–1189.

[25] W.-G. Chang, T. You, S. Seo, S. Kwak, and B. Han, "Domain-specific batch normalization for unsupervised domain adaptation," in *Proc. CVPR*, 2019, pp. 7354–7362.

[26] Y. Shi and F. Sha, "Information-theoretical learning of discriminative clusters for unsupervised domain adaptation," in *Proc. ICML*, 2012, pp. 1275–1282.

[27] W. Hu, T. Miyato, S. Tokui, E. Matsumoto, and M. Sugiyama, "Learning discrete representations via information maximizing self-augmented training," in *Proc. ICML*, 2017, pp. 1158–1167.

[28] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," in *Proc. ICLR*, 2018.

[29] D. Berthelot, N. Carlini, I. Goodfellow, N. Papernot, A. Oliver, and C. A. Raffel, "Mixmatch: A holistic approach to semi-supervised learning," in *Proc. NeurIPS*, 2019, pp. 5049–5059.

[30] K. Saenko, B. Kulis, M. Fritz, and T. Darrell, "Adapting visual category models to new domains," in *Proc. ECCV*, 2010, pp. 213–226.

[31] Z. Cao, M. Long, J. Wang, and M. I. Jordan, "Partial transfer learning with selective adversarial networks," in *Proc. CVPR*, 2018, pp. 2724–2732.

[32] K. Saito, D. Kim, S. Sclaroff, T. Darrell, and K. Saenko, "Semi-supervised domain adaptation via minimax entropy," in *Proc. ICCV*, 2019, pp. 8050–8058.

[33] D. Li, Y. Yang, Y.-Z. Song, and T. M. Hospedales, "Deeper, broader and artier domain generalization," in *Proc. ICCV*, 2017, pp. 5542–5550.

[34] B. Zadrozny, "Learning and evaluating classifiers under sample selection bias," in *Proc. ICML*, 2004, p. 114.

[35] M. Sugiyama, S. Nakajima, H. Kashima, P. V. Buenau, and M. Kawanabe, "Direct importance estimation with model selection and its application to covariate shift adaptation," in *Proc. NeurIPS*, 2008, pp. 1433–1440.

[36] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," *IEEE Transactions on Neural Networks*, vol. 22, no. 2, pp. 199–210, 2010.

[37] M. Long, J. Wang, G. Ding, J. Sun, and P. S. Yu, "Transfer feature learning with joint distribution adaptation," in *Proc. ICCV*, 2013, pp. 2200–2207.

[38] J. Liang, R. He, Z. Sun, and T. Tan, "Aggregating randomized clustering-promoting invariant projections for domain adaptation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 5, pp. 1027–1042, 2018.

[39] R. Gopalan, R. Li, and R. Chellappa, "Unsupervised adaptation across domain shifts by generating intermediate data representations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 11, pp. 2288–2302, 2013.

[40] B. Fernando, A. Habrard, M. Sebban, and T. Tuytelaars, "Unsupervised visual domain adaptation using subspace alignment," in *Proc. ICCV*, 2013, pp. 2960–2967.

[41] M. Long, Y. Cao, Z. Cao, J. Wang, and M. I. Jordan, "Transferable representation learning with deep adaptation networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 12, pp. 3071–3085, 2018.

[42] M. Long, Z. Cao, J. Wang, and M. I. Jordan, "Conditional adversarial domain adaptation," in *Proc. NeurIPS*, 2018, pp. 1640–1650.

[43] Y. Zhang, P. David, H. Foroosh, and B. Gong, "A curriculum domain adaptation approach to the semantic segmentation of urban scenes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 08, pp. 1823–1841, 2020.

[44] P. Panareda Busto and J. Gall, "Open set domain adaptation," in *Proc. ICCV*, 2017, pp. 754–763.

[45] K. You, M. Long, Z. Cao, J. Wang, and M. I. Jordan, "Universal domain adaptation," in *Proc. CVPR*, 2019, pp. 2720–2729.

