# Distant Supervised Centroid Shift: A Simple and Efficient Approach to Visual Domain Adaptation [*]

Jian Liang, Ran He, Zhenan Sun and Tieniu Tan
♮ CRIPAC & NLPR, Institute of Automation, Chinese Academy of Sciences (CAS), ♯ University of Chinese Academy of Sciences, ♭ CAS Center for Excellence in Brain Science and Intelligence Technology
liangjian92@gmail.com, {jian.liang,rhe,znsun,tnt}@nlpr.ia.ac.cn

## Abstract

*Conventional domain adaptation methods usually resort to deep neural networks or subspace learning to find invariant representations across domains. However, most deep learning methods highly rely on large-size source domains and are computationally expensive to train, while subspace learning methods always have a quadratic time complexity that suffers from the large domain size. This paper provides a simple and efficient solution, which could be regarded as a well-performing baseline for domain adaptation tasks.*

*Our method is built upon the nearest centroid classifier, seeking a subspace where the centroids in the target domain are moderately shifted from those in the source domain. Specifically, we design a unified objective without accessing the source domain data and adopt an alternating minimization scheme to iteratively discover the pseudo target labels, invariant subspace, and target centroids. Besides its privacy-preserving property (distant supervision), the algorithm is provably convergent and has a promising linear time complexity. In addition, the proposed method can be readily extended to multi-source setting and domain generalization, and it remarkably enhances popular deep adaptation methods by borrowing the learned transferable features. Extensive experiments on several benchmarks including object, digit, and face recognition datasets validate that our methods yield state-of-the-art results in various domain adaptation tasks.*

## 1. Introduction

Traditional machine learning paradigms always assume that the training data and the testing data come from the same distribution, however, this assumption does not always hold in real-world applications [52, 66]. To avoid the expensive and time-consuming data labeling step, massive efforts

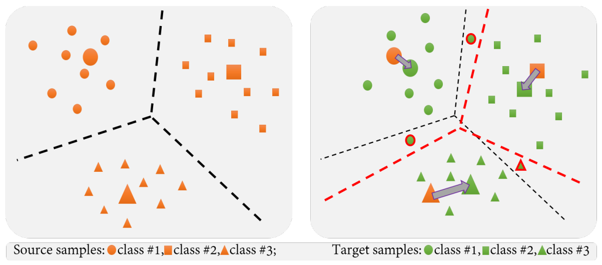

Source samples: ●class #1,■class #2,▲class #3;
Target samples: ●class #1,■class #2,▲class #3

Figure 1. Illustrative example of centroid shift (arrows) in our approach. Larger markers indicate the **source** and **target** centroids.

over the last decade have been devoted to transfer learning [64, 8, 34, 36] and multi-task learning [14, 30, 63, 70] that leverage the latent relationship with previous datasets to learn a new model for an emerging dataset.

Taking a night object image recognition problem for example, it is not desirable to neither train a model on existing day-time object images to recognize these night object images nor acquire massive labeled object images at night to re-train a model on them from scratch. By contrast, we expect to transfer the knowledge from existing day-time object images to recognizing these unlabeled night object images, which is also known as domain adaptation, and such day-time and night images are termed as source and target domains, respectively. Based on the availability of partial labeled target data, domain adaptation can be roughly divided into two categories, unsupervised and semi-supervised domain adaptation. In this paper, we mainly focus on the challenging unsupervised domain adaptation problem where the labels of target data are totally unknown.

Since the degradation in performance mainly arises from the covariate shift (i.e., the change in the data distribution of the source and target domains), early approaches [72, 58] favor an intuitive strategy named instance re-weighting, which tries to align two different domains via estimating naturally the ratio between the likelihoods of being a source or target example. Later studies [42, 7] exploit one favorite distribution measure named Maximum Mean Discrepancy (MMD) [24] to weigh data instances. However,

[*]This work is funded by the National Natural Science Foundation of China (Grants No. 61622310, 61721004), and Beijing Natural Science Foundation (Grants No. JQ18017).

these instance-based adaptation methods require the strict assumptions [52] that are hard to satisfy.

