Deep Co-Training with Task Decomposition for Semi-Supervised Domain Adaptation

Luyu Yang[1], **Yan Wang**[2], **Mingfei Gao**[3],
Abhinav Shrivastava[1], **Kilian Q. Weinberger**[2], **Wei-Lun Chao**[4], **Ser-Nam Lim**[5]

[1]University of Maryland [2]Cornell University [3]Salesforce Research
[4]Ohio State University [5]Facebook AI

Abstract

*Semi-supervised domain adaptation (SSDA) aims to adapt models trained from a labeled source domain to a different but related target domain, from which unlabeled data and a small set of labeled data are provided. Current methods that treat source and target supervision without distinction overlook their inherent discrepancy, resulting in a source-dominated model that has not effectively use the target supervision. In this paper, we argue that the labeled target data needs to be distinguished for effective SSDA, and propose to explicitly decompose the SSDA task into two sub-tasks: a semi-supervised learning (SSL) task in the target domain and an unsupervised domain adaptation (UDA) task across domains. By doing so, the two sub-tasks can better leverage the corresponding supervision and thus yield very different classifiers. To integrate the strengths of the two classifiers, we apply the well established co-training framework, in which the two classifiers exchange their high confident predictions to iteratively "teach each other" so that both classifiers can excel in the target domain. We call our approach **De**ep **Co**-training with **Ta**sk decomposition (**DeCoTa**). DeCoTa requires no adversarial training and is easy to implement. Moreover, DeCoTa is well founded on the theoretical condition of when co-training would succeed. As a result, DeCoTa achieves state-of-the-art results on several SSDA datasets, outperforming the prior art by a notable 4% margin on DomainNet. Code is available at* https://github.com/LoyoYang/DeCoTa.

1. Introduction

Domain adaptation (DA) aims to adapt machine learned models from a source domain to a related but different target domain [4, 14, 53, 13]. DA is particularly important in settings where labeled target data is hard to obtain, but labeled source data is plentiful [63, 41, 21], *e.g.*, adaptation from synthetic to real images [21, 55, 48, 47, 56] and adaptation to a new or rare environment [10, 69, 54, 9]. Most of the existing works focus on the unsupervised domain adaptation (UDA) setting, in which the target domain is completely unlabeled. Several recent works, however, show that adding merely a tiny amount of target labeled data (*e.g.*, just one labeled image per class) can notably boost the performance [51, 26, 45, 1, 31, 30, 12, 74], suggesting that this setting may be more promising for domain adaptation to succeed. In this paper, we thus focus on the latter setting, which is referred to as semi-supervised domain adaptation (SSDA).

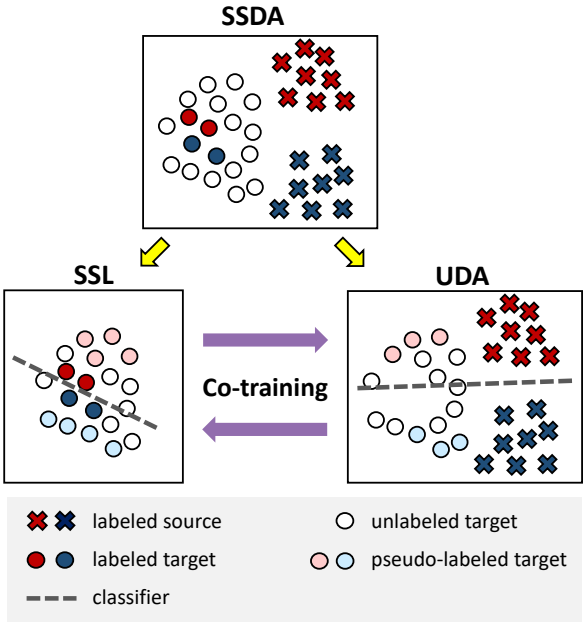

Figure 1: **De**ep **Co**-training with **Ta**sk decomposition (**DeCoTa**). We decompose semi-supervised domain adaptation (SSDA) into two sub-tasks: semi-supervised learning (SSL) in the target domain, and unsupervised DA (UDA) across domains. The two sub-tasks offer different pseudo-label confidences to the unlabeled data (light blue & light red circles), which we leverage via co-training: exchanging their high confident predictions to teach each other.

Despite the seemingly nuanced difference between the two settings, methods that are effective for SSDA and UDA can vary substantially. For instance, [51] showed that directly combining the labeled source and labeled target data and then applying popular UDA algorithms like domain adversarial learning [13] or entropy minimization [16] can hardly improve the performance. In other words, the labeled target data have not been effectively used. Existing methods [51, 45, 26] therefore propose additional objectives to strengthen the influence of labeled target data in SSDA.

Intrigued by these findings, we investigate the characteristics of SSDA further and emphasize two fundamental challenges. First, the amount of labeled source data is much larger than that of labeled target data. Second, the two data are inherently different in their distributions. **A single classifier** learned together with both sources of supervision is thus easily dominated by the labeled source data and is unable to take advantage of the additional labeled target data.

To resolve this issue, we propose to explicitly decompose the two sources of supervision and learn **two distinct classifiers** whose goals are however shared: to classify well on the unlabeled target data. To this end, we pair the labeled source data and the unlabeled target data to learn one classifier, which is essentially a UDA task. For the other classifier, we pair the labeled and unlabeled target data, which is essentially a semi-supervised learning (SSL) task. That is, **we explicitly decompose SSDA into two well-studied tasks.**

For each sub-task, one may apply any existing algorithms independently. In this paper, we however investigate the idea of learning the two classifiers jointly for two compelling reasons. First, the two tasks share the same goal and same unlabeled data, meaning that they are *correlated*. Second, learning with distinct labeled data implies that the two classifiers will converge differently in what types of mistakes they make and on which samples they are confident and correct, meaning that they are *complementary* to each other.

We therefore propose to learn the two classifiers jointly via co-training [6, 2, 8][1], which is arguably one of the most established algorithm for learning with multi views: in our case, two correlating and complementary tasks. The approach is straightforward: train a separate classifier on each task using its labeled data, and use them to create pseudo-labels for the unlabeled data. As the two classifiers are trained with distinct supervision, they will yield different predictions. In particular, there will be samples that only one classifier is confident about (and more likely to be correct). By labeling these samples with the confident classifier's predictions and adding them to the training set of the other classifier to re-train on, the two classifiers are essentially "teaching each other" to improve. To this end, we employ a simple *pseudo-labeling-based algorithm with deep learning*, similar to [5], to train each classifier. Pseudo-labeling-based algorithms have been shown powerful for both the UDA and SSL tasks [70, 27]. In other words, we can apply the same algorithm for both sub-tasks, greatly simplifying our overall framework which we name **DECOTA**: **De**ep **Co**-training with **Ta**sk Decomposition (Fig. 1 gives an illustration).

We evaluate DECOTA on two benchmark datasets for SSDA: DomainNet [41] and Office-home [66]. While very simple to implement and without any adversarial training [51, 45], DECOTA significantly outperforms the state-of-the-art results [45, 26] on DomainNet by over 4% and is on a par with them on Office-home. *We attribute this to the empirical evidence that our task decomposition fits the theoretical condition of relaxed ϵ-expandability [8, 2], which is sufficient for co-training to succeed.* Another strength of DECOTA is that it requires no extra learning process like feature decomposition to create views from data [8, 44, 7]. To the best of our knowledge, our paper is the first to enable deep learning with co-training on SSDA.