[46] G. Kang, L. Jiang, Y. Yang, and A. G. Hauptmann, "Contrastive adaptation network for unsupervised domain adaptation," in *Proc. CVPR*, 2019, pp. 4893–4902.

[47] N. Courty, R. Flamary, D. Tuia, and A. Rakotomamonjy, "Optimal transport for domain adaptation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 9, pp. 1853–1865, 2017.

[48] M. Ghifary, W. B. Kleijn, M. Zhang, D. Balduzzi, and W. Li, "Deep reconstruction-classification networks for unsupervised domain adaptation," in *Proc. ECCV*, 2016, pp. 597–613.

[49] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan, "Domain separation networks," in *Proc. NeurIPS*, 2016, pp. 343–351.

[50] Z. Murez, S. Kolouri, D. Kriegman, R. Ramamoorthi, and K. Kim, "Image to image translation for domain adaptation," in *Proc. CVPR*, 2018, pp. 4500–4509.

[51] Z. Pei, Z. Cao, M. Long, and J. Wang, "Multi-adversarial domain adaptation," in *Proc. AAAI*, 2018, pp. 3934–3941.

[52] S. Cicek and S. Soatto, "Unsupervised domain adaptation via regularized conditional alignment," in *Proc. ICCV*, 2019, pp. 1416–1425.

[53] V. K. Kurmi and V. P. Namboodiri, "Looking back at labels: A class based domain adaptation technique," in *Proc. IJCNN*, 2019, pp. 1–8.

[54] F. M. Cariucci, L. Porzi, B. Caputo, E. Ricci, and S. R. Bulo, "Autodial: Automatic domain alignment layers," in *Proc. ICCV*, 2017, pp. 5077–5085.

[55] X. Wang, Y. Jin, M. Long, J. Wang, and M. I. Jordan, "Transferable normalization: Towards improving transferability of deep neural networks," in *Proc. NeurIPS*, 2019, pp. 1953–1963.

[56] K. Saito, Y. Ushiku, T. Harada, and K. Saenko, "Adversarial dropout regularization," in *Proc. ICLR*, 2018.

[57] S. Lee, D. Kim, N. Kim, and S.-G. Jeong, "Drop to adapt: Learning discriminative features for unsupervised domain adaptation," in *Proc. ICCV*, 2019, pp. 91–100.

[58] J. Yang, R. Yan, and A. G. Hauptmann, "Cross-domain video concept detection using adaptive svms," in *Proc. ACM-MM*, 2007, pp. 188–197.

[59] Y. Mansour, M. Mohri, and A. Rostamizadeh, "Domain adaptation with multiple sources," in *Proc. NeurIPS*, 2009, pp. 1041–1048.

[60] T. Tommasi, F. Orabona, and B. Caputo, "Safety in numbers: Learning categories from few examples with multi model knowledge transfer," in *Proc. CVPR*, 2010, pp. 3081–3088.

[61] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Proc. NeurIPS*, 2014, pp. 3320–3328.

[62] B. Chidlovskii, S. Clinchant, and G. Csurka, "Domain adaptation in the absence of source domain data," in *Proc. KDD*, 2016, pp. 451–460.

[63] J. Liang, R. He, Z. Sun, and T. Tan, "Distant supervised centroid shift: A simple and efficient approach to visual domain adaptation," in *Proc. CVPR*, 2019, pp. 2975–2984.

[64] R. Li, Q. Jiao, W. Cao, H.-S. Wong, and S. Wu, "Model adaptation: Unsupervised domain adaptation without source data," in *Proc. CVPR*, 2020, pp. 9641–9650.

[65] X. Peng, Z. Huang, Y. Zhu, and K. Saenko, "Federated adversarial domain adaptation," in *Proc. ICLR*, 2020.

[66] L. Jing and Y. Tian, "Self-supervised visual feature learning with deep neural networks: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[67] F. M. Carlucci, A. D'Innocente, S. Bucci, B. Caputo, and T. Tommasi, "Domain generalization by solving jigsaw puzzles," in *Proc. CVPR*, 2019, pp. 2229–2238.

