Contrastively Smoothed Class Alignment for Unsupervised Domain Adaptation

Shuyang Dai[1], Yu Cheng[2], Yizhe Zhang[3], Zhe Gan[2],
Jingjing Liu[2], and Lawrence Carin[1]

[1] Duke University, [2] Microsoft Dynamics 365 AI Research, [3] Microsoft Research
{shuyang.dai,lcarin}@duke.edu
{yu.cheng,yizhe.zhang,zhe.gan,jingjl}@microsoft.com

Abstract. Recent unsupervised approaches to domain adaptation primarily focus on minimizing the gap between the source and the target domains through refining the feature generator, in order to learn a better alignment between the two domains. This minimization can be achieved via a domain classifier to detect target-domain features that are divergent from source-domain features. However, when optimizing via such domain-classification discrepancy, ambiguous target samples that are not smoothly distributed on the low-dimensional data manifold are often missed. To solve this issue, we propose a novel Contrastively Smoothed Class Alignment (CoSCA) model, that explicitly incorporates both intra- and inter-class domain discrepancy to better align ambiguous target samples with the source domain. CoSCA estimates the underlying label hypothesis of target samples, and simultaneously adapts their feature representations by optimizing a proposed contrastive loss. In addition, Maximum Mean Discrepancy (MMD) is utilized to directly match features between source and target samples for better global alignment. Experiments on several benchmark datasets demonstrate that CoSCAoutperforms state-of-the-art approaches for unsupervised domain adaptation by producing more discriminative features.

1 Introduction

Deep neural networks (DNNs) have significantly improved the state of the art on many supervised tasks [1–4]. However, without sufficient training data, DNNs often generalize poorly to new tasks or new environments [5]. This is known as dataset bias or a domain-shift problem [6]. Unsupervised domain adaptation (UDA) [7, 8] aims to generalize a model learned from a source domain with rich annotated data to a new target domain without any labeled data. Recently, many approaches have been proposed to learn transferable representations, by simultaneously matching feature distributions across different domains [9, 10].

Motivated by [11], [12, 13] introduced a min-max game: a domain discriminator is learned by minimizing the error of distinguishing data samples from the source and the target domains, while a feature generator learns transferable features that are indistinguishable by the domain discriminator. This imposes that the learned features are domain-invariant. Additionally, a feature classifier ensures that the learned features are discriminative in the source domain. Despite promising results, these adversarial methods suffer from inherent algorithmic weaknesses [14]. Specifically, the generator may

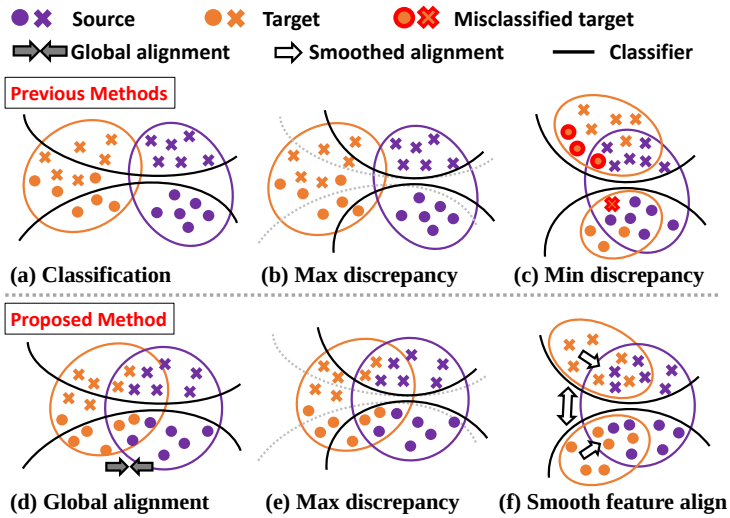


Fig. 1. Comparison between previous classifier-discrepancy-based methods and our proposed CoSCA in the *feature* space. **Top:** The region of vacancy created by maximum discrepancy reduces the smoothness of alignment between ambiguous target samples and source samples, leading to sub-optimal solutions. This problem becomes more severe when global domain alignment is not considered. **Bottom:** Demonstration of global alignment and class-conditional adaptation by using the proposed CoSCA. After classifier discrepancy is maximized, the proposed contrastive loss moves ambiguous target samples near the decision boundary towards their neighbors and separates them from non-neighbors.

manifest ambiguous features near class boundaries [15]: while the generator manages to fool the discriminator, some target-domain features may still be misclassified. In other words, the model merely aligns the global marginal distribution of the two domains and ignores the class-conditional decision boundaries.

To overcome this issue, recent UDA models further align class-level distributions by taking the decision boundary into consideration. These methods either rely on iteratively refining the decision boundary with empirical data [14, 16], or utilizing multi-view information [17]. Alternatively, the maximum classifier discrepancy (MCD) [15] model conducts a min-max game between a feature generator and two classifiers. Ambiguous target samples that are far from source-domain samples can be detected when the discrepancy between the two classifiers is maximized, as shown in Figure 1(b). Meanwhile, as the generator fools the classifiers, the generated target features may fall into the source feature regions. However, the target samples may not be smooth on the low-dimensional manifold [18, 19], meaning that neighboring samples may not belong to the same class. As a result, some generated target features could be miscategorized as shown in Figure 1(c).

In this paper, we propose the **Co**ntrastively **S**moothed **C**lass **A**lignment (CoSCA) model to improve the latent alignment of class-conditional feature distributions between

source and target domains, by alternatively estimating the underlying label hypothesis of target samples to map them into tighter clusters, and adapt feature representations based on a proposed contrastive loss. Specifically, by aligning ambiguous target samples near the decision boundaries with their neighbors and distancing them from non-neighbors, CoSCA enhances the alignment of each class in a contrastive manner. Figure 1(f) demonstrates an enhanced and smoothed version of the class-conditional alignment. Moreover, as shown in Figure 1(d), Maximum Mean Discrepancy (MMD) is included to better merge the source and target domain feature representations. The overall framework is trained end-to-end in an adversarial manner.