Alternatively, a growing number of recent studies focus on learning transferable representations via deep neural networks or subspace discovery since they do not require the strong assumption. Deep domain adaptation methods are roughly divided into three main categories, discrepancy-based methods [60, 39, 44, 73, 68], adversarial-based methods [17, 38, 2, 63, 64], and reconstruction-based methods [19, 3, 76]. However, these batch-wise deep learning methods cannot fully exploit the global information and address small-size source domains well. Subspace alignment methods [23, 15] try to align the subspaces (e.g., PCA) of different domains together. Later studies [59, 32, 73] further consider the second-order and higher-order scatter matrices (moments) across different domains. These subspace-centric methods are rather easy and efficient to deploy, yet, they fail to minimize the data distributions between domains after aligning the subspaces. [41] is a typical data-centric subspace discovery based approach that learns to project both domains onto one subspace where the cross-domain joint distribution discrepancy is minimized. Following studies [74, 36, 67] are built upon [41] by considering coupled projections, discriminative target structure and joint classifier learning, respectively. Even these data-centric methods achieve promising results, yet, they always involve a heuristic pseudo target label estimation step in the EM-like algorithm, making the overall optimization problem hard to converge theoretically. Additionally, all the methods mentioned above involve several large MMD matrices [41], thus, they all have a quadratic time complexity w.r.t. the domain size and can not cope with large-scale datasets well.

In this paper, we propose a simple, efficient, yet effective approach via subspace discovery for unsupervised domain adaptation. Inspired by [12], we develop a unified objective that assumes the centroids in the target domain are moderately shifted from those in the source domain, and each instance is closest to its corresponding centroid for both domains in the projected subspace. *Note that, this objective does not need to access the source domain data like [6], making it a privacy-preserving method.* Then we adopt an alternating minimization scheme to iteratively discover the pseudo target labels, adaptive target centroids, and invariant subspace learning. Specifically, each subproblem has a closed-form solution. Theoretical analysis shows that our algorithm is convergent and efficient with a linear time complexity. In addition, the proposed method can be readily extended to multi-source setting and domain generalization, and it even remarkably enhances popular deep domain methods by borrowing the learned transferable features. Extensive experiments on several benchmarks including object (*i.e.*, Office31 [55], Office-Caltech [21] and Office-Home [66], VLCS [62]), Digits, and face (*i.e.*, PIE [1]) recognition datasets validate that our methods achieve state-of-the-art results in the vanilla unsupervised domain adaptation, domain generalization, and multi-source domain adaptation tasks. Generally, our algorithm is impressively simple and efficient, making it a strong baseline for domain adaptation and generalization tasks.

## 2. Related Work

The last decade has witnessed a boom in studies towards domain adaptation and related applications. We refer the interested reader to [66, 9] for a survey focusing on computer vision applications. As stated above, both feature transformation [51, 41, 74, 36] and feature representation learning [19, 53, 63, 68, 53] are much more favored by recent domain adaptation approaches. Here we analyze several most closely related work from both cases to our method.

Regarding shallow feature transformation based approaches, [36] proposes a general objective in order to pursue that instances from the same class of both domains are dragged closer to each other, which includes many former methods [51, 41] as special cases. Our method can be considered to be built upon [36] by developing a built-in classifier to infer the pseudo target labels, and it only acquires distant supervisions, *i.e.*, class-wise Gaussian estimators of means and covariance matrices, instead of accessing the entire source domain data. Besides, previous methods always involve several large MMD matrices [41, 74, 36, 67] in the optimization procedure with a $\mathcal{O}(n^2)$ computation complexity, making it unsuitable for large-scale adaptation scenarios. In fact, there are several previous studies [11, 10] that attempt to investigate Nearest Class Means (NCM) for domain adaptation. However, they merely integrate NCM with other domain adaptation models [61, 5] as a novel classifier, ignoring the distribution discrepancy after projection.

Benefiting from the rapid development of deep neural networks, deep domain adaptation methods achieve much better performance. Generally, a majority of them (e.g., [65, 17, 60, 64, 68]) equip the source domain a source classification loss function and design another cross-domain loss function (e.g., MMD or adversarial loss in Generative Adversarial Network (GAN) [22]) to align two domains. SimNet [53] replaces the conventional source classifier (e.g., soft-max) with a prototype similarity-based classifier and adopts a domain discriminator for domain confusion, which achieves state-of-the-art performances. Furthermore, [19] utilizes a reconstruction loss for target domain and [53] designed a similarity-based classifier for labeled source domain. However, these methods are always optimized in a batch-wise manner, making it not suitable to minimize a global loss function like MMD. Our approach, applied to learned transferable features extracted from fine-tuned models and deep domain adaptation methods like DAN [39], RevGrad [17] and GTA [56], achieves better performance to these more complex methods and is expected to be incorporated directly into the network structure.