[68] X. Zhai, A. Oliver, A. Kolesnikov, and L. Beyer, "S4l: Self-supervised semi-supervised learning," in *Proc. ICCV*, 2019, pp. 1476–1485.

[69] R. Zhang, P. Isola, and A. A. Efros, "Colorful image colorization," in *Proc. ECCV*, 2016, pp. 649–666.

[70] C. Doersch, A. Gupta, and A. A. Efros, "Unsupervised visual representation learning by context prediction," in *Proc. ICCV*, 2015, pp. 1422–1430.

[71] M. Noroozi and P. Favaro, "Unsupervised learning of visual representations by solving jigsaw puzzles," in *Proc. ECCV*, 2016, pp. 69–84.

[72] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proc. CVPR*, 2020, pp. 9729–9738.

[73] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. ICML*, 2020.

[74] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, "Deep clustering for unsupervised learning of visual features," in *Proc. ECCV*, 2018, pp. 132–149.

[75] M. Caron, P. Bojanowski, J. Mairal, and A. Joulin, "Unsupervised pre-training of image features on non-curated data," in *Proc. ICCV*, 2019, pp. 2959–2968.

[76] J. Xu, L. Xiao, and A. M. López, "Self-supervised domain adaptation for computer vision tasks," *IEEE Access*, vol. 7, pp. 156 694–156 706, 2019.

[77] Y. Sun, E. Tzeng, T. Darrell, and A. A. Efros, "Unsupervised domain adaptation through self-supervision," *arXiv preprint arXiv:1909.11825*, 2019.

[78] K. Saito, D. Kim, S. Sclaroff, and K. Saenko, "Universal domain adaptation through self supervision," *arXiv preprint arXiv:2002.07953*, 2020.

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

[80] W. Zhang, W. Ouyang, W. Li, and D. Xu, "Collaborative and adversarial network for unsupervised domain adaptation," in *Proc. CVPR*, 2018, pp. 3801–3809.

[81] Y. Grandvalet and Y. Bengio, "Semi-supervised learning by entropy minimization," in *Proc. NeurIPS*, 2005, pp. 529–536.

[82] R. Xu, G. Li, J. Yang, and L. Lin, "Larger norm more transferable: An adaptive feature norm approach for unsupervised domain adaptation," in *Proc. ICCV*, 2019, pp. 1426–1435.

[83] K. Saito, Y. Ushiku, and T. Harada, "Asymmetric tri-training for unsupervised domain adaptation," in *Proc. ICML*, 2017, pp. 2988–2997.

[84] R. Shu, H. H. Bui, H. Narui, and S. Ermon, "A dirt-t approach to unsupervised domain adaptation," in *Proc. ICLR*, 2018.

[85] D. Rukhovich and D. Galeev, "Mixmatch domain adaptaion: Prize-winning solution for both tracks of visda 2019 challenge," *arXiv preprint arXiv:1910.03903*, 2019.

[86] R. Müller, S. Kornblith, and G. E. Hinton, "When does label smoothing help?" in *Proc. NeurIPS*, 2019, pp. 4694–4703.

[87] A. Krause, P. Perona, and R. G. Gomes, "Discriminative clustering by regularized information maximization," in *Proc. NeurIPS*, 2010.

[88] T.-H. Vu, H. Jain, M. Bucher, M. Cord, and P. Pérez, "Advent: Adversarial entropy minimization for domain adaptation in semantic segmentation," in *Proc. CVPR*, 2019, pp. 2517–2526.

[89] B. Wallace and B. Hariharan, "Extending and analyzing self-supervised learning across domains," in *Proc. ECCV*, 2020, pp. 717–734.

[90] S. Mishra, K. Saenko, and V. Saligrama, "Surprisingly simple semi-supervised domain adaptation with pretraining and consistency," *arXiv preprint arXiv:2101.12727*, 2021.

[91] Z. Cao, M. Long, J. Wang, and M. I. Jordan, "Partial transfer learning with selective adversarial networks," in *Proc. CVPR*, 2018, pp. 2724–2732.