Our principal contributions are summarized as follows:

- We propose CoSCA, a novel approach that smooths class alignment for maximizing classifier discrepancy with a contrastive loss. CoSCA also provides better global domain alignment via the use of MMD loss.
- We validate the proposed approach on several domain adaptation benchmarks. Extensive experiments demonstrate that CoSCA achieves state-of-the-art results on several benchmarks.

2 Related Work

Unsupervised Domain Adaptation A practical solution for domain adaptation is to learn domain-invariant features whose distribution is similar across the source and the target domains. For example, Sener *et al.* [20] proposed using clustering techniques and pseudo-labels to obtain discriminative features. Long *et al.* proposed DAN [21] and JAN [22] to minimize the MMD or variations of MMD between two domains. Adversarial domain adaptation integrates adversarial learning and domain adaptation in a two-player game [13, 12, 10]. Following this idea, most existing adversarial learning methods reduce feature differences by fooling a domain discriminator [23, 8]. However, these methods fail to consider the relationship between target samples and the class-conditional decision boundaries when aligning features [15], while only merging the source and the target domains.

Class-conditional Alignment To address the aforementioned issue, recent work enforces class-level alignment while aligning global marginal distributions. Associative domain adaptation (ADA) [9] reinforces associations across domains directly in the embedding space, to extract features that are statistically domain-invariant and class-discriminative. Adversarial Dropout Regularization (ADR) [16] and Maximum Classifier Discrepancy (MCD) [15] were proposed to train a neural network in an adversarial manner, avoiding generating non-discriminative features lying in the region near the decision boundary. In [24, 22] the authors considered class information when measuring domain discrepancy. Co-regularized Domain Adaptation (Co-DA) [17] utilized multi-view information to match the marginal feature distributions corresponding to the class-conditional distributions. Compared with previous work that executed the alignment by optimizing "hard" metrics [15, 17], we propose to smooth the alignment iteratively, with explicitly defined loss.

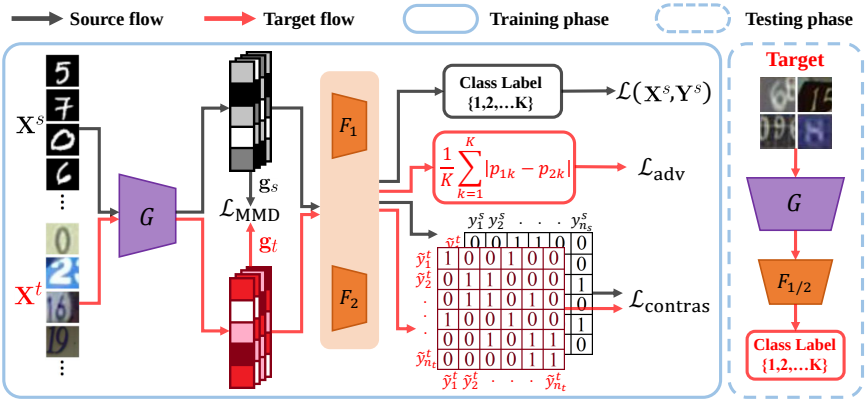


Fig. 2. Framework of the proposed CoSCA. The inputs are $\mathbf{X}^s$ with label $\mathbf{Y}^s$ from the source domain and unlabeled $\mathbf{X}^t$ from the target domain. The model contains a shared feature generator G and two feature classifiers F_1 and F_2. $\mathcal{L}_{\text{MMD}}$ is calculated using the generated feature mean of the source and target, *i.e.*, g_s and g_t respectively. $\mathcal{L}_{\text{adv}}$ is the classifier discrepancy calculated based on the probability outputs p_1 and p_2 of $F_1(G(\mathbf{X}^t))$ and $F_2(G(\mathbf{X}^t))$, respectively. $\mathcal{L}_{\text{contras}}$ is the contrastive loss calculated for both source-and-target and target-and-target samples.

Contrastive Learning The intuition for contrastive learning is to let the model understand the difference between one set ($e.g.$, data points) and another, instead of only characterizing a single set [25]. This idea has been explored in previous works that model intra-class compactness and inter-class separability ($e.g.$, distinctiveness loss [26], contrastive loss [27], triplet loss [28]) and tangent distance [29]. It has also been extended to consider several assumptions in semi-supervised and unsupervised learning [19, 30], such as the low-density region (or cluster) assumption [19, 29] that the decision boundary should lie in the low-density region, rather than crossing the high-density region. Recently, contrastive learning was applied in UDA [31], in which the intra/inter-class domain discrepancy were modeled. In comparison, our work is based on the MCD framework, utilizing the low-density assumption and focusing on separating the ambiguous target data points by optimizing the contrastive objective, allowing the decision boundary to sit in the low-density region, $i.e.$, region of vacancy, and smoothness assumption.

3 Approach

Unsupervised domain adaptation seeks to generalize a learned model from a source domain to a target domain, the latter following a different (but related) data distribution from the former. Specifically, the source- and target-domain samples are denoted $S = f(\mathbf{x}_1^s, y_1^s), ..., (\mathbf{x}_i^s, y_i^s), ..., (\mathbf{x}_{N_s}^s, y_{N_s}^s)g$, and $T = f\mathbf{x}_1^t, ..., \mathbf{x}_i^t, ..., \mathbf{x}_{N_t}^t g$, respectively, where $\mathbf{x}_i^s$ and $\mathbf{x}_i^t$ are the input, and $y_i^s \ 2 \ f1, 2, ..., Kg$ represents the data labels of K classes in the source domain. The target domain shares the same label types as the source domain, but we possess no labeled examples from the target domain. We are

interested in learning a deep network G that reduces domain shift in the data distribution across S and T, in order to make accurate predictions for y_i^t. We use the notation $(\mathbf{X}^s, \mathbf{Y}^s)$ to describe the source-domain samples and labels, and $\mathbf{X}^t$ for the unlabeled target-domain samples.

Adversarial domain adaptation approaches such as [15, 32] achieve this goal via a two-step procedure: i) train a feature generator G and the feature classifiers F_1, F_2 with the source-domain data, to ensure the generated features are class-conditional; ii) train F_1 and F_2 so that the prediction discrepancy between the two classifiers is maximized, and train G to generate features that are distinctively separated. The maximum classifier discrepancy detects the target features that are far from the support of the source domain. As the generator tries to fool the classifiers ($i.e.$, minimizing the discrepancy), these target-domain features are enforced to be categorized and aligned with the source-domain features.