The contributions of this work are as follow. (1) We explicitly decompose the two very different sources of supervision, labeled source and labeled target data, in SSDA. (2) We present DECOTA, a simple deep learning based co-training approach for SSDA to jointly learn two classifiers, one for each supervision. (3) we provide intermediate results and insights that illustrate why DECOTA works. Specifically, we show that DECOTA satisfies the ϵ-expandability requirement [2] of co-training. (4) Lastly, we support this work with strong empirical results that outperform state-of-the-art.

2. Related Work

Unsupervised domain adaptation (UDA). UDA has been studied extensively. Many methods [33, 57, 65] matched the feature distributions between domains by minimizing their divergence. One mainstream approach is by domain adversarial learning [13, 21, 68, 40, 67, 71, 69, 73]. More recent works [52, 53, 29, 57] learn features based on the cluster assumption [16]: classifier boundaries should not cross high density target data regions. For example, [52, 53] attempted to push target features away from the boundary, using minimax training. Some other approaches employ self-training with pseudo-labeling [28, 37, 38, 3] to progressively label unlabeled data and use them to fine-tune the model [7, 25, 78, 24, 62, 32, 23, 27]. A few recent methods use MIXUP [76], but mainly to augment adversarial learning based UDA approaches (*e.g.*, [13]) by stabilizing the domain discriminator [61, 71] or smoothing the predictions [36, 72]. In contrast, we apply MIXUP to create better pseudo-labeled data for co-training, without adversarial learning.

Semi-supervised domain learning (SSDA). SSDA attracts less attention in DA, despite its promising scenario in balancing accuracy and labeling effort. With few labeled target data, SSDA can quickly reshape the class boundaries to boost

[1] We note that, co-training [6] and co-teaching [17] share similar concepts but are fundamentally different. See 2 for a discussion.

the accuracy [51, 45]. Many SSDA works are proposed prior to deep learning [74, 31, 20, 42], matching features while maintaining accuracy on labeled target data. [1, 64] employed knowledge distillation [19] to regularize the training on labeled target data. More recent works use deep learning, and find that the popular UDA principle of aligning feature distributions could fail to learn discriminative class boundaries in SSDA [51]. [51] thus proposed to gradually move the class prototypes (used to derive class boundaries) to the target domain in a minimax fashion; [45] introduced opposite structure learning to cluster target data and scatter source data to smooth the process of learning class boundaries. Both works [45, 51] and [26] concatenate the target labeled data with the source data to expand the labeled data. [30] incorporates meta-learning to search for better initial condition in domain adaptation. SSDA is also related to [60, 43], in which active learning is incorporated to label data for improving domain adaptation.

Co-training. Co-training, a powerful semi-supervised learning (SSL) method proposed in [6], looks at the available data with two views from which two models are trained interactively. By adding the confident predictions of one model to the training set of the other, co-training enables the models to "teach each other". There were several assumptions to ensure co-training's effectiveness [6], which were later relaxed by [2] with the notion of ϵ-expandability. [8] broadened the scope of co-training to a single-view setting by learning to decompose a fixed feature representation into two artificially created views; [7] subsequently extended this framework to use co-training for (semi-supervised) domain adaptation[2]. A recent work [44] extended co-training to deep learning models, by encouraging two models to learn different features and behave differently on single-view data. One novelty of DECOTA is that it works with single-view data (both the UDA and SSL tasks are looking at images) but requires no extra learning process like feature decomposition to artificially create views from such data [8, 44, 7].

Co-training vs. co-teaching. Co-teaching [17] was proposed for learning with noisy data, which shares a similar procedure to co-training by learning two models to filter out noisy data for each other. There are several key differences between them and **DECOTA** *is based on co-training*. As in [17], co-teaching is designed for supervised learning with noisy labels, while co-training is for learning with unlabeled data by leveraging two views. DECOTA decomposes SSDA into two tasks (two views) to leverage their difference to improve the performance — the core concept of co-training [7]. In contrast, co-teaching does not need two views. Further, co-teaching relies on the memorization of neural nets to select small loss samples to teach the other classifiers, while DECOTA selects high confident ones from unlabeled data.

[2]Similar to [45, 51], [7] simply concatenated the target labeled data with the source data to expand the labeled data.

3. Deep Co-training with Task Decomposition

3.1. Approach Overview

Co-training strategies have traditionally been applied to data with two views, e.g., audio and video, or webpages with HTML source and link-graph, after which a classifier is trained in each view and they teach each other on the unlabeled data. This is the original formulation from Blum and Mitchell [6], which is later extended to single-view data by [8] for linear models and by [44] for deep neural networks. Both methods require additional objective functions or tasks (*e.g.*, via generating adversarial examples [15]) to learn to create *artificial* views such that co-training can be applied.

In this paper, we have however discovered that in semi-supervised domain adaptation (SSDA), one can actually conduct co-training using single-view data (all are images) without such an additional learning subroutine. The key is to leverage the inherent discrepancy of the labeled data (*i.e.*, supervision) provided in SSDA: the labeled data from the source domain, $D_{\mathcal{S}} = \{(s_i, y_i)\}_{i=1}^{N_S}$, and the labeled data from the target domain, $D_{\mathcal{T}} = \{(t_i, y_i)\}_{i=1}^{N_T}$, which is usually much smaller than $D_{\mathcal{S}}$. By combining each of them with the unlabeled samples from the target domain, $D_{\mathcal{U}} = \{u_i\}_{i=1}^{N_U}$, we can construct two sub-tasks in SSDA:

- an **unsupervised domain adaptation (UDA)** task that trains a model $\boldsymbol{w}_g$ using $D_{\mathcal{S}}$ and $D_{\mathcal{U}}$,
- a **semi-supervised learning (SSL)** task that trains another model $\boldsymbol{w}_f$ using $D_{\mathcal{T}}$ and $D_{\mathcal{U}}$.

We learn both models by mini-batch stochastic gradient descent (SGD). At every iteration, we sample three data sets, $S = \{(s_b, y_b)\}_{b=1}^{B}$ from $D_{\mathcal{S}}$, $T = \{(t_b, y_b)\}_{b=1}^{B}$ from $D_{\mathcal{T}}$, and $U = \{u_b\}_{b=1}^{B}$ from $D_{\mathcal{U}}$, where B is the mini-batch size. We can then predict on U using the the two models $\boldsymbol{w}_g$ and $\boldsymbol{w}_f$, creating the pseudo-label sets $U^{(f)}$ and $U^{(g)}$ that will be used to update $\boldsymbol{w}_f$ and $\boldsymbol{w}_g$,

$$
\begin{aligned}
U^{(f)} =& \{(u_b, \hat{y}_b = \arg\max_c p(c|u_b; \boldsymbol{w}_g)); \\
& \text{if } \max_c p(c|u_b; \boldsymbol{w}_g) > \tau\}, \\
U^{(g)} =& \{(u_b, \hat{y}_b = \arg\max_c p(c|u_b; \boldsymbol{w}_f)); \\
& \text{if } \max_c p(c|u_b; \boldsymbol{w}_f) > \tau\},
\end{aligned} \tag{1}
$$

where u_b is an unlabeled sample drawn from U, $p(c|u_b; \cdot)$ is the predicted probability for a class c, and τ is the threshold for pseudo-label selection. In other words, we use one model's (say $\boldsymbol{w}_g$) high confident prediction to create pseudo-labels for u_b, which is then included in $U^{(f)}$ that will be used to train the other model $\boldsymbol{w}_f$. By looking at $U^{(f)}$ and $U^{(g)}$ jointly, we are indeed asking one model to simultaneously be a *teacher* and a *student*: it provides confident pseudo-labels for the other model to learn from, and learns from the other model's confident pseudo-labels.