## 3. Methodology

### 3.1. Problem Definition

In conventional domain adaptation with two domains, we have access to labeled images $X_s = \{(x_i^s, y_i^s)\}_{i=1}^{n_s}$ drawn from a source domain distribution $p_s(x, y)$ and target images $X_t = \{(x_i^t, y_i^t)\}_{i=1}^{n_t}$ drawn from a target domain distribution $p_t(x, y)$. It is commonly assumed that the label space of both domains are identical, i.e., $y_i^s, y_i^t \in [1, 2, \cdots, C]$ and the length of domain feature space is the same, i.e., $x_i^s, x_i^t \in \mathbf{R}^d$. Note that, in the unsupervised setting, we have no information about the labels on the target domain. Without loss of generality, we assume that each feature vector is normalized to satisfy $\|x_i^q\|^2 = 1, q \in \{s, t\}$, then zero-centered, i.e., $\sum_i x_i^s = \mathbf{0}$ and $\sum_i x_i^t = \mathbf{0}$.

In fact, instead of using the original data in the source domain, we only require the number of samples in each class (i.e., $m_r$), the maximum likelihood estimators of the means $\hat{\mu}_r$, and covariance matrices $\hat{\Sigma}_r$, $r = [1, 2, \cdots, C]$ if we assume the data of the $r$-th class in the source domain follow a $d$-variate Gaussian distribution. Obviously, these two estimators are defined as $\hat{\mu}_r = \frac{1}{m_r}\sum_{y_i^s=r} x_i^s$ and $\hat{\Sigma}_r = \frac{1}{m_r}\sum_{y_i^s=r}(x_i^s - \hat{\mu}_r)(x_i^s - \hat{\mu}_r)^T$, respectively.

### 3.2. Formulation

To tackle the covariate shift, we propose a shallow feature transformation based domain adaptation method. Our method consists of two main components, *i.e.*, source domain classification and target domain classification, and relate these two parametric classifiers by assuming small perturbations on parameters [71]. Since we have no access to the labels on the target domain, it is hard to directly consider a target domain classification task as well as some conditional distribution discrepancies. To address this issue, JDA [41] exploits the pseudo target labels as supervision signals for feature transformation learning. This strategy has proven to work well for unsupervised domain adaptation and has been re-utilized by later works [74, 36]. Thus, we also consider such a discriminative component to deal with pseudo-labeled target instances.

Inspired by the popular supervised adaptive feature reduction method [12], we first introduce a feature transformation matrix $W \in \mathbf{R}^{k\times d}$ for the labeled source data points and expect the following objective with respect to $W$ to be minimized as much as possible.

$$\min_W \frac{\sum_i d_W(x_i^s, \hat{\mu}_{y_i^s})}{\sum_i d_W(x_i^s, \hat{\mu})} = \frac{\sum_r \sum_{y_i^s=r} d_W(x_i^s, \hat{\mu}_r)}{\sum_i d_W(x_i^s, \mathbf{0})}, \quad (1)$$

where $d_W(x, x') = \|Wx - Wx'\|_2^2$, and $\hat{\mu}^s = \mathbf{0}$ denotes the overall mean in the source domain. Furthermore, we can rewrite the objective above as

$$\min_W \frac{\text{trace}(WS_w^sW^T)}{\text{trace}(WS_t^sW^T)}, \quad (2)$$

where $S_w^s = \sum_i(x_i^s - \hat{\mu}_{y_i^s})(x_i^s - \hat{\mu}_{y_i^s})^T = \sum_{r=1}^C m_r\hat{\Sigma}_r$ and $S_t^s = \sum_i(x_i^s)(x_i^s)^T = S_w^s + \sum_r m_r(\hat{\mu}_r - \mathbf{0})(\hat{\mu}_r - \mathbf{0})^T$ are also well known as the within-class scatter and the total scatter matrices in the source domain, respectively.

The only information we know about the target domain is that it shares the same classes with the source domain, therefore, we also expect that data points from the target domain are well separated as those from the source domain,

$$\min_{W, \hat{\mu}_r^t, \hat{y}_i^t} \frac{\sum_r \sum_{\hat{y}_i^t=r} d_W(x_i^t, \hat{\mu}_r^t)}{\sum_i d_W(x_i^t, \mathbf{0})}. \quad (3)$$

Since no labels are available for the target domain, we also need to estimate the optimal pseudo label $\hat{y}_i^t$ for each target instance and the target means $\hat{\mu}_r^t, 1 \le r \le C$.

For simplicity, we follow the popular framework to learn a unique projection $W$ for two different domains. Different from previous works [41, 74, 36] that generate pseudo target labels via merely exploiting the source data points, we aim to obtain a self-training target classifier and assume that the parameters are shifted from the parameters in the source classifier [71]. Concretely, we adopt the nearest centroid classifier and force the target centroids are close to their corresponding source centroids in Fig. 1, that is to say, we want to minimize the following objective function:

$$\begin{aligned}\min_{W, \Delta_r^t, \hat{y}_i^t} & \frac{\sum_r \sum_{y_i^s=r} d_W(x_i^s, \hat{\mu}_r)}{\sum_i d_W(x_i^s, \mathbf{0})} + \sum_r \beta_r\|W\Delta_r\|_2^2 \\ & + \frac{\sum_r \sum_{\hat{y}_i^t=r} d_W(x_i^t, \hat{\mu}_r + \Delta_r)}{\sum_i d_W(x_i^t, \mathbf{0})},\end{aligned} \quad (4)$$

where $\Delta_r \in \mathbf{R}^{d\times 1}, 1 \le r \le C$ are the the so-called perturbation variables and $\beta_r$ are the trade-off parameters which rely on the cluster size of each class.