However, only measuring divergence between F_1 and F_2 can be considered first-order moment matching, which may be insufficient for adversarial training. Previous work also observed similar issues [33, 34]. We address this challenge by adding the Maximum Mean Discrepancy (MMD) loss, that matches the difference via higher-order moments. Also, the class alignment in existing UDA methods takes into account the intra-class domain discrepancy only, which makes it difficult to separate samples within the same class that are close to the decision boundary. Thus, in addition to the discrepancy loss, we also measure both intra- and inter-class discrepancy across domains. Specifically, we propose to minimize the distance among target-domain features that fall into the same class based on decision boundaries, and separate those features from different categories. During this process, ambiguous target features are simultaneously kept away from the decision boundaries and mapped into the high-density region, achieving better class alignment.

3.1 Global Alignment with MMD

Following [15], we first train a feature generator $G(\,)$ and two classifiers $F_1(G(\,))$ and $F_2(G(\,))$ to minimize the softmax cross-entropy loss using the data from the labeled source domain S, defined as:

$$L(\mathbf{X}^s, \mathbf{Y}^s) = \mathsf{E}_{(\mathbf{x}^s, y^s)\ (\mathbf{X}^s, \mathbf{Y}^s)} \Big[\sum_{k=1}^{K} \mathbb{1}_{[k=y^s]} \log p_1(\mathbf{y}|\mathbf{x}^s) + \sum_{k=1}^{K} \mathbb{1}_{[k=y^s]} \log p_2(\mathbf{y}|\mathbf{x}^s) \Big] \tag{1}$$

where $p_1(\mathbf{y}|\mathbf{x})$ and $p_2(\mathbf{y}|\mathbf{x})$ are the probabilistic output of the two classifiers $F_1(G(\mathbf{x}))$ and $F_2(G(\mathbf{x}))$, respectively.

In addition to (1), we explicitly minimize the distance between the source and target feature distributions with MMD. The main idea of MMD is to estimate the distance between two distributions as the distance between sample means of the projected embeddings in a Hilbert space. Minimizing MMD is equivalent to minimizing all orders

of moments [35]. In practice, the squared value of MMD is estimated with empirical kernel mean embeddings:

$$\begin{aligned} L_{\text{MMD}}(\mathbf{X}^s, \mathbf{X}^t) &= \sum_{i=1}^{n_s} \sum_{j=1}^{n_t} k(\phi(\frac{\mathbf{g}_s}{jj\mathbf{g}_s jj}), \phi(\frac{\mathbf{g}_t}{jj\mathbf{g}_t jj})) \\ \mathbf{g}_s &= \frac{1}{n_s} \sum_{i=1}^{n_s} G(\mathbf{x}_i^s), \quad \mathbf{g}_t = \frac{1}{n_t} \sum_{i=1}^{n_t} G(\mathbf{x}_i^t) \end{aligned} \tag{2}$$

where $\phi(\ )$ is the kernel mapping, $\mathbf{g}_s \ 2 \ R^n$, $\mathbf{g}_t \ 2 \ R^n$, with n_t and n_s denoting the size of a training mini-batch of the data from the source domain S and the target domain T, respectively; $jj \quad jj$ denotes the ℓ_2-norm. With the MMD loss L_{MMD}, the normalized features in the two domains are encouraged to be distributed identically, leading to better global domain alignment.

3.2 Contrastively Smoothed Class Alignment

Discrepancy Loss The discrepancy loss represents the level of disagreement between the two feature classifiers in prediction for target-domain samples. Specifically, the discrepancy loss between F_1 and F_2 is defined as:

$$d(p_1(\mathbf{y}j\mathbf{x}), p_2(\mathbf{y}j\mathbf{x})) = \frac{1}{K} \sum_{k=1}^{K} \left| p_{1_k}(\mathbf{y}j\mathbf{x}) \quad p_{2_k}(\mathbf{y}j\mathbf{x}) \right| \tag{3}$$

where $j \quad j$ denotes the ℓ_1-norm, and $p_{1_k}(\ )$ and $p_{2_k}(\ )$ are the probability output of p_1 and p_2 for the k-th class, respectively. Accordingly, we can define the discrepancy loss over the target domain T:

$$L_{\text{adv}}(\mathbf{X}^t) = \mathbb{E}_{\mathbf{x}^t \quad \mathbf{X}^t} \left[d(p_1(\mathbf{y}j\mathbf{x}^t), p_2(\mathbf{y}j\mathbf{x}^t)) \right] \tag{4}$$

Adversarial training is conducted in the MCD [15] setup:

$$\begin{aligned} &\min_{F_1, F_2} L(\mathbf{X}^s, \mathbf{Y}^s) \quad \lambda L_{\text{adv}}(\mathbf{X}^t) \\ &\min_{G} L_{\text{adv}}(\mathbf{X}^t) \end{aligned} \tag{5}$$

where λ is a hyper-parameter. Minimizing the discrepancy between the two classifiers F_1 and F_2 induces smoothness for the clearly classified target-domain features, while the region in the vacancy among the ambiguous ones remains non-smooth. Moreover, MCD only utilizes the unlabeled target-domain samples, while ignoring the labeled source-domain data when estimating the discrepancy.