[92] S. Xie, Z. Zheng, L. Chen, and C. Chen, "Learning semantic representations for unsupervised domain adaptation," in *Proc. ICML*, 2018, pp. 5423–5432.

[93] T. Salimans and D. P. Kingma, "Weight normalization: A simple reparameterization to accelerate training of deep neural networks," in *Proc. NeurIPS*, 2016, pp. 901–909.

[94] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. ICML*, 2015, pp. 448–456.

[95] B. Gong, Y. Shi, F. Sha, and K. Grauman, "Geodesic flow kernel for unsupervised domain adaptation," in *Proc. CVPR*, 2012, pp. 2066–2073.

[96] H. Venkateswara, J. Eusebio, S. Chakraborty, and S. Panchanathan, "Deep hashing network for unsupervised domain adaptation," in *Proc. CVPR*, 2017, pp. 5018–5027.

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

[98] Z. Deng, Y. Luo, and J. Zhu, "Cluster alignment with a teacher for unsupervised domain adaptation," in *Proc. ICCV*, 2019, pp. 9944–9953.

[99] C.-Y. Lee, T. Batra, M. H. Baig, and D. Ulbricht, "Sliced wasserstein discrepancy for unsupervised domain adaptation," in *Proc. CVPR*, 2019, pp. 10 285–10 295.

[100] Z. Lu, Y. Yang, X. Zhu, C. Liu, Y.-Z. Song, and T. Xiang, "Stochastic classifiers for unsupervised domain adaptation," in *Proc. CVPR*, 2020, pp. 9111–9120.

[101] X. Chen, S. Wang, M. Long, and J. Wang, "Transferability vs. discriminability: Batch spectral penalization for adversarial domain adaptation," in *Proc. ICML*, 2019, pp. 1081–1090.

[102] Y. Zhang, T. Liu, M. Long, and M. Jordan, "Bridging theory and algorithm for domain adaptation," in *Proc. ICML*, 2019, pp. 7404–7413.

[103] S. Cui, S. Wang, J. Zhuo, L. Li, Q. Huang, and Q. Tian, "Towards discriminability and diversity: Batch nuclear-norm maximization under label insufficient situations," in *Proc. CVPR*, 2020, pp. 3941–3950.

[104] S. Cui, S. Wang, J. Zhuo, C. Su, Q. Huang, and Q. Tian, "Gradually vanishing bridge for adversarial domain adaptation," in *Proc. CVPR*, 2020, pp. 12 455–12 464.
[105] J. Zhang, Z. Ding, W. Li, and P. Ogunbona, "Importance weighted adversarial nets for partial domain adaptation," in *Proc. CVPR*, 2018, pp. 8156–8164.
[106] Z. Cao, K. You, M. Long, J. Wang, and Q. Yang, "Learning to transfer examples for partial domain adaptation," in *Proc. CVPR*, 2019, pp. 2985–2994.
[107] S. Li, C. H. Liu, Q. Lin, Q. Wen, L. Su, G. Huang, and Z. Ding, "Deep residual correction network for partial domain adaptation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.
[108] Z. Chen, C. Chen, Z. Cheng, B. Jiang, K. Fang, and X. Jin, "Selective transfer with reinforced transfer network for partial domain adaptation," in *Proc. CVPR*, 2020, pp. 12 706–12 714.
[109] J. Liang, Y. Wang, D. Hu, R. He, and J. Feng, "A balanced and uncertainty-aware approach for partial domain adaptation," in *Proc. ECCV*, 2020, pp. 123–140.
[110] C.-X. Ren, P. Ge, P. Yang, and S. Yan, "Learning target-domain-specific classifier for partial domain adaptation," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 5, pp. 1989–2001, 2020.
[111] R. Xu, Z. Chen, W. Zuo, J. Yan, and L. Lin, "Deep cocktail network: Multi-source unsupervised domain adaptation with category shift," in *Proc. CVPR*, 2018, pp. 3964–3973.
[112] M. Mancini, L. Porzi, S. Rota Bulò, B. Caputo, and E. Ricci, "Boosting domain adaptation by discovering latent domains," in *Proc. CVPR*, 2018, pp. 3771–3780.
[113] L. Yang, Y. Balaji, S.-N. Lim, and A. Shrivastava, "Curriculum manager for source selection in multi-source domain adaptation," in *Proc. ECCV*, 2020, pp. 608–624.
[114] C. Qin, L. Wang, Q. Ma, Y. Yin, H. Wang, and Y. Fu, "Opposite structure learning for semi-supervised domain adaptation," in *Proc. SDM*, 2021, pp. 576–584.
[115] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
[116] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. CVPR*, 2016, pp. 770–778.
[117] J. Liang, D. Hu, R. He, and J. Feng, "Distill and fine-tune: Effective adaptation from a black-box source model," *arXiv preprint arXiv:2104.01539*, 2021.
[118] D. Li and T. Hospedales, "Online meta-learning for multi-source and semi-supervised domain adaptation," in *Proc. ECCV*, 2020, pp. 382–403.
[119] N. Venkat, J. N. Kundu, D. K. Singh, A. Revanur, and R. V. Babu, "Your classifier can secretly suffice multi-source domain adaptation," in *Proc. NeurIPS*, 2020, pp. xx–xx.
[120] Z. Huang, K. Sheng, W. Dong, X. Mei, C. Ma, F. Huang, D. Zhou, and C. Xu, "Effective label propagation for discriminative semi-supervised domain adaptation," *arXiv preprint arXiv:2012.02621*, 2020.