Algorithm 1: The **DECOTA** algorithm

Input : $\boldsymbol{w}_f$ and $\boldsymbol{w}_g$, learning rate η, batch size B, iteration $N_{\max}$, beta distribution coefficient α, confidence threshold τ, data $D_{\mathcal{S}}$, $D_{\mathcal{T}}$, $D_{\mathcal{U}}$;

for $n \leftarrow 1$ **to** $N_{\max}$ **do**

  Sample $S = \{(s_b, y_b)\}_{b=1}^{B}$ from $D_{\mathcal{S}}$,

  Sample $T = \{(t_b, y_b)\}_{b=1}^{B}$ from $D_{\mathcal{T}}$,

  Sample $U = \{u_b\}_{b=1}^{B}$ from $D_{\mathcal{U}}$;

  Set $U^{(f)} = \emptyset$, $U^{(g)} = \emptyset$;

  for $b \leftarrow 1$ **to** B **do**

    if $\max_c p(c|u_b; \boldsymbol{w}_g) > \tau$ **then**

      Update $U^{(f)} \leftarrow U^{(f)} + \{(u_b, \hat{y}_b)\}$, $\hat{y}_b = \arg\max_c p(c|u_b; \boldsymbol{w}_g)$;

    end

    if $\max_c p(c|u_b; \boldsymbol{w}_f) > \tau$ **then**

      Update $U^{(g)} \leftarrow U^{(g)} + \{(u_b, \hat{y}_b)\}$, $\hat{y}_b = \arg\max_c p(c|u_b; \boldsymbol{w}_f)$;

    end

  end

  Obtain $\tilde{U}^{(f)} = \{\text{MIXUP}(U_i^{(f)}, T_i; \alpha)\}_{i=1}^{|U^{(f)}|}$;

  Obtain $\tilde{U}^{(g)} = \{\text{MIXUP}(U_i^{(g)}, S_i; \alpha)\}_{i=1}^{|U^{(g)}|}$;

  Update $\boldsymbol{w}_f \leftarrow \boldsymbol{w}_f - \eta\left(\nabla\mathcal{L}(\boldsymbol{w}_f, T) + \nabla\mathcal{L}(\boldsymbol{w}_f, \tilde{U}^{(f)})\right)$;

  Update $\boldsymbol{w}_g \leftarrow \boldsymbol{w}_g - \eta\left(\nabla\mathcal{L}(\boldsymbol{w}_g, S) + \nabla\mathcal{L}(\boldsymbol{w}_g, \tilde{U}^{(g)})\right)$;

end

Output : $\boldsymbol{w}_f$ and $\boldsymbol{w}_g$ (for model ensemble).

We call this approach **DECOTA**, which stands for Deep Co-training with Task Decomposition. In the following, we will discuss how to improve the pseudo-label quality (*i.e.*, its coverage and accuracy) for **DECOTA**, and provide in-depth analysis why DECOTA works.

3.1.1 DECOTA with High-quality Pseudo-labels

The pseudo-labels acquired from each model are understandably noisy. At the beginning of the training, this problem is especially acute, and affects the efficacy of the model as the training progresses. Our experience shows that mitigation is necessary to handle noise in the pseudo-labels to further enhance **DECOTA**, for which we follow recent works of SSL [5] to apply MIXUP [76, 35]. MIXUP is an operation to construct virtual examples by convex combinations. Given two labeled examples (x_1, y_1) and (x_2, y_2), we define $\text{MIXUP}\left((x_1, y_1), (x_2, y_2); \alpha\right)$

$$\lambda \sim \text{Beta}(\alpha, \alpha),$$
$$\tilde{x} = (1-\lambda)x_1 + \lambda x_2, \qquad \tilde{y} = (1-\lambda)\boldsymbol{e}_{y_1} + \lambda \boldsymbol{e}_{y_2} \quad (2)$$

to obtain a virtual example $(\tilde{x}, \tilde{y})$, where $\boldsymbol{e}_y$ is a one-hot vector with the y^{th} element being 1. λ controls the degree of MIXUP while Beta refers to the standard beta distribution.

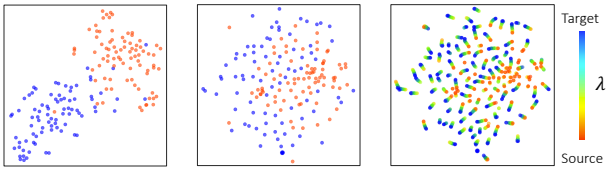

Figure 2: t-SNE visualization of S (red dots, sampled from $D_{\mathcal{S}}$) and U (blue dots, sampled from $D_{\mathcal{U}}$): (a) before and (b) after including MIXUP in calculating the projection; (c) t-SNE of S, U, and $\text{MIXUP}(S, U)$. We see a clear data transition along λ.

We perform MIXUP between labeled and pseudo-labeled data: *i.e.*, between samples in $U^{(f)}$ and T, and between samples in $U^{(g)}$ and S to obtain two sets of virtual examples $\tilde{U}^{(f)}$ and $\tilde{U}^{(g)}$. We then update $\boldsymbol{w}_f$ and $\boldsymbol{w}_g$ by SGD,

$$\boldsymbol{w}_g \leftarrow \boldsymbol{w}_g - \eta\left(\nabla\mathcal{L}(\boldsymbol{w}_g, S) + \nabla\mathcal{L}(\boldsymbol{w}_g, \tilde{U}^{(g)})\right), \quad (3)$$
$$\boldsymbol{w}_f \leftarrow \boldsymbol{w}_f - \eta\left(\nabla\mathcal{L}(\boldsymbol{w}_f, T) + \nabla\mathcal{L}(\boldsymbol{w}_f, \tilde{U}^{(f)})\right),$$

where η is the learning rate and $\mathcal{L}$ is the averaged loss over examples. We use the cross-entropy loss.

In our experiments, we have found that MIXUP can

- effectively **denoise** an incorrect pseudo-label by mixing it with a correct one (from S or T). The resulting $\tilde{y}$ at least contains a λ portion of correct labels;
- smoothly **bridge** the domain gap between U and S. This is done by interpolating between $U^{(g)}$ and S. The resulting $\tilde{x}$ can be seen as an intermediate example between domains.

In other words, MIXUP encourages the models to behave linearly between accurately labeled and pseudo-labeled data, which reduces the undesirable oscillations caused by noisy pseudo-labels and stabilizes the predictions across domains. We note that, our usage of MIXUP is fundamentally different from [61, 71, 36, 72] that employed MIXUP as auxiliary losses to augment existing DA algorithms like [13].

We illustrate this in Fig. 2. A model pre-trained on $D_{\mathcal{S}}$ is used to generate feature embeddings. We then employ t-SNE [34] to perform two tasks simultaneously, namely clustering the embedded samples as well as projecting them into a 2D space for visualization. In (a), only S sampled from $D_{\mathcal{S}}$ and U sampled from $D_{\mathcal{U}}$ are embedded, while in (b) and (c), additional samples from MIXUP of S and U were added to the fold to influence t-SNE's clustering step. (b) shows only the finally projected S and U samples afterwards while (c) shows the additional projected MIXUP samples as a function of λ. One can easily see that MIXUP effectively closes the gap between the source and target domain. We summarize our proposed algorithm in Algorithm 1.