Contrastive Loss To further optimize G to estimate the underlying label hypothesis of target-domain samples, we propose to measure the intra- and inter-class discrepancy across domains, conditional on class information. By using an indicator defined as $c(y, y') = \begin{cases} 1, & y = y' \\ 0, & y \neq y' \end{cases}$, we define the contrastive loss between S and T as:

$$L_{\text{contras}}^{S \$ T} = \sum_{\mathbf{x}_i^s 2 S, \mathbf{x}_j^t 2 T} L_{\text{dis}}(G(\mathbf{x}_i^s), G(\mathbf{x}_j^t), c(y_i^s, \widetilde{y}_j^t)) \tag{6}$$

where L_{dis} is a distance measure (defined below), and $\widetilde{y}_j^t$ is the predicted target label for $\mathbf{x}_j^t$. Specifically, (6) covers two types of class-aware domain discrepancies: i) intra-class domain discrepancy ($y_i^s = \widetilde{y}_j^t$); and ii) inter-class domain discrepancy ($y_i^s \neq \widetilde{y}_j^t$). Note that y_i^s is known, providing some supervision for parameter learning. Similarly, we can define the constrastive loss between T and T as:

$$L_{\text{contras}}^{T\,\$\,T} = \sum_{\mathbf{x}_i^t, \mathbf{x}_j^t \in T} L_{\text{dis}}(G(\mathbf{x}_i^t), G(\mathbf{x}_j^t), c(\widetilde{y}_i^t, \widetilde{y}_j^t)) \tag{7}$$

To obtain the indicator $c(y, y')$, estimated target label $\widetilde{y}_i^t$ is required. Specifically, for each data sample $\mathbf{x}_j^t$, a pseudo label is predicted based on the maximum posterior probability of the two classifiers:

$$\begin{aligned} \widetilde{y}_j^t = \underset{k \in \{1,2,\ldots,K\}}{\arg\max} \Big\{ & p(F_1(G(\mathbf{x}_j^t)) = k | \mathbf{x}) \\ & + p(F_2(G(\mathbf{x}_j^t)) = k | \mathbf{x}) \Big\} \end{aligned} \tag{8}$$

Ideally, based on the indicator, L_{dis} should ensure the gathering of features that fall in the same class, while separating those in different categories. Following [19], we utilize contrastive Siamese networks [36], which can learn an invariant mapping to a smooth and coherent feature space and perform well in practice:

$$L_{\text{dis}} = \begin{cases} \|G(\mathbf{x}_i) - G(\mathbf{x}_j)\|^2 & c_{ij} = 1 \\ \max(0, m - \|G(\mathbf{x}_i) - G(\mathbf{x}_j)\|)^2 & c_{ij} = 0 \end{cases} \tag{9}$$

where $c_{ij} = c(y_i, y_j)$ and m is a pre-defined margin. The margin loss constrains the neighboring features to be consistent. Based on the above definitions of source-and-target and target-and-target contrastive losses, the overall objective is:

$$L_{\text{contras}}(\mathbf{X}^s, \mathbf{Y}^s, \mathbf{X}^t) = L_{\text{contras}}^{S\,\$\,T} + L_{\text{contras}}^{T\,\$\,T} \tag{10}$$

Minimizing the contrastive loss L_{contras} encourages features in the same class to aggregate together while pushing unrelated pairs away from each other. In other words, the semantic feature approximation is enhanced to induce smoothness between data in the feature space.

3.3 Training Procedure

We optimize G, F_1 and F_2 by combining all of the aforementioned losses, performed in an adversarial training manner. Specifically, we first train the classifiers F_1 and F_2 and the generator G to minimize the objective:

$$\min_{F_1, F_2, G} L(\mathbf{X}^s, \mathbf{Y}^s) + \lambda_1 L_{\text{MMD}}(\mathbf{X}^s, \mathbf{X}^t) \tag{11}$$

We then train the classifiers F_1 and F_2 while keeping the generator G fixed. The objective is:

$$\min_{F_1, F_2} L(\mathbf{X}^s, \mathbf{Y}^s) - \lambda_2 L_{\text{adv}}(\mathbf{X}^t) \tag{12}$$

Algorithm 1 Training procedure of CoSCA.

Input: Source domain samples $\{\mathbf{x}_i^s; y_i^s\}$, and target domain samples $\{\mathbf{x}_j^t\}$. Hyper-parameters $_1$, $_2$, $_3$, and inner-loop iteration and .
Output: Classifiers F_1 and F_2, and generator G.

for *iter* from 1 to *max_iter* **do**
  Sample a mini-batch of source samples $[\mathbf{x}_i^s; y_i^s]$ and target samples $\left[\mathbf{x}_j^t\right]$.
  # Update both the generator and the classifiers
  Compute $\mathcal{L}(\mathbf{X}^s; \mathbf{Y}^s)$ on $[\mathbf{x}_i^s; y_i^s]$.
  Compute $\mathcal{L}_{\text{MMD}}(\mathbf{X}^s; \mathbf{X}^t)$ on $\left[\mathbf{x}_i^s; \mathbf{x}_j^t\right]$.
  Update G, F_1 and F_2 using (11).
  # Update the classifiers
  for *inner_loop_iter*$_1$ from 1 to **do**
    Compute $\mathcal{L}(\mathbf{X}^s; \mathbf{Y}^s)$ on $[\mathbf{x}_i^s; y_i^s]$.
    Compute $\mathcal{L}_{\text{adv}}(\mathbf{X}^t)$ on $\mathbf{x}_j^t$.
    Fix G, update F_1 and F_2 using (12).
  end for
  # Update the feature generator
  for *inner_loop_iter*$_2$ from 1 to **do**
    Compute $\mathcal{L}_{\text{adv}}(\mathbf{X}^t)$ on $\mathbf{x}_j^t$.
    Compute $\mathcal{L}_{\text{contras}}(\mathbf{X}^s; \mathbf{Y}^s; \mathbf{X}^t)$ on $\left[\mathbf{x}_i^s; y_i^s; \mathbf{x}_j^t\right]$.
    Fix F_1 and F_2, update G using (13).
  end for
end for

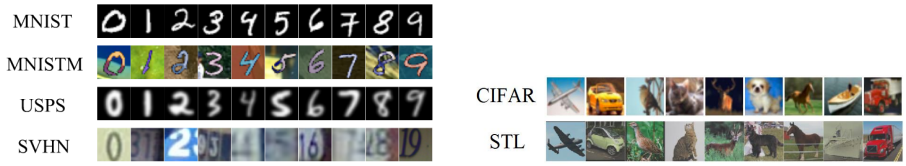
Lastly, we train the generator G with the following objective, while keeping both F_1 and F_2 fixed:

$$\min_G \ \lambda_2 L_{\text{adv}}(\mathbf{X}^t) + \lambda_3 L_{\text{contras}}(\mathbf{X}^s, \mathbf{Y}^s, \mathbf{X}^t) \tag{13}$$

where λ_1, λ_2 and λ_3 are hyper-parameters that balance the different objectives. These steps are repeated, with the full approach summarized in Algorithm 1. In our experiments, the inner-loop iteration numbers τ and δ are both set to 2.

