Abstract

Unsupervised Domain Adaptation (UDA) makes predictions for the target domain data while manual annotations are only available in the source domain. Previous methods minimize the domain discrepancy neglecting the class information, which may lead to misalignment and poor generalization performance. To address this issue, this paper proposes Contrastive Adaptation Network (CAN) optimizing a new metric which explicitly models the intra-class domain discrepancy and the inter-class domain discrepancy. We design an alternating update strategy for training CAN in an end-to-end manner. Experiments on two real-world benchmarks Office-31 and VisDA-2017 demonstrate that CAN performs favorably against the state-of-the-art methods and produces more discriminative features.

1. Introduction

Recent advancements in deep neural networks have successfully improved a variety of learning problems [40, 8, 26, 19, 20]. For supervised learning, however, massive labeled training data is still the key to learning an accurate deep model. Although abundant labels may be available for a few pre-specified domains, such as ImageNet [7], manual labels often turn out to be difficult or expensive to obtain for every ad-hoc target domain or task. The absence of in-domain labeled data hinders the application of data-fitting models in many real-world problems.

In the absence of labeled data from the target domain, Unsupervised Domain Adaptation (UDA) methods have emerged to mitigate the domain shift in data distributions [2, 1, 5, 37, 30, 18, 3, 17]. It relates to unsupervised learning as it requires manual labels only from the source domain and zero labels from the target domain. Among the recent work on UDA, a seminal line of work proposed by Long et al. [22, 25] aims at minimizing the discrepancy between the source and target domain in the deep neural network, where the domain discrepancy is measured by Maximum Mean Discrepancy (MMD) [22] and Joint MMD (JMMD) [25]. MMD and JMMD have proven effective in many computer vision problems and demonstrated the state-of-the-art results on several UDA benchmarks [22, 25].

Despite the success of previous methods based on MMD and JMMD, most of them measure the domain discrepancy at the domain level, neglecting the class label of the sample, and hence may lead to sub-optimal solutions. Consequently, the target samples of one label may be misaligned with source samples of a different label. Right: Our method performs class-aware alignment across domains. To avoid the misalignment, only the intra-class domain discrepancy is minimized. The inter-class domain discrepancy is maximized to enhance the model’s generalization ability.
To address the above issues, we introduce a new Contrastive Domain Discrepancy (CDD) objective to enable class-aware UDA. We propose to minimize the intra-class discrepancy, i.e. the domain discrepancy within the same class, and maximize the inter-class margin, i.e. the domain discrepancy between different classes. Considering the toy example in Fig. 1, CDD will draw closer the source and target samples of the same underlying class (e.g. the blue and red triangles), while pushing apart the samples from different classes (e.g. the blue triangle and the red star).

Unfortunately, to estimate and optimize with CDD, we may not train a deep network out-of-the-box as we need to overcome the following two technical issues. First, we need labels from both domains to compute CDD, however, target labels are unknown in UDA. A straightforward way, of course, is to estimate the target labels by the network outputs during training. However, because the estimation can be noisy, we find it can harm the adaptation performance (see Section 4.3). Second, during the mini-batch training, for a class $C$, the mini-batch may only contain samples from one domain (source or target), rendering it infeasible to estimate the intra-class domain discrepancy of $C$. This can result in a less efficient adaptation. The above issues require special design of the network and the training paradigm.

In this paper, we propose Contrastive Adaptation Network (CAN) to facilitate the optimization with CDD. During training, in addition to minimizing the cross-entropy loss on labeled source data, CAN alternatively estimates the underlying label hypothesis of target samples through clustering, and adapts the feature representations according to the CDD metric. After clustering, the ambiguous target data (i.e. far from the cluster centers) and ambiguous classes (i.e. containing few target samples around the cluster centers) are zeroed out in estimating the CDD. Empirically we find that during training, an increasing amount of samples will be taken into account. Such progressive learning can help CAN capture more accurate statistics of data distributions. Moreover, to facilitate the mini-batch training of CAN, we employ the class-aware sampling for both source and target domains, i.e. at each iteration, we sample data from both domains for each class within a randomly sampled class subset. Class-aware sampling can improve the training efficiency and the adaptation performance.