For the sake of simple optimization, we reformulate these three terms in one trace-ratio objective and obtain a relaxed ratio-trace objective in the following,

$$\min_{W, \Delta_r^t, \hat{y}_i^t} \frac{\text{trace}(W(S_w^s + S_w^t + \sum_r \beta_r\Delta_r\Delta_r^T + \lambda I)W^T)}{\text{trace}(W(S_t^s + S_t^t)W^T)}, \quad (5)$$

$$\hookrightarrow \min_{W, \Delta_r^t, \hat{y}_i^t} \text{trace}\{\frac{W(S_w^s + S_w^t + \sum_r \beta_r\Delta_r\Delta_r^T + \lambda I)W^T}{W(S_t^s + S_t^t)W^T}\}, \quad (6)$$

$$s.t.\ W(S_t^s + S_t^t)W^T = I,$$

where $S_w^t = \sum_i(x_i^t - \hat{\mu}_{\hat{y}_i^t} - \Delta_{\hat{y}_i^t})(x_i^t - \hat{\mu}_{\hat{y}_i^t} - \Delta_{\hat{y}_i^t})^T$ and $S_t^t = \sum_i(x_i^t)(x_i^t)^T$ are the corresponding scatter variances in the target domain. In fact, to avoid a solution where all the rows in $W$ are identical [46], we impose an orthonormal constraint $W(S_t^s + S_t^t)W^T = I$ on the projection $W$. Under this constraint, Fukunaga [16] showed that Eq. (5) is essentially identical to Eq. (6). Besides, we follow previous works [41, 74, 36, 51] in this literature and impose a regularization parameter $\lambda$ to guarantee the optimization problem to be well-defined.

### 3.3. Optimization

To optimize the above objective in Eq. (6), we exploit a popular alternating optimization scheme. In the following, we provide the solutions to each sub-problem.

$W$**-step**: Once we fix these two other groups of variables $\{\Delta_r^t\}_{r=1}^C$ and $\{\hat{y}_i^t\}_{i=1}^{n_t}$, the objective function w.r.t. $W$ becomes a classical ratio-trace optimization problem:

$$\min_{W\in\mathbf{R}^{k\times d}} \text{trace}\{(WS_tW^T)^{-1}(W(S_w+\lambda I)W^T)\}, \tag{7}$$

where $S_t = S_t^s + S_t^t$ and $S_w = S_w^s + S_w^t + \sum_r \beta_r \Delta_r \Delta_r^T$ are two $d \times d$ scatter variance matrices. Interestingly, this problem can be efficiently solved by generalized eigenvalue decomposition (GEVD) $S_t w_a = \gamma_a (S_w + \lambda I) w_a$, where $\gamma_a$ is the $a$-th smallest generalized eigenvalue. The matrix $W$ is then constituted of the corresponding eigenvectors $w_a \in \mathbf{R}^{d\times 1}, 1 \le a \le k$ as rows.

$\hat{Y}^t$**-step**: Once we obtain the domain-invariant projection $W$ and perturbation variables $\{\Delta_r^t\}_{r=1}^C$, the objective function w.r.t. $\{\hat{y}_i^t\}_{i=1}^{n_t}$ has the form

$$\begin{aligned}
&\min_{\{\hat{y}_i^t\}_{i=1}^{n_t}} \text{trace}\{(WS_tW^T)^{-1}(WS_wW^T)\}\\
&= \sum_{i=1}^{n_t} \min_{\hat{y}_i^t\in[1,C]} \{(x_i^t - \hat{\mu}_{\hat{y}_i^t} - \Delta_{\hat{y}_i^t})^T S_p (x_i^t - \hat{\mu}_{\hat{y}_i^t} - \Delta_{\hat{y}_i^t})\},\\
&= \sum_{i=1}^{n_t} \{(x_i^t)^T S_p (x_i^t) + \max_{\hat{y}_i^t\in[1,C]} (h_{\hat{y}_i^t} x_i^t - b_{\hat{y}_i^t})\},
\end{aligned} \tag{8}$$