Class-aware sampling When training with the contrastive loss, it is important to sample a mini-batch of data with all the classes, to allow (10) to be fully trained. Following [31], we use a class-aware sampling strategy. Specifically, we randomly select a subset of each class, from which a mini-batch is sampled. Consequently, in each mini-batch, we are able to estimate the intra/inter-class discrepancy.

Dynamic parameterization of λ_3 In our implementation, we adapt a dynamic $\omega(t)$ to parameterize λ_3. We set $\omega(t) = \exp[\ \theta(1 - \frac{t}{\text{max-epochs}})]\lambda_3$, which is a Gaussian curve ranging from 0 to λ_3; this is employed to prevent unlabeled target features gathering in the early stage of training, as the pseudo labels might be unreliable.



(a) The Digit dataset. (b) The CIFAR-10 dataset and the STL dataset.

Fig. 3. Sample images from the Digit, CIFAR-10 and STL datasets. Images from each column belong to the same class, while each row corresponds to a domain.

4 Experiments

We evaluate the proposed model primarily on image datasets. To compare with MCD [15] as well as the state-of-the-art results in [14, 17], we evaluate on the same datasets used in those studies: the digit datasets ($i.e.$, MNIST, MNISTM, Street View House Numbers (SVHN), and USPS), CIFAR-10, and STL-10. We also conduct experiments on the VisDA dataset, *i.e.*, large-scale images. Our model can also be applied to non-visual domain adaptation tasks. Specifically, to show the flexibility of our model, we also evaluate it on the Amazon Reviews dataset.

For visual domain adaptation tasks, the proposed model is implemented based on VADA [14] and Co-DA [17] to avoid any incidental difference caused by network architecture. However, different from these methods, our model does not require a discriminator, and only adopts the architecture for the feature generator G and the classifier F. Specifically, G has 9 convolutional layers with several dropout, max-pool, Gaussian noise and global pool layers (details can be found in the Supplementary Material). Both F_1 and F_2 are one-layer MLPs. We also include instance normalization [14, 37], achieving superior results on several benchmarks. For the VisDA dataset, we implemented our model based on Self-ensembling Domain Adaptation (SEDA) [38]. To compare with MCD and Contrastive Adaptation Network (CAN) [31] (codebase not available) in both experiments, we re-implemented them using the exact architecture as our model.

In addition to the aforementioned baseline models, we also include results from recently proposed unsupervised domain adaptation models. Note that standard domain adaptation methods (such as Transfer Component Analysis (TCA) [39] and Subspace Alignment (SA) [40]) are not included; these models only work on pre-extracted features, and are often not scalable to large datasets. Instead, we mainly compare our model with methods based on adversarial neural networks.

For the non-visual task, we adopt a one-layer CNN structure from previous work [41]. The feature generator G consists of three components, including a 300-dimensional word embedding layer using GloVe [42], a one-layer CNN with ReLU, and a max-over-time pooling through which the final sentence representation is obtained. The classifiers F_1 and F_2 can be decomposed into one dropout layer and one fully connected output layer.

Source Domain Target Domain	MNIST SVHN	SVHN MNIST	MNIST MNISTM	MNIST USPS	CIFAR STL	STL CIFAR
MMD [21]	-	71.1	76.9	81.1	-	-
DANN [8]	35.7	71.1	81.5	77.1	-	-
DSN [44]	40.1	82.7	83.2	91.3	-	-
ATT [45]	52.8	86.2	94.2	-	-	-
With Instance-Normalized Input:						
Souce-Only	40.9	82.4	59.9	76.7	77.0	62.6
VADA [14]	73.3	94.5	95.7	-	78.3	71.4
Co-DA [17]	**81.3**	98.6	97.3	-	80.3	74.5
MCD [15]	68.7	96.2$^{\gamma}$	96.7	94.2$^{\gamma}$	78.1	69.2
SEDA [38]	37.5	**99.2**	-	98.2	80.1	74.2
CAN [31]	67.1	94.8	96.2	97.5	77.3	70.4
CoSCA	80.7	98.7	**98.9**	**99.3**	**81.7**	**75.2**

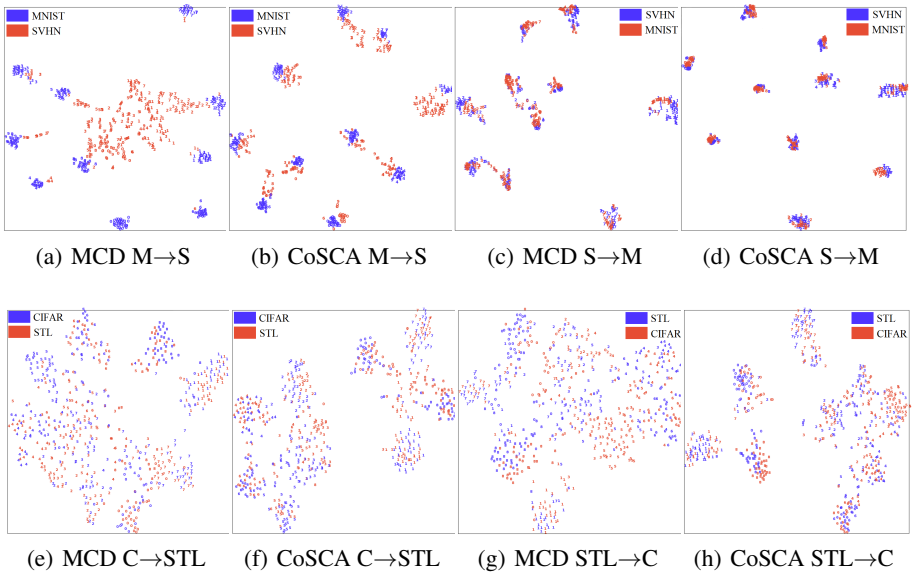
Table 1. Results on visual domain adaptation tasks. Source-Only corresponds to training a classifier in the source domain and applying it directly to the target domain, without any adaptation. Models with instance-normalized input are implemented using the same network architecture. Results with † are reported in [15].