3.2. Constraints for Effective Co-training

In DECOTA, we perform co-training via a decomposition of tasks on single-view data. To explain further why

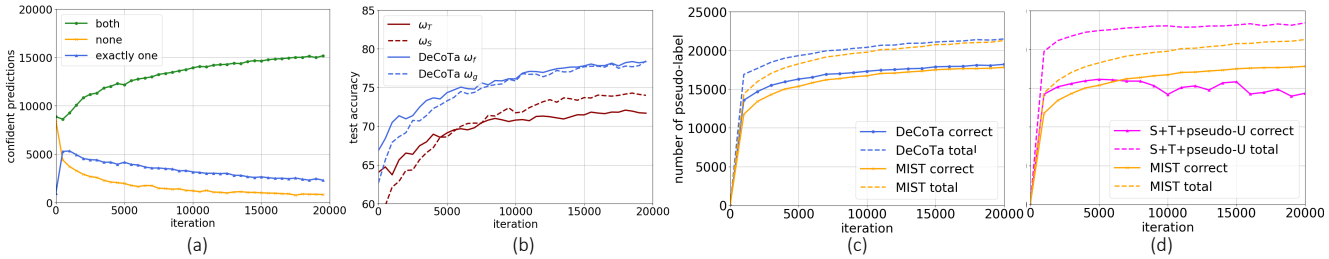

Figure 3: **Analysis on the two-task decomposition.** We use DomainNet [41] (Real to Clipart; three-shot). (a) We show the number of test examples that *both*, *exactly one*, and *none* of the models have high confidence on (in total, $18,325$). The two tasks hold unique expertise (*i.e.*, there is a 14% portion of the data that exactly one view is confident on), satisfying the condition of co-training in Eq. (6). (b) We show the power of co-training: the same tasks without co-training perform worse, indicating that the models benefit from each other. See Section 3.2 for details. The analysis is on DomainNet (R to C; three-shot) and we will clarify it. We further analyze pseudo-labels in (c) and (d). For every 1K iterations (*i.e.*, 24K unlabeled data with possible repetition), we accumulate the number of data that have confident (> 0.5) and correct predictions by at least one classifier. See Section 4 for details. (c) Comparison of pseudo-label quantity and quality using DECOTA vs. MIST. (d) MIST vs. self-training (S+T+pseudo-U). It can be observed that DECOTA has the largest number of correct pseudo-labels.

DECOTA works, we provide analysis in this subsection on the difference made by splitting the SSDA problem into two tasks for co-training. That is, we would like to verify that the decomposition leads to two tasks that fit into the assumption of co-training [2]. To begin with, we train two models: one model, $\boldsymbol{w}_S$, is trained with S and $\tilde{U}^{(S)}$ while the other model, $\boldsymbol{w}_T$, is trained with T and $\tilde{U}^{(T)}$. $\tilde{U}^{(S)}$ is obtained from applying $\boldsymbol{w}_S$ to U for pseudo-labels, follow by MIXUP with S. The same definition goes for $\tilde{U}^{(T)}$. Essentially, both the UDA and SSL task prepare their own pseudo-labels *independently* using their respective model in a procedure that is similar to self-training [28, 37, 38, 3].

After training, we apply $\boldsymbol{w}_T$ to the entire $D_{\mathcal{U}}$ and compute for each $u \in D_{\mathcal{U}}$ the binary confidence indicator

$$h_T(u) = \begin{cases} 1 & \text{if } \max_c p(c|u; \boldsymbol{w}_T) > \tau, \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

Here, high confident examples will get a value 1, otherwise 0. We also apply $\boldsymbol{w}_S$ to $D_{\mathcal{U}}$ to obtain $h_S(u)$. Denote by $\bar{h}_T(u) = 1 - h_T(u)$ the not function of $h_T(u)$, we compute the following three indicators to summarize the entire $D_{\mathcal{U}}$

$$\begin{aligned} h_{\text{both}} &: \sum_{u \in D_{\mathcal{U}}} h_T(u) h_S(u), \\ h_{\text{one}} &: \sum_{u \in D_{\mathcal{U}}} h_T(u) \bar{h}_S(u) + \bar{h}_T(u) h_S(u), \\ h_{\text{none}} &: \sum_{u \in D_{\mathcal{U}}} \bar{h}_T(u) \bar{h}_S(u), \end{aligned} \tag{5}$$

corresponding to the number of examples that *both*, *exactly one*, and *none* of the models have high confidence on, respectively. Intuitively, if the two models are exactly the same, h_{one} will be 0, meaning that they are either both confident or not on an example. On the contrary, if the two models are well optimized but hold their specialties, both h_{one} and h_{both} will be of high values and h_{none} will be low.

We ran the study on DomainNet [41], in which we use *Real* as source and *Clipart* as target. (See Section 4 for details.) We consider a 126-class classification problem, in which $|D_{\mathcal{S}}| = 70,358$, $|D_{\mathcal{U}}| = 18,325$, and $|D_{\mathcal{T}}| = 378$ (*i.e.*, a three-shot setting where each class in the target domain is given three labeled samples). We initialize $\boldsymbol{w}_S$ and $\boldsymbol{w}_T$ with a ResNet [18] pre-trained on $D_{\mathcal{S}}$, and evaluate Eq. (4) and Eq. (5) every 500 iterations (with a $\tau = 0.5$ confidence threshold in selecting pseudo-labels.).

Fig. 3 (a) shows the results. *The two models do hold their specialties (*i.e.*, yield different high-confident predictions).* Even at the end of training, there is a 14% portion of data that one model is confident on but not the other (the blue curve). Thus, if we can properly fuse their specialties during training — one model provides the pseudo-labels to the data on which the other model is uncertain — we are likely to jointly learn stronger models at the end.

This is indeed the core idea of our co-training proposal. Theoretically, the two "views" (or, tasks in our case) must satisfy certain conditions, *e.g.*, ϵ-expandability [2]. [8, 7] relaxed it and only needed the expanding condition to hold on average in the unlabeled set, which can be formulated as follows, using h_{both}, h_{one}, and h_{none}

$$h_{\text{one}} \geq \epsilon \min(h_{\text{both}}, h_{\text{none}}). \tag{6}$$

To satisfy Eq. 6, there must be sufficient examples that exactly one model is confident on so that the two models can benefit from teaching each other. Referring to Fig. 3 (a) again, our two tasks consistently hold a ϵ around 2 after the first 500 iterations (*i.e.*, after the models start to learn the task-specific idiosyncrasies), suggesting the feasibility of applying co-training to our decomposition. The power of co-training is clearly illustrated in Fig. 3 (b). The two models without co-training, $\boldsymbol{w}_T$ and $\boldsymbol{w}_S$, perform worse than their co-training counterparts, $\boldsymbol{w}_f$ and $\boldsymbol{w}_g$ (see Section 3.1, Eq. (1), Eq. (3)), even using the same architecture and data.

Table 1: Comparing with deep co-training methods [44] for SSDA on DomainNet, 3-shot. (See Section 4 for details.)