We validate our method on two public UDA benchmarks: Office-31 [30] and VisDA-2017 [29]. The experimental results show that our method performs favorably against the state-of-the-art UDA approaches, i.e. we achieve the best-published result on the Office-31 benchmark and very competitive result on the challenging VisDA-2017 benchmark. Ablation studies are presented to verify the contribution of each key component in our framework.

In a nutshell, our contributions are as follows,

- We introduce a new discrepancy metric Contrastive Domain Discrepancy (CDD) to perform class-aware alignment for unsupervised domain adaptation.
- We propose a network Contrastive Adaptation Network to facilitate the end-to-end training with CDD.
- Our method achieves the best-published result on the Office-31 benchmark [30] and competitive performance compared to the state-of-the-art on the challenging VisDA-2017 benchmark [29].

2. Related Work

Class-agnostic domain alignment. A common practice for UDA is to minimize the discrepancy between domains to obtain domain-invariant features [10, 4, 25, 22, 24, 36, 21]. For example, Tzeng et al. [38] proposed a kind of domain confusion loss to encourage the network to learn both semantically meaningful and domain invariant representations. Long et al. proposed DAN [22] and JAN [25] to minimize the MMD and Joint MMD distance across domains respectively, over the domain-specific layers. Ganin et al. [10] enabled the network to learn domain invariant representations in adversarial way by back-propagating the reverse gradients of the domain classifier. Unlike these domain-discrepancy minimization methods, our method performs class-aware domain alignment.

Discriminative domain-invariant feature learning. Some previous works pay efforts to learn more discriminative features while performing domain alignment [35, 13, 31, 32, 28, 39]. Adversarial Dropout Regularization (ADR) [31] and Maximum Classifier Discrepancy (MCD) [32] were proposed to train a deep neural network in adversarial way to avoid generating non-discriminative features lying in the region near the decision boundary. Similar to us, Long et al. [23] and Pei et al. [28] take the class information into account while measuring the domain discrepancy. However, our method differs from theirs mainly in two aspects. Firstly, we explicitly model two types of domain discrepancy, i.e. the intra-class domain discrepancy and the inter-class domain discrepancy. The inter-class domain discrepancy, which has been ignored by most previous methods, is proved to be beneficial for enhancing the model adaptation performance. Secondly, in the context of deep neural networks, we treat the training process as an alternative optimization over target label hypothesis and features.

Intra-class compactness and inter-class separability modeling. This paper is also related to the work that explicitly models the intra-class compactness and the inter-class separability, e.g. the contrastive loss [12] and the triplet loss [33]. These methods have been used in various applications, e.g. face recognition [6], person re-identification [16], etc. Different from these methods designed for a single domain, our work focuses on adaptation across domains.
3. Methodology

Unsupervised Domain Adaptation (UDA) aims at improving the model’s generalization performance on target domain by mitigating the domain shift in data distribution of the source and target domain. Formally, given a set of source domain samples $S = \{ (x^s_1, y^s_1), \cdots, (x^s_N, y^s_N) \}$, and target domain samples $T = \{ x^t_1, \cdots, x^t_M \}$, $x^s$, $x^t$ represent the input data, and $y^s \in \{0, 1, \cdots, M - 1\}$ denote the source data label of $M$ classes. The target data label $y^t \in \{0, 1, \cdots, M - 1\}$ is unknown. Thus, in UDA, we are interested in training a network using labeled source domain data $S$ and unlabeled target domain data $T$ to make accurate predictions $\{ \hat{y}^t \}$ on $T$.

We discuss our method in the context of deep neural networks. In deep neural networks, a sample owns hierarchical features/representations denoted by the activations of each layer $l \in \mathcal{L}$. In the following, we use $\phi_l(x)$ to denote the outputs of layer $l$ in a deep neural network $\phi_D$ for the input $x$, where $\phi_l(\cdot)$ denotes the mapping defined by the deep neural network from the input to a specific layer.

In the rest of this section, we start our discussions by briefly reviewing the relevant concepts in MMD in Section 3.1. Section 3.2 introduces a new domain discrepancy metric. Finally, Section 3.3 and Section 3.4 discuss the objective and the training procedure of proposed deep network.