4.1 Digit Datasets

There are four types of digit images ($i.e.$, four domains). MNIST and USPS are both hand-written gray-scale images, with relatively small domain difference. MNISTM [8] is a dataset built upon MNIST by adding randomly colored image patches from BSD500 dataset [43]. SVHN includes colored images of street numbers. All images are rescaled to 32 32 3. Sample images of all four digit datasets are presented in Figure 3(a).

MNIST *!* SVHN While both MNIST and SVHN include images of digits, there exists a large domain gap between these two datasets. As gray-scale handwritten digits, MNIST has much lower dimensionality than SVHN, which contains cropped street-view images of house numbers. Specifically, each image from SVHN has a colored background, which sometimes contains multiple digits, and might be blurry. This makes MNIST *!* SVHN a much harder adaptation task than other digit datasets. It is shown recently in [14] that instance normalization allows the classifier to be invariant to channel-wide scaling and shifting of the input pixel intensities, greatly improving the adaptation performance on MNIST *!* SVHN (73.3%). With instance normalization, our proposed CoSCA achieves test accuracy of 80.7%, as shown in Table 1, competitive with state-of-the-art results from [17].

Notice that MCD does not provide adequate performance. Figure 4(a) plots the t-SNE embedding of the features learned by MCD. Domains are indicated by different colors, and classes are indicated by different digit numbers. MCD fails to align the features of the two domains globally due to the large domain gap. In other words, the maximized discrepancy provides too many ambiguous target-domain samples. As a result, the feature generator may not be able to properly align them with the source-domain samples. In comparison, as shown in Figure 4(b), CoSCA utilizes the MMD between the source and the target domain features, thus maintaining a better global



(a) MCD M→S (b) CoSCA M→S (c) MCD S→M (d) CoSCA S→M

(e) MCD C→STL (f) CoSCA C→STL (g) MCD STL→C (h) CoSCA STL→C

Fig. 4. t-SNE embedding of the features $G(x)$ for MNIST (M) → SVHN (S) and STL → CIFAR (C). Color indicates domain, and the digit number is the label. The ideal situation is to mix the two colors with the same label, representing domain-invariant features. The t-SNE plots for the other datasets are provided in the Supplementary Material.

domain alignment. With further smoothed class-conditional adaptation, it outperforms MCD.

SVHN *!* MNIST Classification with the MNIST dataset is easier than others. As shown in Table 1, source-only achieves 82.4% on SVHN *!* MNIST with instance normalization. Therefore, even with the same amount of domain difference, performance on SVHN *!* MNIST is much better than MNIST *!* SVHN across all compared models. The test accuracy of our model achieves 98.7%.

MNIST *!* MNISTM Since MNISTM is a colored version of MNIST, there exists a one-to-one matching between the two datasets, $i.e.$, a domain adaptation model would perform well as long as domain-invariant features are properly extracted. CoSCA provides better results than Co-DA, yielding a test accuracy of 98.9%.

MNIST *!* USPS Evaluation on MNIST and USPS datasets is also conducted to compare our model with other baselines. The proposed method achieves an excellent result of 99.3%.

4.2 CIFAR-10 and STL-10 Datasets

CIFAR-10 and STL-10 are both 10-class datasets, with each image containing an animal or a type of transportation. Images from each class are much more diverse than the digit datasets, with higher intrinsic dimensionality, which makes it a harder domain

Model	plane	bcycl	bus	car	horse	knife	mcycl	person	plant	sktbrd	train	truck	mean
Source Only	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$^{\gamma}$
MMD [21]	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$^{\gamma}$
DANN [8]	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$^{\gamma}$
MCD [15]	89.1	80.8	82.9	70.9	91.6	56.5	89.5	79.3	90.9	76.1	88.3	29.3	77.1
CAN [31]	91.4	78.9	79.1	72.8	93.2	63.4	82.4	68.6	93.2	88.3	84.1	**39.2**	77.9
SEDA [38]	95.3	87.1	84.2	58.3	94.4	**89.6**	87.9	79.1	92.8	**91.3**	**89.6**	37.4	82.2
CoSCA	**95.7**	**87.4**	**85.7**	**73.5**	**95.3**	72.8	**91.5**	**84.8**	**94.6**	87.9	87.9	36.8	**82.9**

Table 2. Test accuracy of ResNet101 model fine-tuned on the VisDA dataset. Results with † are reported in [15], while the others are implemented using the same network architecture.

adaptation task. There are 9 overlapping classes between these two datasets. Figure 3(b) shows some sample images from each class. CIFAR provides images of size 32 32 and a large training set of 50,000 image samples, while STL contains higher quality images of size 96 96, but with a much smaller training set of 5,000 samples. Following [38, 14, 17], we remove non-overlapping classes from these two datasets and resize the images from STL to 32 32.

STL *!* CIFAR is more difficult than CIFAR *!* STL, due to the small training set in STL. For the latter, the source-only model with no adaptation involved achieves an accuracy of 77.0%. With adaptation, the margin-of-improvement is relatively small, while CoSCA provides the best improvement of 4.7% among all the models (Table 1). For STL *!* CIFAR, our model yields a 12.6% margin-of-improvement and an accuracy of 75.2%. Figures 4(e), 4(f), 4(g), and 4(h) provide t-SNE plots for MCD and our model, respectively, which shows our model achieves much better alignment for each class.

4.3 VisDA Dataset

The VisDA dataset is a large-scale image dataset that evaluates the adaptation from synthetic-object to real-object images. Images from the source domain are synthetic renderings of 3D models from different angles and lighting conditions. There are 152,397 image samples in the source domain, and 55,388 image samples in the target domain. The image size, after rescaling as in [15], is 224 224 3. A model architecture with ResNet101 [4] pre-trained on Imagenet is required. There are 12 different object categories in VisDA, shared by the source and the target domains.

Table 2 shows the test accuracy of different models in all object classes. The class-aware methods, namely MCD [15], SEDA [38] and our proposed CoSCA, outperform the source only model in all categories. In comparison, the methods that are mainly based on distribution matching do not perform well in some of the categories. CoSCA outperforms MCD, showing the effectiveness of contrastive loss and MMD global alignment. In addition, it performs better than SEDA in most categories, demonstrating its robustness in handling large scale images.