**Jian Liang** received the B.E. degree in Electronic Information and Technology from Xi'an Jiaotong University and Ph.D. degree in Pattern Recognition and Intelligent Systems from NLPR, CASIA in July 2013, and January 2019, respectively. He is a research fellow at National University of Singapore from June 2019 to April 2021. Now he joins NLPR as an associated professor. His research interests focus on transfer learning, pattern recognition, and computer vision.

**Dapeng Hu** received the B.Sc. degree in Electronic Information Science and Technology from Nanjing University in 2017. He is currently pursuing a Ph.D. degree at National University of Singapore. His research interests focus on domain adaptation, self-supervised learning, and computer vision.

**Yunbo Wang** received the B.E. degree and the M.S. degree in Electronics and Information Engineering from Sichuan University in 2012 and 2015, respectively, and the Ph.D. degree in pattern recognition and intelligent systems from CASIA, in January 2020. He is currently a postdoctoral researcher at Peking University. His research interests include image/ video retrieval, cross-modal analysis and reasoning, pattern recognition, and computer vision.

**Ran He** received the B.E. degree in Computer Science from Dalian University of Technology, the M.S. degree in Computer Science from Dalian University of Technology, and the Ph.D. degree in Pattern Recognition and Intelligent Systems from CASIA in 2001, 2004, and 2009, respectively. Since September 2010, Dr. He has joined NLPR where he is currently a full Professor. He has published more than 150 papers in international journals and conferences, including reputable international journals such as IEEE TPAMI, IEEE TIP, IEEE TNN, IEEE TCSVT, IEEE TIFS and top-level international conferences like CVPR, ICCV, NIPS, ECCV, AAAI, and IJCAI. He is serving as the Associate Editor of Pattern Recognition, Neurocomputing. He has served as the Area Chair, Senior PC of international conferences like ICPR and IJCAI. He is the Fellow of IAPR and a Senior Member of IEEE. His research interests focus on information-theoretic learning, pattern recognition, and computer vision.

**Jiashi Feng** received the B.Eng. degree from the University of Science and Technology, China, in 2007, and the Ph.D. degree from the National University of Singapore in 2014. He was a Postdoctoral Researcher with the University of California from 2014 to 2015. He is currently an Assistant Professor with the Department of Electrical and Computer Engineering with the National University of Singapore. His current research interests include machine learning and computer vision techniques for large-scale data analysis. Specifically, he has done work in object recognition, deep learning, machine learning, high-dimensional statistics, and big data analysis.