Method	R to C	R to P	P to C	C to S	S to P	R to S	P to R	Mean
Deep Co-Training [44] w/o MIXUP	73.7	67.6	73.2	63.9	66.7	64.1	79.3	69.7
Deep Co-Training [44] with MIXUP	74.2	69.1	72.3	64.1	67.9	65.1	79.4	70.3
DECOTA	80.4	75.2	78.7	68.6	72.7	71.9	81.5	75.6

3.3. Comparing to Other Co-training Approaches

With our approach outlined, it is worthwhile to contrast DECOTA with prior co-training work in domain adaptation. In particular, DECOTA is notably different from the approach known as Co-training for DA (CODA) [7]. While CODA also utilizes co-training for SSDA using single-view data, it differs from DECOTA fundamentally as follow:

1. CODA takes a feature-centric view in that the two *artificial* views in its co-training procedure are constructed by decomposing the feature dimensions into two mutually exclusive subsets. DECOTA on the other hand achieves effective co-training with a two-task decomposition.
2. The two views in CODA do not exchange high confident pseudo-labels in a mini-batch fashion like DECOTA. Nor does CODA utilize MIXUP, which we have shown to be valuable for SSDA. Instead, CODA explicitly conducts feature alignment by minimizing the difference between the distributions of the source and target domains.
3. CODA trains a logistic regression classifier. In the era of deep learning, while co-training has been used in multiple vision tasks, DECOTA is *the first work in SSDA* utilizing deep learning, co-training, and mixup in a cohesive and principled fashion, achieving state of the art performance.

Since CODA is not deep learning based, to further justify the efficacy of DECOTA, we took the deep co-training work described in [44] that was designed for semi-supervised image recognition, and customize it for SSDA. [44] constructs multi-views for co-training via two different adversarial perturbations on the same image samples, after which the two networks are trained to make different mistakes on the same adversarial examples. For fair comparison, we compare [44] both with and without MIXUP, using the DomainNet [41] dataset. The results are given in Table 1. DECOTA outperforms [44] by a margin. See Section 4 for detailed setups.

4. Experiments

We consider the one-/three-shot settings, following [51], where each class is given one or three labeled target examples. We train with $D_{\mathcal{S}}$, $D_{\mathcal{T}}$, and unlabeled $D_{\mathcal{U}}$. We then reveal the true label of $D_{\mathcal{U}}$ for evaluation.

Datasets. We use **DomainNet** [41], a large-scale benchmark dataset for domain adaptation that has 345 classes and 6 domains. We follow [51], using a 126-class subset with 4 domains (*i.e.*, R: Real, C: Clipart, P: Painting, S: Sketch.) and report 7 different adaptation scenarios. We also use **Office-Home** [66], another benchmark that contains 65 classes, with 12 adaptation scenarios constructed from 4 domains (*i.e.*, R: Real world, C: Clipar t, A: Art, P: Product).

Implementation details. We implement using Pytorch [39]. We follow [51] to use ResNet-34 [18] on DomainNet and VGG-16 [58] on Office-Home. We also provide ResNet-34 results on Office-Home in order to fairly compare with [26] in supplementary. The networks are pre-trained on ImageNet [11, 49]. We follow [51, 46] to replace the last linear layer with a K-way cosine classifier (*e.g.*, $K = 126$ for DomainNet) and train it at a fixed temperature (0.05 in all our experiments). We initialize $\boldsymbol{w}_f$ with a model first fine-tuned on $D_{\mathcal{S}}$, and initialize $\boldsymbol{w}_g$ with a model first fine-tuned on $D_{\mathcal{S}}$ and then fine-tuned on $D_{\mathcal{T}}$. We do so to encourage the two models to be different at the beginning. At each iteration, we sample three mini-batches $S \subset D_{\mathcal{S}}$, $T \subset D_{\mathcal{T}}$, and $U \subset D_{\mathcal{U}}$ of equal sizes $B = 24$ (cf. Section 3.1.1). We set the confidence threshold $\tau = 0.5$, and beta distribution coefficient $\alpha = 1.0$. We use SGD with momentum of 0.9 and an initial learning rate of 0.001, following [51]. We train for 50K/10K iterations on DomainNet/Office-Home. We note that, DECOTA does not increase the training time since at each iteration, it only updates and learns from the pseudo-labels of the current mini-batch of unlabeled data, not the entire unlabeled data.

Baselines. We compare to four state-of-the-art SSDA approaches, **MME** [51], **UODA** [45], **APE** [26], and **ELP** [22]. We also compare to **S+T**, a model trained with $D_{\mathcal{S}}$ and $D_{\mathcal{T}}$, without using $D_{\mathcal{U}}$. Additionally, we compare to **DANN** [13] (domain adversarial learning) and **ENT** [16] (entropy minimization), both of which are important prior work on UDA. We modify them such that $D_{\mathcal{S}}$ and $D_{\mathcal{T}}$ are used jointly to train the classifier, following [51]. We denote by **S** the model trained only with the source data $D_{\mathcal{S}}$.

Variants of our approach. We consider variants of our approach for extensive ablation studies. We first introduce a model we called MIXUP Self-Training (MIST). MIST is trained as follows

$$\boldsymbol{w} \leftarrow \boldsymbol{w} - \eta \nabla \mathcal{L}(\boldsymbol{w}, S) + \nabla \mathcal{L}(\boldsymbol{w}, T) + \nabla \mathcal{L}(\boldsymbol{w}, \tilde{U}_S^{(\boldsymbol{w})}) + \nabla \mathcal{L}(\boldsymbol{w}, \tilde{U}_T^{(\boldsymbol{w})})), \tag{7}$$

where $\tilde{U}_S^{(\boldsymbol{w})}$ and $\tilde{U}_T^{(\boldsymbol{w})}$ are pseudo-labels obtained from $\boldsymbol{w}$, followed by MIXUP with S and T, respectively. MIST basically lumps all the pseudo and hard labeled samples together during training, and is intended for comparing with the effect of co-training. **S+T+pseudo-U** is the model trained

Table 2: Accuracy on DomainNet (%) for three-shot setting with 4 domains, using ResNet-34.

Method	R to C	R to P	P to C	C to S	S to P	R to S	P to R	Mean
S+T	60.8	63.6	60.8	55.6	59.5	53.3	74.5	61.2
DANN [13]	62.3	63.0	59.1	55.1	59.7	57.4	67.0	60.5
ENT [51]	67.8	67.4	62.9	50.5	61.2	58.3	79.3	63.9
MME [51]	72.1	69.2	69.7	59.0	64.7	62.2	79.0	68.0
UODA [45]	75.4	71.5	73.2	64.1	69.4	64.2	80.8	71.2
APE [26]	76.6	72.1	76.7	63.1	66.1	67.8	79.4	71.7
ELP [22]	74.9	72.1	74.4	64.3	69.7	64.9	81.0	71.6
DECOTA	**80.4**	**75.2**	**78.7**	**68.6**	**72.7**	**71.9**	**81.5**	**75.6**

Table 3: Accuracy on Office-Home (%) for three-shot setting with 4 domains, using VGG-16.