	Source-Only	DANN [8]	PBLM [46]	MCD [15]	DAS [47]	CoSCA
Accuracy	79.13	80.29$^{\gamma}$	80.40$^{\gamma}$	81.35	81.96$^{\gamma}$	**83.17**

Table 3. Results on the Amazon Reviews dataset. Results with † are reported by [47, 46].

Model	MNIST SVHN	STL CIFAR	Amazon Reviews
MCD [15]	68.7	69.2	81.35
MCD+MMD	72.1	70.2	81.73
MCD+Contras	75.9	73.4	82.56
CoSCA	**80.7**	**75.2**	**83.17**

Table 4. Ablation study on CoSCA with different variations of MCD on MNIST→SVHN, STL→CIFAR, and Amazon Reviews.

4.4 Amazon Reviews Dataset

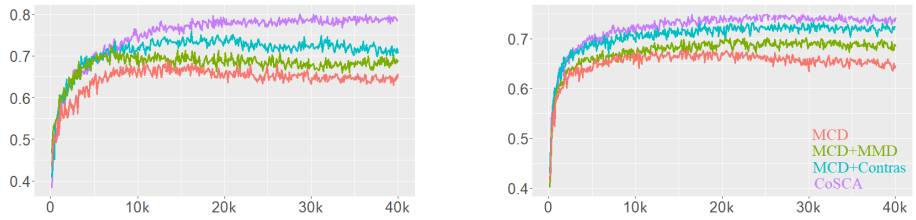
We also evaluate CoSCA on the Amazon Reviews dataset collected by Blitzer *et al.* [48]. It contains reviews from several different domains, with 1000 positive and 1000 negative reviews in each domain.

Table 3 shows the average classification accuracy of different methods, including DANN [8] DAS [47] and PBLM [46]. We use the same model architecture and parameter setting for MCD and the source-only model. Results show that the proposed CoSCA outperforms all other methods. Specifically, it improves the performance from test accuracy of 81.96% to 83.17%, when compared to the state-of-the-art method DAS. MCD achieves 81.35%, which is also outperformed by CoSCA.

4.5 Ablation Study

To further demonstrate the improvement of CoSCA over MCD [15], we conduct an ablation study. Specifically, with the same network architecture and setup, we compare model performance among 1) MCD, 2) MCD with smooth alignment (MCD+Contras), 3) MCD with global alighnment (MCD+MMD), and 4) CoSCA, to validate the effectiveness of adding contrastive loss L_{contras} and MMD loss L_{MMD} to MCD. As MCD has already achieved superior performance on some of the benchmark datasets, we mainly choose those tasks on which MCD does not perform very well, in order to better analyze the margin of improvement. Therefore, MNIST *!* SVHN, STL *!* CIFAR and Amazon Reviews are selected for this experiment (Table 4).

Effect of Contrastive Alignment We compare CoSCA with MCD as well as its few variations, to validate the effectiveness of the proposed contrastive alignment. Table 4 provides the test accuracy for every model across the selected benchmark datasets. For MNIST *!* SVHN, MCD+Contrastive outperforms MCD by 7.2%. For STL *!* CIFAR and Amazon Reviews, the margin of improvement is 4.2% and 1.21%, respectively (less significant than MNIST *!* SVHN, possibly due to the smaller domain difference).



(a) Validation accuracy vs number of iterations for MNIST→SVHN.

(b) Validation accuracy vs number of iterations for STL→CIFAR.

Fig. 5. Ablation study on CoSCA and different variations of MCD.

Note that the results of MCD+Contras are still worse than CoSCA, demonstrating the effectiveness of the global domain alignment and the framework design of our model.

Effect of MMD We further investigate how the MMD loss can impact the performance of our proposed CoSCA. Specifically, for MNIST *!* SVHN, MCD+MMD achieves a test accuracy of 72.1%, only lifting the original result of MCD by 3.4%. For STL *!* CIFAR and Amazon Reviews, the margin-of-improvement is 1.0% and 0.38%, respectively. While this validates the effectiveness of having global alignment in the MCD framework, the improvement is small. Without a smoothed class-conditional alignment, MCD still encounters misclassified target features during training, leading to a sub-optimal solution. Notice that when comparing CoSCA with MCD+Contras, the improvement for MNIST *!* SVHN is significant (as shown in Figure 5(a)), with validation accuracy and training stability enhanced. This demonstrates the importance of global alignment when there exists a large domain difference.

5 Conclusions

We have proposed Contrastively Smoothed Class Alignment (CoSCA) for the UDA problem, by explicitly combining intra-class and inter-class domain discrepancy and optimizing class alignment through end-to-end mini-batch training. Experiments on several benchmarks demonstrate that our model can outperform state-of-the-art baselines. Our experimental analysis shows that CoSCA learns more discriminative target domain features, and the introduced MMD feature matching improves the global domain alignment. For future work, we will extend our model to other domain-adaptation tasks. Another direction to explore concerns development of a theoretical interpretation of contrastive learning for domain adaptation, particularly characterizing its effects on the alignment of source and target domain feature distributions.

Acknowledgements

The authors would like to thank the anonymous reviewers for their insightful comments. The research at Duke University was supported in part by DARPA, DOE, NIH, NSF and ONR.