where $S_p = W^T(WS_tW^T)^{-1}W$ is a positive definite matrix, and the parameters in the classification function $f_r(x_i^t) = h_r x_i^t - b_r$ of the $r$-th class are $h_r = 2(\hat{\mu}_r + \Delta_r)S_p \in \mathbf{R}^{1\times d}$ and $b_r = -(\hat{\mu}_r + \Delta_r)^T S_p (\hat{\mu}_r + \Delta_r)$, respectively. That is to say, we can estimate each the pseudo target label $\hat{y}_i^t$ independently via the following rule

$$\hat{y}_i^t = \arg\max_{r\in[1,C]} h_r x_i^t - b_r. \tag{9}$$

$\Delta$**-step**: Once we get the domain-invariant projection $W$ and pseudo target labels $\{\hat{y}_i^t\}_{i=1}^{n_t}$, the objective function w.r.t. each perturbation variable $\Delta_r$ can be written as

$$\begin{aligned}
&\min_{\Delta_r} \text{trace}\{S_p(\sum_{\hat{y}_i^t=r}(x_i^t - \hat{\mu}_r - \Delta_r)(x_i^t - \hat{\mu}_r - \Delta_r)^T + \beta_r\Delta_r\Delta_r^T)\},\\
&\Rightarrow \min_{\Delta_r} \text{trace}\{S_p((\beta_r + n_r)\Delta_r\Delta_r^T - 2\sum_{\hat{y}_i^t=r}(x_i^t - \hat{\mu}_r)\Delta_r^T)\},\\
&\Rightarrow \min_{\Delta_r} (\Delta_r - \frac{\sum_{\hat{y}_i^t=r} x_i^t - n_r\hat{\mu}_r}{\beta_r + n_r})^T S_p (\Delta_r - \frac{\sum_{\hat{y}_i^t=r} x_i^t - n_r\hat{\mu}_r}{\beta_r + n_r}),
\end{aligned} \tag{10}$$

where $n_r$ is the size of the $r$-th class in the target domain. Since $S_p$ is easily proven to be a positive definite matrix, *i.e.*, $\forall x \in \mathbf{R}^{d\times 1}, x^T S_p x \ge 0$. Thus, the optimal variable $\Delta_r$ can be obtained via the following equation

$$\Delta_r = (\sum\nolimits_{\hat{y}_i^t=r} x_i^t - n_r\hat{\mu}_r)/(\beta_r + n_r), r \in [1, C], \tag{11}$$

$$\text{and} \quad \hat{\mu}_r + \Delta_r = (\beta_r\hat{\mu}_r + \sum\nolimits_{\hat{y}_i^t=r} x_i^t)/(\beta_r + n_r). \tag{12}$$

Carefully checking the term above, we can easily discover that the learned perturbations indeed place the optimal target centroids/ prototypes in the routines between source class means and pseudo target class means. Besides, when the value $\beta_r/n_r$ becomes larger, the learned target centroids are much closer to their corresponding source class means.

Towards this end, we have provided three closed-form solutions in Eq. (7), Eq. (9), and Eq. (11) for each subproblem, and the complete algorithm is summarized in Algorithm. 1.

**Algorithm 1** Unsupervised Domain Adaptation with Minimum Centroid Shift (MCS)

```
Input: Source domain information $\{m_r, \hat{\mu}_r, \hat{\Sigma}_r\}_{r=1}^C$ and target
    domain $\{x_i^t\}_{i=1}^{n_t}$; subspace dimensionality $k$, parameters $\lambda$
    and $\alpha$, inner/ outer maximum iterations $T_i$ = 5 and $T_o$ = 10.
Output: Feature transformation matrix $W \in \mathbf{R}^{k\times d}$, perturbation
    variables $\{\Delta_r\}_{r=1}^C$ and target labels $\{\hat{y}_i^t\}_{i=1}^{n_t}$.
 1: Compute the scatter matrices $S_w^s, S_t^s, S_t^t$;
 2: Initialize $S_w^t$ and $\Delta_r$ as 0, and calculate $W$ via Eq. (7)
 3: Estimate $\{\hat{y}_i^t\}_{i=1}^{n_t}$ using $W$ and $\{\hat{\mu}_r\}_{r=1}^C$ and update $S_w^t$;
 4: Compute the size of the $r$-th target class $l_r$ and let $\beta_r = \alpha * l_r$;
 5: while not converge iter $\le T_o$ do
 6:     while not converge and iter $\le T_i$ do
 7:         Update perturbation variables $\{\Delta_r\}_{r=1}^C$ via Eq. (11);
 8:         Update pseudo target labels $\{\hat{y}_i^t\}_{i=1}^{n_t}$ via Eq. (9);
 9:     end while
10:     Update $S_w^t$ and solve the GEVD problem in Eq. (7) for $W$.
11: end while
```

### 3.4. Convergence and Time Complexity

It is obvious that each solution above (*i.e.*, Eqs. (7), (9), (11)) monotonically decreases the overall objective function $\text{trace}\{(WS_tW^T)^{-1}(W(S_w + \lambda I)W^T)\}$ in each iteration and the objective value is always larger than zero, that is to say, the proposed iterative algorithm in Algorithm. 1 converges to the local minimizer after certain iterations.