Method	R to C	R to P	R to A	P to R	P to C	P to A	A to P	A to C	A to R	C to R	C to A	C to P	Mean
S+T	49.6	78.6	63.6	72.7	47.2	55.9	69.4	47.5	73.4	69.7	56.2	70.4	62.9
DANN [13]	56.1	77.9	63.7	73.6	52.4	56.3	69.5	50.0	72.3	68.7	56.4	69.8	63.9
ENT [51]	48.3	81.6	65.5	76.6	46.8	56.9	73.0	44.8	75.3	72.9	59.1	77.0	64.8
MME [51]	56.9	82.9	65.7	76.7	53.6	59.2	75.7	54.9	75.3	72.9	61.1	76.3	67.6
UODA [45]	57.6	83.6	67.5	**77.7**	54.9	**61.0**	77.7	**55.4**	**76.7**	73.8	61.9	**78.4**	68.9
APE [26]	56.0	81.0	65.2	73.7	51.4	59.3	75.0	54.4	73.7	71.4	61.7	75.1	66.5
ELP [22]	57.1	83.2	67.0	76.3	53.9	59.3	75.9	55.1	76.3	73.3	61.9	76.1	68.0
DECOTA	**59.9**	**83.9**	**67.7**	77.3	**57.7**	60.7	**78.0**	54.9	76.0	**74.3**	**63.2**	**78.4**	**69.3**

with self-training, but without MIXUP. **Two-view MIST** is the direct ensemble of independently trained models, one for each view, using MIST (cf. Section 3.2). **Vanilla-Ensemble** is the ensemble model by combining two MIST trained on $D_{\mathcal{S}}$, $D_{\mathcal{T}}$, and $D_{\mathcal{U}}$ but with different initialization. For all the variants that train only one model, we initialize it with a pre-trained model fine-tuned on $D_{\mathcal{S}}$ and then fine-tuned on $D_{\mathcal{T}}$. Otherwise, we initialize the two models in the same way as DECOTA. *We note that, for any methods that involve two models, we perform ensemble on their output probability.*

Main results. We summarize the comparison with baselines in Table 2 and Table 3. We mainly report the three-shot results and leave the one-shot results in the supplementary material. DECOTA outperforms other methods by a large margin on DomainNet, and outperforms all methods on Office-Home (mean). The smaller gain on Office-Home may be due to its smaller data size and limited scenes. DomainNet is larger and more diverse; the significant improvement on it is a stronger indicator of the effectiveness of our algorithm.

We further provide detailed analysis on DECOTA. We mainly report the DomainNet three-shot results. Other detailed results can be found in the supplementary material.

Task decomposition. We first compare DECOTA to MIST. As shown in Table 4 (a)-(b), DECOTA outperforms MIST by 1% on DomainNet and 5% on Office-Home on the three-shot setup. Fig. 3 (c) further shows the number of pseudo-labels involved in model training (those with confidence larger than $\tau = 0.5$). We see that DECOTA always generates more pseudo-label data with a higher accuracy than MIST (also in Fig. 3 (b)), justifying our claim that the decomposition helps keep $D_{\mathcal{S}}$'s and $D_{\mathcal{T}}$'s specialties, producing high confident predictions on more unlabeled data as a result.

Co-training. We compare DECOTA to **two-view MIST**. Both methods decompose the data into a SSL and a UDA task. The difference is in how the pseudo-label set was generated (cf. Eq. (1)): **Two-view MIST** constructs each set independently (cf. Section 3.2). DECOTA outperforms two-view MIST by a margin, not only on ensemble, but also on each view alone, justifying the effectiveness of two models exchanging their specialties to benefit each other. As in Table 4 (c), each model of DECOTA outperforms MIST.

MIXUP. We examine the importance of MIXUP. Specifically, we compare MIST and **S+T+pseudo-U**. The second model trains in the same way as MIST, except that it does not apply MIXUP. On DomainNet (3-shot), MIST outperforms **S+T+pseudo-U** by **9%** on average. We attribute this difference to the *denoising* effect by MIXUP: MIXUP is performed after the pseudo-label set is defined, so it does not directly affect the number of pseudo-labels, but the quality. We further calculate the number of correctly assigned pseudo-labels along training, as shown in Fig. 3 (d). With MIXUP, the correct pseudo-label pool boosts consistently. In contrast, **S+T+pseudo-U** reinforces itself with wrongly assigned pseudo-labels; the percentage thus remains constantly low. Comparison results are shown in Table 4 (d).

Comparison to vanilla model ensemble. Since DECOTA combines $\boldsymbol{w}_f$ and $\boldsymbol{w}_g$ in making predictions, for a fair comparison we train two MIST models (both use $D_{\mathcal{S}} + D_{\mathcal{T}} + D_{\mathcal{U}}$), each with different initialization, and perform model ensemble. As shown in Table 4 (a)-(b), DECOTA outperforms this vanilla model ensemble, especially on Office-Home, suggesting that our improvement does not simply

Table 4: Ablation Study (three shots). (a)-(b): comparison of MIST and DECOTA and the vanilla ensemble of two independently trained MIST; (c): comparison of Two-view MIST (without co-training) and DECOTA; (d) comparison of MIST and S+T+pseudo-U without MIXUP; (e) each model of DECOTA on the source domain test data, comparing to supervised training on source (S), average of DomainNet. All accuracy in (%).

(a) Comparing MIST, Vanilla-Ensemble of two MIST (with different initialization), and DECOTA on DomainNet

Method	R to C	R to P	P to C	C to S	S to P	R to S	P to R	Mean
MIST	78.1	75.2	76.7	68.3	72.6	71.5	79.8	74.6
Vanilla-Ensemble	79.7	75.0	77.2	68.4	72.1	70.8	79.7	74.7
DECOTA	80.4	75.2	78.7	68.6	72.7	71.9	81.5	75.6

(b) Comparing MIST, Vanilla-Ensemble of two MIST (with different initialization), and DECOTA on Office-Home

Method	R to C	R to P	R to A	P to R	P to C	P to A	A to P	A to C	A to R	C to R	C to A	C to P	Mean
MIST	54.7	81.2	64.0	69.4	51.7	58.8	69.1	47.6	70.6	65.3	60.8	73.8	63.9
Vanilla-Ensemble	56.1	81.8	63.4	72.9	54.1	55.1	74.2	49.5	72.1	67.4	55.2	75.6	64.7
DECOTA	59.9	83.9	67.7	77.3	57.7	60.7	78.0	54.9	76.0	74.3	63.2	78.4	69.3

(c) Comparing the decomposed tasks trained independently to using DECOTA

Method	Task	R to C	R to P	P to C	C to S	S to P	R to S	P to R	Mean
Decomposed tasks (without co-training)	$\boldsymbol{w}_f$	72.1	65.7	71.8	61.0	63.0	59.9	75.9	67.0
	$\boldsymbol{w}_g$	76.3	72.2	70.3	63.7	69.4	66.9	76.1	70.7
	Ensemble	77.3	72.0	75.1	65.7	69.3	66.1	78.7	72.0
DECOTA	$\boldsymbol{w}_f$	80.1	74.6	78.6	68.4	72.5	71.2	81.1	75.2
	$\boldsymbol{w}_g$	80.0	74.5	78.4	68.3	72.2	71.3	80.6	75.0
	Ensemble	80.4	75.2	78.7	68.6	72.7	71.9	81.5	75.6

(d) Comparing MIST and the S+T+pseudo-U with no MIXUP on DomainNet

Method	R to C	R to P	P to C	C to S	S to P	R to S	P to R	Mean
S+T+pseudo-U	70.0	67.2	68.3	57.2	61.1	58.7	71.2	65.6
MIST	78.1	75.2	76.7	68.3	72.6	71.5	79.8	74.6

(e) Accuracy on source domain

$\boldsymbol{w}_f$	$\boldsymbol{w}_g$	**DECOTA**	**S**
65.3	98.2	93.5	98.8

come from model ensemble, but from co-training.