3.1. Maximum Mean Discrepancy Revisit

In Maximum Mean Discrepancy (MMD), $\{x^s_l\}$ and $\{x^t_l\}$ are i.i.d. sampled from the marginal distributions $P(X^s)$ and $Q(X^t)$ respectively. Based on the observed samples, MMD [34] performs a kernel two-sample test to determine whether to accept the null hypothesis $P = Q$ or not. MMD is motivated by the fact that if two distributions are identical, all of their statistics should be the same. Formally, MMD defines the difference between two distributions with their mean embeddings in the reproducing kernel Hilbert space (RKHS), i.e.

$$D_{\mathcal{H}}(P, Q) \triangleq \sup_{f \sim \mathcal{H}} | \mathbb{E}_{X^s} [f(X^s)] - \mathbb{E}_{X^t} [f(X^t)] |_{\mathcal{H}},$$

where $\mathcal{H}$ is class of functions.

In practice, for a layer $l$, the squared value of MMD is estimated with the empirical kernel mean embeddings

$$D_{\mathcal{H}}^{\text{mmd}} = \frac{1}{n_s^2} \sum_{i=1}^{n_s} \sum_{j=1}^{n_s} k_l(\phi_l(x^s_i), \phi_l(x^s_j))$$

$$+ \frac{1}{n_t^2} \sum_{i=1}^{n_t} \sum_{j=1}^{n_t} k_l(\phi_l(x^t_i), \phi_l(x^t_j))$$

$$- \frac{2}{n_s n_t} \sum_{i=1}^{n_s} \sum_{j=1}^{n_t} k_l(\phi_l(x^s_i), \phi_l(x^t_j)),$$

where $x^s \in S \subset S$, $x^t \in T \subset T$, $n_s = |S|$, $n_t = |T|$. The $S'$ and $T'$ represent the mini-batch source and target data sampled from $S$ and $T$ respectively. And $k_l$ denotes the kernel selected for the $l$-th layer of deep neural network.

3.2. Contrastive Domain Discrepancy

We propose to explicitly take the class information into account and measure the intra-class and inter-class discrepancy across domains. The intra-class domain discrepancy is minimized to compact the feature representations of samples within a class, whereas the inter-class domain discrepancy is maximized to push the representations of each other further away from the decision boundary. The intra-class and inter-class discrepancies are jointly optimized to improve the adaptation performance.

The proposed Contrastive Domain Discrepancy (CDD) is established on the difference between conditional data distributions across domains. Without any constraint on the type (e.g. marginal or conditional) of data distributions, MMD is convenient to measure such difference between $P(\phi(X^s)|Y^s)$ and $Q(\phi(X^t)|Y^t)$, i.e. $D_{\mathcal{H}}(P, Q) \triangleq \sup_{f \sim \mathcal{H}} | \mathbb{E}_{X^s} [f(\phi(X^s)|Y^s)] - \mathbb{E}_{X^t} [f(\phi(X^t)|Y^t)] |_{\mathcal{H}}$.

Supposing $\mu_{cc}(y, y') = \left\{ \begin{array}{ll} 1 & \text{if } y = c, y' = c' \\ 0 & \text{otherwise.} \end{array} \right.$, for two classes $c_1, c_2$ (which can be same or different), the kernel mean embedding estimation for squared $D_{\mathcal{H}}(P, Q)$ is

$$\hat{D}_{\mathcal{H}}^{c_1c_2}(\hat{y}^t_1, \hat{y}^t_2, \cdots, \hat{y}^t_{n_t}, \phi) = e_1 + e_2 - 2e_3$$

where

$$e_1 = \sum_{i=1}^{n_s} \sum_{j=1}^{n_s} \frac{\mu_{cc_1c_2}(y^s_i, y^s_j) k(\phi_l(x^s_i), \phi_l(x^s_j))}{\sum_{i=1}^{n_s} \sum_{j=1}^{n_s} \mu_{cc_1c_2}(y^s_i, y^s_j)}$$

$$e_2 = \sum_{i=1}^{n_t} \sum_{j=1}^{n_t} \frac{\mu_{cc_1c_2}(\hat{y}^t_i, \hat{y}^t_j) k(\phi_l(x^t_i), \phi_l(x^t_j))}{\sum_{i=1}^{n_t} \sum_{j=1}^{n_t} \mu_{cc_1c_2}(\hat{y}^t_i, \hat{y}^t_j)}$$

$$e_3 = \sum_{i=1}^{n_s} \sum_{j=1}^{n_t} \frac{\mu_{cc_1c_2}(y^s_i, \hat{y}^t_j) k(\phi_l(x^s_i), \phi_l(x^t_j))}{\sum_{i=1}^{n_s} \sum_{j=1}^{n_t} \mu_{cc_1c_2}(y^s_i, \hat{y}^t_j)}.$$