Regarding the computation complexity, the GEVD step occupies $\mathcal{O}(kd^2)$, other matrix multiplies occupy $\mathcal{O}(n_td^2 + n_tdk)$, and the $\hat{Y}^t$ step occupies $\mathcal{O}(Cn_tk^2 + Cn_td + k^3)$. In summary, the overall complexity of our method is $\mathcal{O}(T_ikd^2 + T_iT_oCn_td)$. When compared with massive previous methods [41, 74, 36, 67] with $\mathcal{O}(n^2)$ complexity, our method is more favorable by large-scale datasets. *Specifically, for large-scale digit datasets like SVHN and MNIST, previous methods (eg., [41, 74, 36, 67]) are not flexible due to the limited memory.*

## 4. Experiment

In this section, we conduct extensive experiments to evaluate the effectiveness of the proposed approaches for unsupervised cross-domain image recognition problems, including vanilla domain adaptation, multi-source domain adaptation and domain generalization where the target domain is totally unknown in the training phase.

### 4.1. Unsupervised Domain Adaptation (UDA)

#### 4.1.1 Datasets and Settings

The **Office31** [55] dataset includes images of 31 objects taken from 3 domains, i.e., Amazon (**A**, images downloaded from the online web merchants), DSLR (**D**, high-resolution images captured by a digital SLR camera), and Webcam (**W**, low-resolution images recorded by a web camera). They contain 2,817, 498 and 795 images, respectively.

The **Office-Caltech** dataset consists of images from 10 overlapping object classes between Office31 and Caltech256 [25], and these domains include 958, 157, 295 and 1,123 samples for **A**, **D**, **W** and Caltech (**C**), respectively.

The **Office-Home** [66] dataset is a new benchmark that contains 4 domains, with each domain containing 65 kinds of everyday objects, i.e., Art (**Ar**, artistic depictions of objects), Clipart (**Cl**, clipart images), Product (**Pr**, objects without a background) and Real-World (**Re**, objects captured with a regular camera). Besides, they contain 2,421, 4,365, 4,428 and 4,357 samples, respectively.

**Baseline methods.** We compare the proposed method with several state-of-the-art unsupervised domain adaptation approaches that can be roughly divided into two main categories, shallow feature transformation approaches, and deep domain adaptation approaches. 1NN is a basic method that is trained on the raw source data without any feature transformation. Shallow UDA methods mainly include SA [15], JDA [41], CORAL [59], Invariant Latent Space (ILS) [27], JGSA [74], LDA-inspired Domain Adaptation (LDADA) [45], and DICE [36]. There are also some popular baseline methods, including ATI [4], PUnDA [20], MEDA [67], and GAKT [13].

Regarding deep end-to-end UDA approaches, we collect the reported accuracies from recent studies that share the same protocol with our method. Some representative deep methods are listed in the following, DAN [39], RevGrad [17], DRCN [19], RTN-res [43], ADDA [64], JAN-A [44], GTA [56], and CDAN+E [40].

Note that A→B indicates that A is the source domain and B is the target domain. We evaluate different UDA methods in terms of classification accuracy (%) on the target.

#### 4.1.2 Results on the Office31 dataset

We follow the full protocol that has been widely adopted in previous studies [44, 36, 56] by using the entire labeled data in the source domain and unlabeled data in the target domain. We further exploit the AlexNet-FC$_7$ features [59] fine-tuned on the source domain, making it fair to be compared with deep UDA methods via AlexNet.

Table 1. Accuracy (%) on Office31 with the evaluation setup of [60, 36]. The best results of methods with deep features are in **bold red** and deep models with **bold underlined** results are better than MCS (ours). [* using ResNet-34]