References

1. Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., Darrell, T.: Decaf: A deep convolutional activation feature for generic visual recognition. In: ICML. (2014)
2. Yosinski, J., Clune, J., Bengio, Y., Lipson, H.: How transferable are features in deep neural networks? In: NeurIPS. (2014)
3. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: ICLR. (2015)
4. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. (2016)
5. Torralba, A., Efros, A.A.: Unbiased look at dataset bias. In: CVPR. (2011)
6. Gretton, A., Smola, A.J., Huang, J., Schmittfull, M., Borgwardt, K.M., Schölkopf, B.: Covariate shift by kernel mean matching. In: MIT press. (2009)
7. Pan, S.J., Yang, Q., et al.: A survey on transfer learning. IEEE Transactions on knowledge and data engineering (2010)
8. Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., Lempitsky, V.: Domain-adversarial training of neural networks. JMLR (2016)
9. Haeusser, P., Frerix, T., Mordvintsev, A., Cremers, D.: Associative domain adaptation. In: ICCV. (2017)
10. Tzeng, E., Hoffman, J., Darrell, T., Saenko, K.: Simultaneous deep transfer across domains and tasks. In: ICCV. (2015)
11. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: NeurIPS. (2014)
12. Tzeng, E., Hoffman, J., Saenko, K., Darrell, T.: Adversarial discriminative domain adaptation. In: CVPR. (2017)
13. Ganin, Y., Lempitsky, V.: Unsupervised domain adaptation by backpropagation. arXiv preprint arXiv:1409.7495 (2014)
14. Shu, R., Bui, H.H., Narui, H., Ermon, S.: A dirt-t approach to unsupervised domain adaptation. In: ICLR. (2018)
15. Saito, K., Watanabe, K., Ushiku, Y., Harada, T.: Maximum classifier discrepancy for unsupervised domain adaptation. In: CVPR. (2018)
16. Saito, K., Ushiku, Y., Harada, T., Saenko, K.: Adversarial dropout regularization. arXiv preprint arXiv:1711.01575 (2017)
17. Kumar, A., Sattigeri, P., Wadhawan, K., Karlinsky, L., Feris, R., Freeman, B., Wornell, G.: Co-regularized alignment for unsupervised domain adaptation. In: NeurIPS. (2018)
18. Chapelle, O., Scholkopf, B., Zien, A.: Semi-supervised learning. IEEE Transactions on Neural Networks (2009)
19. Luo, Y., Zhu, J., Li, M., Ren, Y., Zhang, B.: Smooth neighbors on teacher graphs for semi-supervised learning. In: CVPR. (2017)
20. Sener, O., Song, H.O., Saxena, A., Savarese, S.: Learning transferrable representations for unsupervised domain adaptation. In: NeurIPS. (2016)
21. Long, M., Cao, Y., Wang, J., Jordan, M.: Learning transferable features with deep adaptation networks. In: ICML. (2015)
22. Long, M., Zhu, H., Wang, J., Jordan, M.I.: Unsupervised domain adaptation with residual transfer networks. In: NeurIPS. (2016)
23. Long, M., CAO, Z., Wang, J., Jordan, M.I.: Conditional adversarial domain adaptation. In: NeurIPS. (2018)
24. Pei, Z., Cao, Z., Long, M., Wang, J.: Multi-adversarial domain adaptation. In: AAAI. (2018)
25. Zou, J.Y., Hsu, D.J., Parkes, D.C., Adams, R.P.: Contrastive learning using spectral methods. In: NeurIPS. (2013)

26. Dai, B., Lin, D.: Contrastive learning for image captioning. In: NeurIPS. (2017)
27. Hadsell, R., Chopra, S., LeCun, Y.: Dimensionality reduction by learning an invariant mapping. In: CVPR. (2006)
28. Wang, J., Song, Y., Leung, T., Rosenberg, C., Wang, J., Philbin, J., Chen, B., Wu, Y.: Learning fine-grained image similarity with deep ranking. In: CVPR. (2014)
29. Rifai, S., Dauphin, Y.N., Vincent, P., Bengio, Y., Muller, X.: The manifold tangent classifier. In: NeurIPS. (2011)
30. Li, C., Xu, K., Zhu, J., Zhang, B.: Triple generative adversarial nets. In: NeurIPS. (2017)
31. Kang, G., Jiang, L., Yang, Y., Hauptmann, A.G.: Contrastive adaptation network for unsupervised domain adaptation. In: CVPR. (2019)
32. Kim, M., Sahu, P., Gholami, B., Pavlovic, V.: Unsupervised visual domain adaptation: A deep max-margin gaussian process approach. arXiv preprint arXiv:1902.08727 (2019)
33. Arora, S., Ge, R., Liang, Y., Ma, T., Zhang, Y.: Generalization and equilibrium in generative adversarial nets (GANs). In: ICML. (2017)
34. Tsai, Y.H., Hung, W.C., Schulter, S., Sohn, K., Yang, M.H., Chandraker, M.: Learning to adapt structured output space for semantic segmentation. In: CVPR. (2018)
35. Gretton, A., Borgwardt, K.M., Rasch, M.J., Schölkopf, B., Smola, A.: A kernel two-sample test. JMLR (2012)
36. Bromley, J., Guyon, I., LeCun, Y., Säckinger, E., Shah, R.: Signature verification using a "siamese" time delay neural network. In: NeurIPS. (1994)
37. Ulyanov, D., Vedaldi, A., Lempitsky, V.: Instance normalization: The missing ingredient for fast stylization. arXiv preprint arXiv:1607.08022 (2016)
38. French, G., Mackiewicz, M., Fisher, M.: Self-ensembling for domain adaptation. In: ICLR. (2018)
39. Pan, S.J., Tsang, I.W., Kwok, J.T., Yang, Q.: Domain adaptation via transfer component analysis. IEEE Transactions on Neural Networks (2011)
40. Fernando, B., Habrard, A., Sebban, M., Tuytelaars, T.: Unsupervised visual domain adaptation using subspace alignment. In: ICCV. (2013)
41. Kim, Y.: Convolutional neural networks for sentence classification. In: EMNLP. (2014)
42. Pennington, J., Socher, R., Manning, C.: Glove: Global vectors for word representation. In: EMNLP. (2014)
43. Arbelaez, P., Maire, M., Fowlkes, C., Malik, J.: Contour detection and hierarchical image segmentation. PAMI (2011)
44. Bousmalis, K., Trigeorgis, G., Silberman, N., Krishnan, D., Erhan, D.: Domain separation networks. In: NeurIPS. (2016)
45. Saito, K., Ushiku, Y., Harada, T.: Asymmetric tri-training for unsupervised domain adaptation. In: ICML. (2017)
46. Ziser, Y., Reichart, R.: Pivot based language modeling for improved neural domain adaptation. In: ACL. (2018)
47. He, R., Lee, W.S., Ng, H.T., Dahlmeier, D.: Adaptive semi-supervised learning for cross-domain sentiment classification. In: ACL. (2018)
48. Blitzer, J., Dredze, M., Pereira, F.: Domain adaptation for sentiment classification. In: ACL. (2007)