Note that Eq. (3) defines two kinds of class-aware domain discrepancy, 1) when $c_1 = c_2 = c$, it measures intra-class domain discrepancy; 2) when $c_1 \neq c_2$, it becomes the inter-class domain discrepancy. To compute the mask $\mu_{cc_1c_2}(\hat{y}^t_1, \hat{y}^t_2)$ and $\mu_{cc_1c_2}(y^t_1, \hat{y}^t_2)$, we need to estimate target labels $\{ \hat{y}^t_s \}$, which will be discussed in Section 3.4.

Based on the above definitions, the CDD is calculated as

(The $\hat{y}^t_1, \hat{y}^t_2, \cdots, \hat{y}^t_{n_t}$ is abbreviated as $\hat{y}^t_{1:n_t}$)
\[
\hat{D}_{cdd} = \frac{1}{M} \sum_{c=1}^{M} \hat{D}_{c}^{\text{intra}}(\hat{y}_{1:n_i}, \phi) \\
- \frac{1}{M(M-1)} \sum_{c=1}^{M} \sum_{c' \neq c}^{M} \hat{D}_{c}^{\text{inter}}(\hat{y}_{1:n_i}, \phi), \tag{5}
\]

where the intra- and inter-class domain discrepancies will be optimized in the opposite direction.

Note although the estimation of the labels \(\{\hat{y}_i^t\}\) can be noisy, the CDD (which is established on MMD) in itself is robust to the noise to an extent. Because MMD is determined by the mean embeddings of distributions in the RKHS, the sufficient statistics is less likely to be severely affected by the label noise, especially when the amount of data is large. We will discuss and verify this in Section 4.3.

### 3.3. Contrastive Adaptation Network

Deep convolutional neural networks (CNNs) is able to learn more transferable features than shallow methods. However, the discrepancy still exists for domain-specific layers. Specifically, the convolutional layers extracting general features are more transferable, while the fully-connected (FC) layers which exhibit abstract and domain-specific features should be adapted [22, 25].

In this paper, we start from ImageNet [7] pretrained networks, e.g. ResNet [14, 15], and replace the last FC layer with task-specific ones. We follow the general practice that minimizes the domain discrepancy of last FC layers and fine-tunes the convolutional layers through back-propagation. Then our proposed CDD can be readily incorporated into the objective as an adaptation module over the activations of FC layers. We name our network Contrastive Adaptation Network (CAN).

**The overall objective.** In a deep CNN, we need to minimize CDD over multiple FC layers, i.e. minimizing

\[
\hat{D}_{cdd}^L = \sum_{l=1}^{L} \hat{D}_{cdd}^l. \tag{6}
\]

Besides, we train the network with labeled source data through minimizing the cross-entropy loss,

\[
\ell_{cse} = -\frac{1}{n_s} \sum_{i'=1}^{n_s} \log P_{\theta}(y_i^s|\phi_{1}(x_i^s)) \tag{7}
\]

where \(y_i^s \in \{0, 1, \cdots, M-1\}\) is the ground-truth label of sample \(x_i^s\). \(P_{\theta}(y|\phi)\) denotes the predicted probability of label \(y\) with the network parameterized by \(\theta\), given input \(x\).

Therefore, the overall objective can be formulated as

\[
\min _{\theta} \ell = \ell_{cse} + \beta \hat{D}_{cdd}^L \tag{8}
\]

where \(\beta\) is the weight of the discrepancy penalty term. Through minimizing \(\hat{D}_{cdd}^L\), the intra-class domain discrepancy is minimized and the inter-class domain discrepancy is maximized to perform class-aware domain alignment.