| Method | A→D | A→W | D→A | D→W | W→A | W→D | Avg. |
|---|---|---|---|---|---|---|---|
| 1NN | 59.4 | 57.5 | 47.2 | 96.1 | 44.8 | 99.0 | 67.3 |
| SA | 61.0 | 59.5 | 46.9 | 95.1 | 46.6 | 98.2 | 67.9 |
| JDA | 66.5 | 68.8 | 56.3 | 97.7 | 53.5 | 99.6 | 73.7 |
| CORAL | 60.4 | 57.0 | 47.6 | 96.2 | 46.3 | 99.0 | 67.8 |
| JGSA | 67.5 | 62.3 | 55.6 | **98.1** | 52.0 | **99.8** | 72.5 |
| ILS | 62.9 | 63.9 | 50.0 | 97.2 | 48.8 | 99.4 | 70.4 |
| ATI [4] | 70.3 | 68.7 | 55.3 | 95.0 | 56.9 | 98.7 | 74.2 |
| LDADA | 65.9 | 68.1 | 55.5 | 94.7 | 53.4 | 98.4 | 72.6 |
| DICE | 66.7 | 71.4 | 56.5 | 96.9 | **58.6** | **99.8** | 75.0 |
| MEDA [67] | 69.5 | 69.9 | 58.0 | 94.0 | 56.0 | 96.8 | 74.0 |
| MCS(ours) | **71.9** | **75.1** | **58.8** | 96.7 | 57.2 | 99.4 | **76.5** |
| RevGrad | 72.3 | 73.0 | 53.4 | 96.4 | 51.2 | 99.2 | 74.3 |
| DAN [39] | 67.0 | 68.5 | 54.0 | 96.0 | 53.1 | 99.0 | 72.9 |
| DRCN [19] | 66.8 | 68.7 | 56.0 | 96.4 | 54.9 | 99.0 | 73.6 |
| RTN-res [43] | 71.0 | 73.3 | 50.5 | **96.8** | 51.0 | **99.6** | 73.7 |
| ADDA | 71.6 | 73.5 | 54.6 | 96.2 | 53.5 | 98.8 | 74.0 |
| JAN-A [44] | **72.8** | **75.2** | 57.5 | 96.6 | 56.3 | **99.6** | 76.3 |
| I2I-Adapt [49]* | 71.1 | **75.3** | 50.1 | 96.5 | 52.1 | **99.6** | 74.1 |

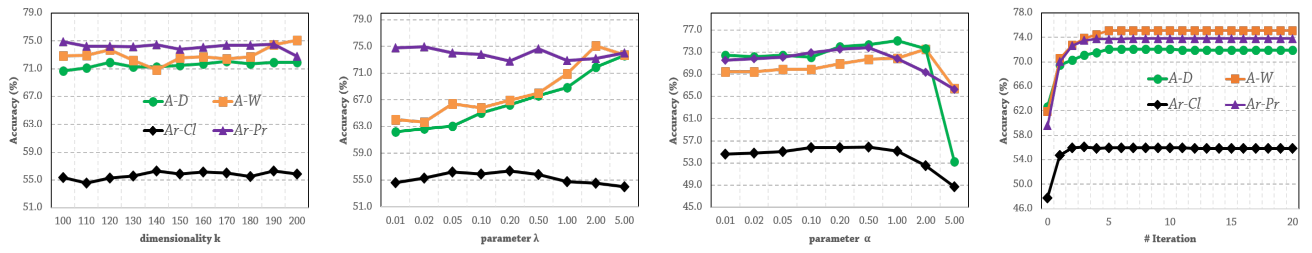
As shown in Table 1, our method outperforms all deep methods and other shallow counterparts in terms of the average accuracy. Firstly, for the small and easy tasks D↔W, all the UDA methods achieve promising results, and our method performs worse than several shallow methods (e.g., JGSA [74]). Secondly, our method significantly outperforms all the shallow methods and is competitive to state-of-the-art deep method JAN-A [44] for some relatively challenging adaptation tasks, including A→D and A→W. Thirdly, when a small source domain (i.e., D and W) is adapted to a large target domain A, our method obviously ranks the first and second among all the methods.

Generally speaking, our method outperforms several state-of-the-art shallow methods (*i.e.*, DICE [36] and MEDA [67]) by a large margin and achieves quite competitive results to the state-of-the-art deep UDA method. Note that, even I2I-Adapt [49] explores a more powerful ResNet-34 model, our method still performs much better than it.

#### 4.1.3 Results on the Office-Caltech dataset

We explore two kinds of deep features [27] produced by different full convolution layers in VGG-net and follow the sampling protocol [21, 27, 20] by using few labeled data in the source domain. Concretely, we randomly select 8 instances per class for domain D and 20 instances per class for other domains (i.e., A, C and W) as final sources.

As can be seen from Table 2, JGSA [74], PUnDA [20] and DICE [36] are three best performing methods among



(a) Dimensionality $k$ (b) Parmeter $\lambda$ (c) Parmeter $\alpha$ (d) # Iterations

Figure 2. Domain adaptation performance accuracy (%) on the Office31 (A→D and A→W) and Office-Home (Ar→Cl and Ar→Pr) datasets w.r.t. dimensionality $k$, parameters $\lambda$, $\alpha$ and the number of iterations.