On the "two-classifier-convergence" problem [75]. DECOTA is based on co-training and thus does not suffer the problem. This is shown in Table 4 (a, b): MIST and Vanilla-Ensemble are based on self-training and DECOTA outperformed them. Even at the end of training when two classifiers have similar accuracy (see Table 4 (c)), combining them still boosts the accuracy: *i.e.*, they make different predictions.

Results on the source domain. While $\boldsymbol{w}_f$ and $\boldsymbol{w}_g$ have similar accuracy on $D_{\mathcal{U}}$, the fact that $\boldsymbol{w}_f$ does not learn from $D_{\mathcal{S}}$ suggest their difference in classifying source domain data. We verify this in Table 4 (e), where we apply each model individually on a hold-out set from the source domain (provided by DomainNet). We see that $\boldsymbol{w}_g$ clearly dominates $\boldsymbol{w}_f$. Its accuracy is even on a par with a model trained only on $D_{\mathcal{S}}$, showing one advantage of DECOTA— the model can keep its discriminative ability on the source domain.

5. Conclusion

We introduce **DECOTA**, a simple yet effective approach for semi-supervised domain adaptation (SSDA). Our key contribution is the novel insight that the two sources of supervisions (*i.e.*, the labeled target and labeled source data) are inherent different and should not be combined directly. DECOTA thus explicitly decomposes SSDA into two tasks (*i.e.*, views), a semi-supervised learning task and an unsupervised domain adaptation task, in which each supervision can be better leveraged. To encourage knowledge sharing and integration between the two tasks, we employ co-training, a well-established technique that allows for distinct views to learn from each other. We provided empirical evidence that the two tasks satisfy the theoretical condition of co-training, which makes DECOTA well founded, simple (without adversarial learning), and superior in performance.

Acknowledgement. This research was supported by independent awards from Facebook AI, NSF (III-1618134, III-1526012, IIS-1149882, IIS-1724282, TRIPODS-1740822, OAC-1934714, DMR-1719875), ONR DOD (N00014-17-1-2175), DARPA SAIL-ON (W911NF2020009), and the Bill and Melinda Gates Foundation. We are thankful for generous support by Ohio Supercomputer Center and AWS Cloud Credits for Research.

References

[1] Shuang Ao, Xiang Li, and Charles X Ling. Fast generalized distillation for semi-supervised domain adaptation. In *AAAI*, 2017. 1, 3

[2] Maria-Florina Balcan, Avrim Blum, and Ke Yang. Co-training and expansion: Towards bridging theory and practice. In *NIPS*, 2005. 2, 3, 5

[3] Michele Banko and Eric Brill. Scaling to very very large corpora for natural language disambiguation. In *Proceedings of the 39th annual meeting on association for computational linguistics*, pages 26–33. Association for Computational Linguistics, 2001. 2, 5

[4] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine learning*, 79(1-2):151–175, 2010. 1

[5] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. In *NeurIPS*, 2019. 2, 4

[6] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *COLT*, 1998. 2, 3

[7] Minmin Chen, Kilian Q Weinberger, and John Blitzer. Co-training for domain adaptation. In *NIPS*, 2011. 2, 3, 5, 6

[8] Minmin Chen, Kilian Q Weinberger, and Yixin Chen. Automatic feature decomposition for single view co-training. In *ICML*, 2011. 2, 3, 5

[9] Yuhua Chen, Wen Li, Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Domain adaptive faster r-cnn for object detection in the wild. In *CVPR*, 2018. 1

[10] Yi-Hsin Chen, Wei-Yu Chen, Yu-Ting Chen, Bo-Cheng Tsai, Yu-Chiang Frank Wang, and Min Sun. No more discrimination: Cross city adaptation of road scene segmenters. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1992–2001, 2017. 1

[11] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 6

[12] Jeff Donahue, Judy Hoffman, Erik Rodner, Kate Saenko, and Trevor Darrell. Semi-supervised domain adaptation with instance constraints. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 668–675, 2013. 1

[13] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *JMLR*, 17(1):2096–2030, 2016. 1, 2, 4, 6, 7, 13, 14, 16

[14] Boqing Gong, Kristen Grauman, and Fei Sha. Learning kernels for unsupervised domain adaptation with applications to visual object recognition. *JMLR*, 109(1-2):3–27, 2014. 1

[15] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *ICLR*, 2015. 3

[16] Yves Grandvalet and Yoshua Bengio. Semi-supervised learning by entropy minimization. In *NIPS*, 2005. 2, 6

[17] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *NeurIPS*, 2018. 2, 3

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 5, 6

[19] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 3

[20] Judy Hoffman, Erik Rodner, Jeff Donahue, Trevor Darrell, and Kate Saenko. Efficient learning of domain-invariant image representations. In *ICLR*, 2013. 3

[21] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei A Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *ICML*, 2018. 1, 2

[22] Zhiyong Huang, Kekai Sheng, Weiming Dong, Xing Mei, Chongyang Ma, Feiyue Huang, Dengwen Zhou, and Changsheng Xu. Effective label propagation for discriminative semi-supervised domain adaptation. *arXiv preprint arXiv:2012.02621*, 2020. 6, 7, 13

[23] Naoto Inoue, Ryosuke Furuta, Toshihiko Yamasaki, and Kiyoharu Aizawa. Cross-domain weakly-supervised object detection through progressive domain adaptation. In *CVPR*, 2018. 2

[24] Mehran Khodabandeh, Arash Vahdat, Mani Ranjbar, and William G Macready. A robust learning approach to domain adaptive object detection. In *ICCV*, 2019. 2

[25] Seunghyeon Kim, Jaehoon Choi, Taekyung Kim, and Changick Kim. Self-training and adversarial background regularization for unsupervised domain adaptive one-stage object detection. In *ICCV*, 2019. 2

[26] Taekyung Kim and Changick Kim. Attract, perturb, and explore: Learning a feature alignment network for semi-supervised domain adaptation. In *ECCV*, 2020. 1, 2, 3, 6, 7, 13, 14

[27] Ananya Kumar, Tengyu Ma, and Percy Liang. Understanding self-training for gradual domain adaptation. In *International Conference on Machine Learning*, pages 5468–5479. PMLR, 2020. 2

[28] Dong-Hyun Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, 2013. 2, 5

[29] Seungmin Lee, Dongwan Kim, Namil Kim, and Seong-Gyun Jeong. Drop to adapt: Learning discriminative features for unsupervised domain adaptation. In *ICCV*, 2019. 2

[30] Da Li and Timothy Hospedales. Online meta-learning for multi-source and semi-supervised domain adaptation. *arXiv preprint arXiv:2004.04398*, 2020. 1, 3

[31] Limin Li and Zhenyue Zhang. Semi-supervised domain adaptation by covariance matching. *TPAMI*, 41(11):2724–2739, 2018. 1, 3

[32] Jian Liang, Ran He, Zhenan Sun, and Tieniu Tan. Distant supervised centroid shift: A simple and efficient approach to visual domain adaptation. In *CVPR*, 2019. 2

[33] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael I Jordan. Learning transferable features with deep adaptation networks. *arXiv preprint arXiv:1502.02791*, 2015. 2

[34] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008. 4

[35] Zhijun Mai, Guosheng Hu, Dexiong Chen, Fumin Shen, and Heng Tao Shen. Metamixup: Learning adaptive interpolation policy of mixup with meta-learning. *arXiv preprint arXiv:1908.10059*, 2019. 4