Note that we independently sample the labeled source data to minimize the cross-entropy loss \(\ell_{cse}\) and those to estimate the CDD \(\hat{D}_{cdd}^L\). In this way, we are able to design more efficient sampling strategy (see Section 3.4) to facilitate the mini-batch stochastic optimization with CDD, while not disturbing the conventional optimization with cross-entropy loss on labeled source data.

### 3.4. Optimizing CAN

The framework of CAN is illustrated in Fig. 2. In this section, we mainly focus on discussing how to minimize CDD loss in CAN.

**Alternative optimization (AO).** As shown in Eq. (5), we need to jointly optimize the target label hypothesis \(\hat{y}_{1:n_i}^t\) and the feature representations \(\phi_{1:L}\). We adopt alternative steps to perform such optimization. In detail, at each loop, given current feature representations, i.e. fixing \(\theta\), we update target labels through clustering. Then, based on the updated target labels \(\hat{y}_i^t\), we estimate and minimize CDD to adapt the features, i.e. update \(\theta\) through back-propagation.

We employ the input activations \(\phi_{1}(-)\) of the first task-specific layer to represent a sample. For example, in ResNet, each sample can be represented as the outputs of the global average pooling layer, which are also the inputs of the following task-specific layer. Then the spherical K-means is adopted to perform the clustering of target samples and attach corresponding labels. The number of clusters is the same as the number of underlying classes \(M\). For each class, the target cluster center \(\hat{O}^{c}\) is initialized as the source cluster center \(O^{c}\), i.e. \(O^{c} \leftarrow O^{c}\), where \(O^{c} = \sum_{n_i=1}^{N} y_{i}^{c} = \frac{1}{\|\phi_{1}(x_{i})\|} \phi_{1}(x_{i})\) and \(c = \{0, 1, \cdots, M - 1\}\). For the metric measuring the distance between points \(a\) and \(b\) in the feature space, we apply the cosine dissimilarity, i.e. \(\text{dist}(a, b) = \frac{1}{2} (1 - \frac{a \cdot b}{\|a\|\|b\|})\).

Then the clustering process is iteratively 1) attaching labels for each target sample: \(\hat{y}_{i}^t \leftarrow \text{argmin}_{c} \text{dist}(\phi_{1}(x_{i}), O^{c})\), and 2) updating the cluster centers: \(O^{c} \leftarrow \sum_{i=1}^{N} \text{1}_{y_{i}^{c} = c} \frac{\phi_{1}(x_{i})}{\|\phi_{1}(x_{i})\|}\) till convergence or reaching the maximum clustering steps.

After clustering, each target sample \(x_{i}^t\) is assigned a label \(\hat{y}_{i}^t\) same as its affiliated clusters. Moreover, ambiguous data, which is far from its affiliated cluster center, is discarded, i.e. we select a subset \(\mathcal{T} = \{(x^t, \hat{y}^t)\} | \text{dist}(\phi_{1}(x^t), O^{(\hat{y}^t)}) < D_0\} \), where \(D_0 \in [0, 1]\) is a constant.

Moreover, to give a more accurate estimation of the distribution statistics, we assume that the minimum number of samples in \(\mathcal{T}\) assigned to each class, should be guaran-
Algorithm 1: Optimization of CAN at loop $T_e$.

**Input:**
- source data: $S = \{(x_1^{s}, y_1^{s}), \ldots, (x_{N_s}, y_{N_s})\}$,
- target data: $T = \{x_1^{t}, \ldots, x_{N_t}^{t}\}$

**Procedure:**
1. Forward $S$ and compute the $M$ cluster centers $O^{sc}$;
2. Initialize $O^{tc} \leftarrow O^{sc}$;
3. Cluster target samples $T$ using spherical K-means;
4. Filter the ambiguous target samples and classes;
5. for $(k \leftarrow 1; k \leq K; k \leftarrow k + 1)$ do
6. Class-aware sampling based on $C_{T_e}$, $\tilde{T}$, and $S$;
7. Compute $D_{\text{cdd}}$ using Eq. (6);
8. Sample from $S$ and compute $\ell^{tc}$ using Eq. (7);
9. Back-propagate with the objective $\ell$ (Eq. (8));
10. Update network parameters $\theta$.
11. end
4. Experiments

4.1. Setups

Datasets: We validate our method on two public benchmarks, Office-31 [30] is a common dataset for real-world domain adaptation tasks. It consists of 4,110 images belonging to 31 classes. This dataset contains three distinct domains, i.e., images which are collected from the 1) Amazon website (Amazon domain), 2) web camera (Webcam domain), and 3) digital SLR camera (DSLR domain) under different settings, respectively. The dataset is imbalanced across domains, with 2,817 images in A domain, 795 images in W domain, and 498 images in D domain. VisDA-2017 [29] is a challenging testbed for UDA with the domain shift from synthetic data to real imagery. In this paper, we validate our method on its classification task. In total there are ~280k images from 12 categories. The images are split into three sets, i.e., a training set with 152,397 synthetic images, a validation set with 55,388 real-world images, and a test set with 72,372 real-world images. The gallery of two datasets is shown in Fig. 3

Baselines: We compare our method with class-agnostic discrepancy minimization methods: RevGrad [10, 11], DAN [22], and JAN [25]. Moreover, we compare our method with the ones which explicitly or implicitly take the class information or decision boundary into consideration to learn more discriminative features: MADA [28], MCD [32], and ADR [31]. The descriptions of these methods can be found in Section 2. We implement DAN and JAN using the released code 1. For a comparison under optimal parameter setting, we cite the performance of MADA, RevGrad, MCD and ADR reported in their corresponding papers [28, 31, 32, 10].

Implementation details: We use ResNet-50 and ResNet-101 [14, 15] pretrained on ImageNet [7] as our backbone networks. We replace the last FC layer with the task-specific FC layer, and fine-tune the model with labeled source domain data and unlabeled target domain data. All the network parameters are shared between the source domain and target domain data other than those of the batch normalization layers which are domain-specific. The hyper-parameters are selected following the same protocol as described in [22], i.e., we train a domain classifier and perform selection on a validation set (of labeled source samples and unlabeled target samples) by jointly evaluating the test errors of the source classifier and the domain classifier.

We use mini-batch stochastic gradient descent (SGD) with momentum of 0.9 to train the network. We follow the same learning rate schedule as described in [10, 22, 25], i.e., the learning rate $\eta_p$ is adjusted following $\eta_p = \frac{\eta_0}{(1+ap^2)}$, where $p$ linearly increases from 0 to 1. The $\eta_0$ is the initial learning rate, i.e., 0.001 for the convolutional layers and 0.01 for the task-specific FC layer. For Office-31, $a = 10$ and $b = 0.75$, while for VisDA-2017, $a = 10$ and $b = 2.25$. The $\beta$ selected is 0.3. The thresholds ($D_0$, $N_0$) are set to (0.05, 3) for Office-31 tasks $A\rightarrow W$ and $A\rightarrow D$. And we don’t filter target samples and classes for other tasks during training.

4.2. Comparison with the state-of-the-art

Table 1 shows the classification accuracy on six tasks of Office-31. All domain adaptation methods yield notable improvement over the ResNet model (first row) which is fine-tuned on labeled source data only. CAN outperforms other baseline methods across all tasks, achieving the state-of-the-art performance. On average, it boosts the accuracy of JAN by a absolute 6.3% and that of MADA by 5.4%

We visualize the distribution of learned features by t-SNE [27]. Fig. 4 illustrates a representative task $W \rightarrow A$. Compared to JAN, as expected, the target data representations learned by CAN demonstrate higher intra-class compactness and much larger inter-class margin. This suggests that our CDD produces more discriminative features for the target domain and substantiates our improvement in Table 1.

Table 2 lists the accuracy over 12 classes on VisDA-2017 with the validation set as the target domain. Our method outperforms the other baseline methods. The mean accuracy of our model (87.2%) outperforms the self-ensembling (SE) method [9] (84.3%) which wins the first place in the VisDA-2017 competition, by 2.9%. It is worth noting that SE mainly deals with UDA by ensemble and data augmentation, which is orthogonal to the topic of this paper and thus can be easily combined to boost the performance further.