[36] Xudong Mao, Yun Ma, Zhenguo Yang, Yangbin Chen, and Qing Li. Virtual mixup training for unsupervised domain adaptation. *arXiv preprint arXiv:1905.04215*, 2019. 2, 4

[37] David McClosky, Eugene Charniak, and Mark Johnson. Effective self-training for parsing. In *ACL*, 2006. 2, 5

[38] David McClosky, Eugene Charniak, and Mark Johnson. Reranking and self-training for parser adaptation. In *ACL*, 2006. 2, 5

[39] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017. 6

[40] Zhongyi Pei, Zhangjie Cao, Mingsheng Long, and Jianmin Wang. Multi-adversarial domain adaptation. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018. 2

[41] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1406–1415, 2019. 1, 2, 5, 6

[42] Luis AM Pereira and Ricardo da Silva Torres. Semi-supervised transfer subspace for domain adaptation. *Pattern Recognition*, 75:235–249, 2018. 3

[43] Viraj Prabhu, Arjun Chandrasekaran, Kate Saenko, and Judy Hoffman. Active domain adaptation via clustering uncertainty-weighted embeddings. *arXiv preprint arXiv:2010.08666*, 2020. 3

[44] Siyuan Qiao, W. Shen, Zhishuai Zhang, Bo Wang, and A. Yuille. Deep co-training for semi-supervised image recognition. In *ECCV*, 2018. 2, 3, 6

[45] Can Qin, Lichen Wang, Qianqian Ma, Yu Yin, Huan Wang, and Yun Fu. Opposite structure learning for semi-supervised domain adaptation. *arXiv preprint arXiv:2002.02545*, 2020. 1, 2, 3, 6, 7, 12, 13

[46] Rajeev Ranjan, Carlos D Castillo, and Rama Chellappa. L2-constrained softmax loss for discriminative face verification. *arXiv preprint arXiv:1703.09507*, 2017. 6

[47] Stephan R Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. In *ECCV*, 2016. 1

[48] German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio M Lopez. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *CVPR*, 2016. 1

[49] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 115(3):211–252, 2015. 6

[50] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *ECCV*, 2010. 13

[51] Kuniaki Saito, Donghyun Kim, Stan Sclaroff, Trevor Darrell, and Kate Saenko. Semi-supervised domain adaptation via minimax entropy. In *ICCV*, 2019. 1, 2, 3, 6, 7, 12, 13, 14, 16

[52] Kuniaki Saito, Yoshitaka Ushiku, Tatsuya Harada, and Kate Saenko. Adversarial dropout regularization. *arXiv preprint arXiv:1711.01575*, 2017. 2

[53] Kuniaki Saito, Kohei Watanabe, Yoshitaka Ushiku, and Tatsuya Harada. Maximum classifier discrepancy for unsupervised domain adaptation. In *CVPR*, 2018. 1, 2

[54] Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Semantic foggy scene understanding with synthetic data. *International Journal of Computer Vision*, 126(9):973–992, 2018. 1

[55] Fatemeh Sadat Saleh, Mohammad Sadegh Aliakbarian, Mathieu Salzmann, Lars Petersson, and Jose M Alvarez. Effective use of synthetic data for urban scene semantic segmentation. In *ECCV*. Springer, 2018. 1

[56] Swami Sankaranarayanan, Yogesh Balaji, Arpit Jain, Ser Nam Lim, and Rama Chellappa. Learning from synthetic data: Addressing domain shift for semantic segmentation. In *CVPR*, 2018. 1

[57] Rui Shu, Hung H Bui, Hirokazu Narui, and Stefano Ermon. A dirt-t approach to unsupervised domain adaptation. In *ICLR*, 2018. 2

[58] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 6

[59] Kihyuk Sohn, David Berthelot, Chun-Liang Li, Zizhao Zhang, Nicholas Carlini, Ekin D Cubuk, Alex Kurakin, Han Zhang, and Colin Raffel. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In *NeurIPS*, 2020. 13, 16

[60] Jong-Chyi Su, Yi-Hsuan Tsai, Kihyuk Sohn, Buyu Liu, Subhransu Maji, and Manmohan Chandraker. Active adversarial domain adaptation. In *WACV*, 2020. 3

[61] Yuhua Tang, Zhipeng Lin, Haotian Wang, and Liyang Xu. Adversarial mixup synthesis training for unsupervised domain adaptation. In *ICASSP*, 2020. 2, 4

[62] Qingyi Tao, Hao Yang, and Jianfei Cai. Zero-annotation object detection with web knowledge transfer. In *ECCV*, 2018. 2

[63] Yi-Hsuan Tsai, Wei-Chih Hung, Samuel Schulter, Kihyuk Sohn, Ming-Hsuan Yang, and Manmohan Chandraker. Learning to adapt structured output space for semantic segmentation. In *CVPR*, 2018. 1

[64] Eric Tzeng, Judy Hoffman, Trevor Darrell, and Kate Saenko. Simultaneous deep transfer across domains and tasks. In *ICCV*, 2015. 3

[65] Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474*, 2014. 2

[66] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *CVPR*, 2017. 2, 6

[67] Riccardo Volpi, Pietro Morerio, Silvio Savarese, and Vittorio Murino. Adversarial feature augmentation for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5495–5504, 2018. 2

[68] Ximei Wang, Liang Li, Weirui Ye, Mingsheng Long, and Jianmin Wang. Transferable attention for domain adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5345–5352, 2019. 2

[69] Yan Wang, Chen Xiangyu, You Yurong, Erran Li Li, Bharath Hariharan, Mark Campbell, Kilian Q. Weinberger, and Chao Wei-Lun. Train in germany, test in the usa: Making 3d object detectors generalize. In *CVPR*, 2020. 1, 2

[70] Colin Wei, Kendrick Shen, Yining Chen, and Tengyu Ma. Theoretical analysis of self-training with deep networks on unlabeled data. *arXiv preprint arXiv:2010.03622*, 2020. 2

[71] Minghao Xu, Jian Zhang, Bingbing Ni, Teng Li, Chengjie Wang, Qi Tian, and Wenjun Zhang. Adversarial domain adaptation with domain mixup. In *AAAI*, 2020. 2, 4

[72] Shen Yan, Huan Song, Nanxiang Li, Lincan Zou, and Liu Ren. Improve unsupervised domain adaptation with mixup training. *arXiv preprint arXiv:2001.00677*, 2020. 2, 4

[73] Luyu Yang, Yogesh Balaji, Ser-Nam Lim, and Abhinav Shrivastava. Curriculum manager for source selection in multi-source domain adaptation. *arXiv preprint arXiv:2007.01261*, 2020. 2

[74] Ting Yao, Yingwei Pan, Chong-Wah Ngo, Houqiang Li, and Tao Mei. Semi-supervised domain adaptation with subspace learning for visual recognition. In *CVPR*, 2015. 1, 3

[75] Xingrui Yu, Bo Han, Jiangchao Yao, Gang Niu, Ivor Tsang, and Masashi Sugiyama. How does disagreement help generalization against label corruption? In *International Conference on Machine Learning*, pages 7164–7173. PMLR, 2019. 8

[76] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *ICLR*, 2018. 2, 4, 16

[77] Xiaojin Jerry Zhu. Semi-supervised learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 2005. 15

[78] Yang Zou, Zhiding Yu, BVK Vijaya Kumar, and Jinsong Wang. Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In *ECCV*, 2018. 2