Moreover, we also perform adaptation on the VisDA-2017 test set (as the target domain), and submit our predictions to official evaluation server. Our goal is to evaluate the effectiveness of our proposed technique based on a vanilla backbone (ResNet-101). We choose not to use ensemble or additional data augmentation which is commonly used to boost the performance in the competition. Anyhow, our single model achieves a very competitive accuracy of 87.4%, which is comparable to the method which ranks at

1https://github.com/thuml/Xlearn
and examines two key components of our method: “CAS” uses conventional class-agnostic sampling instead of leaving-one-component-out of our framework at a time. In this way, we achieve better accuracy than MADA (+5.4%), ADR (+12.4%), and MCD (+15.3%).

### 4.3. Ablation studies

**Effect of inter-class domain discrepancy.** We compare our method (“CAN”) with that trained using intra-class discrepancy only (“intra only”), to verify the merits of introducing inter-class domain discrepancy measure. The results are shown in the last two rows in Table 1 and 2. It can be seen that introducing the inter-class domain discrepancy improves the adaptation performance. We believe the reason is that it is impossible to completely eliminate the intra-class domain discrepancy, maximizing the inter-class domain discrepancy may alleviate the possibility of the model overfitting to the source data and benefits the adaptation.

**Effect of alternative optimization and class-aware sampling.** Table 3 examines two key components of CAN, *i.e.* alternative optimization (or “AO”), and class-aware sampling (or “CAS”). We perform ablation study by leaving-one-component-out of our framework at a time. In Table 3, the method “w/o. AO” directly employs the outputs of the network at each iteration as pseudo target labels to estimate CDD and back-propagates to update the network. It can be regarded as updating the feature representations and pseudo target labels simultaneously. The method “w/o. CAS” uses conventional class-agnostic sampling instead of CAS. The comparisons to these two special cases verify the contributions of AO and CAS in our method.

Interestingly, even without alternative optimization, the method “w/o. AO” improves over class-agnostic methods, *e.g.* DAN, JAN, etc. This suggests our proposed CDD in itself is robust to the label noise to some extent, and MMD is a suitable metric to establish CDD (see Section 3.2).
The “pseudo-labeled” data is due to the large capacity of deep neural networks.

The discrepancy makes the model better adapted and less likely to be affected by the label noise.

As shown in Table 4, “pseudo0” leads to a model whose accuracy exactly matches with that of the initial clustering, due to the large capacity of deep neural networks. The “pseudo1” achieves significantly better results than “pseudo0”, but is still worse than our CAN, which verifies that our way of explicitly modeling the class-aware domain discrepancy makes the model better adapted and less likely to be affected by the label noise.

**CDD value during training.** In our training, we generate target label hypothesis to estimate CDD. We expect that the underlying metric computed with the ground-truth target labels would decrease steadily during training until convergence. To do so, during training, we evaluate the ground-truth CDD (denoted by CDD-G) for JAN and CAN with the ground-truth target labels. The trend of CDD and the test accuracy during training are plotted in Fig. 5.

As we see, for JAN (the blue curve), the ground-truth CDD rapidly becomes stable at a high level after a short decrease. This indicates that JAN cannot minimize the contrastive domain discrepancy effectively. For CAN (the red curve), although we can only estimate the CDD using inaccurate target label hypothesis, its CDD value steadily decreases along training. The result illustrates our estimation works as a good proxy of ground-truth contrastive domain discrepancy. And from the accuracy curve illustrated in Fig. 5, we see that minimizing CDD leads to notable accuracy improvement of CAN, compared to JAN.

**Hyper-parameter sensitivity.** We study the sensitivity of CAN to the important balance weight $\beta$ on two example tasks $A \rightarrow D$ and $D \rightarrow A$ in Fig. 5. Generally, our model is less sensitive to the change of $\beta$. In a vast range, the performance of CAN outperforms the baseline method with a large margin (the blue dashed curve). As the $\beta$ gets larger, the accuracy steadily increases before decreasing. The bell-shaped curve illustrates the regularization effect of CDD.

### 5. Conclusion

In this paper, we proposed Contrastive Adaptation Network to perform class-aware alignment for UDA. The intra-class and inter-class domain discrepancy are explicitly modeled and optimized through end-to-end mini-batch training. Experiments on real-world benchmarks demonstrate the superiority of our model compared with the strong baselines.

**Acknowledgement.** This work was supported in part by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior/Interior Business Center (DOI/IBC) contract number D17PC00340. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation/herein. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DOI/IBC, or the U.S. Government.
References