Table 2. Accuracy (%) on Office-Caltech using the VGG-FC$_6$ (above) and VGG-FC$_7$ (below) features with the evaluation setup of [21, 20]. The best (**bold red**), the second best (red).

| Source | A | | | C | | | D | | | W | | | Avg. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Target | C | D | W | A | D | W | A | C | W | A | C | D | |
| 1NN | 70.1 | 52.3 | 60.9 | 81.9 | 55.6 | 65.9 | 57.0 | 48.0 | 86.7 | 66.4 | 60.2 | 91.3 | 66.4 |
| SA | 77.1 | 64.9 | 76.0 | 83.9 | 66.2 | 76.0 | 69.0 | 62.3 | 90.5 | 80.2 | 71.9 | 94.2 | 76.0 |
| JDA | 80.2 | 71.9 | 80.6 | 88.7 | 72.0 | 82.2 | 81.8 | 73.4 | 94.0 | 89.0 | 79.9 | 96.2 | 82.5 |
| CORAL | 79.0 | 67.1 | 74.8 | 89.4 | 67.6 | 77.6 | 75.8 | 64.7 | 94.6 | 82.3 | 75.9 | 96.0 | 78.7 |
| JGSA | 79.9 | 71.7 | 82.6 | 90.2 | 76.4 | 84.5 | 82.7 | 73.5 | 95.4 | **91.6** | 79.0 | **96.3** | 83.6 |
| ILS | 78.9 | 72.5 | 82.4 | 87.6 | 73.0 | 84.4 | 79.2 | 66.5 | 94.2 | 87.2 | 79.9 | 89.3 | 81.3 |
| PUnDA | 82.3 | **76.2** | 82.7 | 90.3 | 76.2 | **88.3** | 83.1 | 69.2 | 93.4 | 86.9 | 82.6 | 89.8 | 83.4 |
| LDADA | 82.3 | 64.0 | 80.3 | 89.7 | 67.7 | 82.2 | 70.9 | 60.6 | 86.9 | 90.2 | 82.5 | 87.8 | 78.8 |
| DICE | 83.0 | 66.4 | 75.9 | 91.9 | 67.4 | 83.7 | 84.4 | **78.6** | 94.8 | 90.3 | 80.7 | 93.8 | 82.6 |
| MCS(ours) | **87.1** | 74.8 | **84.8** | **92.3** | **77.3** | 87.1 | **84.7** | 76.0 | **95.9** | 88.9 | **87.4** | 92.9 | **85.8** |
| 1NN | 72.6 | 50.8 | 64.0 | 82.6 | 54.9 | 65.3 | 61.2 | 52.8 | 88.2 | 67.8 | 64.2 | 88.8 | 67.8 |
| SA | 76.2 | 60.7 | 75.0 | 82.6 | 63.2 | 73.6 | 66.0 | 59.4 | 89.5 | 76.4 | 69.0 | 94.0 | 73.8 |
| JDA | 79.9 | 69.2 | 80.1 | 87.3 | 71.5 | 80.1 | 78.5 | 70.9 | 92.4 | 86.9 | 78.3 | 94.1 | 80.8 |
| CORAL | 78.6 | 61.3 | 71.8 | 88.6 | 63.8 | 76.0 | 71.2 | 63.0 | 93.5 | 82.0 | 73.7 | 94.6 | 76.5 |
| JGSA | 81.1 | 72.3 | 81.4 | 88.3 | 72.3 | 82.5 | 78.9 | 72.3 | 93.6 | 89.8 | 79.8 | **95.8** | 82.3 |
| ILS | 78.4 | 71.3 | 80.9 | 87.1 | 67.1 | 80.1 | 76.5 | 66.2 | 91.8 | 86.7 | 76.3 | 88.2 | 79.2 |
| PUnDA | 81.0 | **75.8** | 81.4 | 91.1 | 70.8 | 83.8 | 80.4 | 69.1 | 92.0 | 85.7 | 80.1 | 90.1 | 81.7 |
| LDADA | 83.3 | 72.5 | 83.7 | 91.5 | 71.5 | 84.5 | 71.8 | 58.4 | 88.3 | 88.0 | 80.1 | 86.8 | 80.0 |
| DICE | 83.7 | 62.9 | 79.3 | 91.7 | 63.8 | 84.3 | 82.3 | 76.4 | 94.2 | 89.4 | 82.1 | 91.0 | 81.7 |
| MCS(ours) | **86.3** | 72.8 | **86.6** | **92.8** | **73.0** | **89.3** | **84.6** | **76.5** | **95.5** | **90.4** | **85.6** | 88.9 | **85.2** |

the shallow UDA approaches for both kinds of features. Comparing them with our method, we can easily find that MCS always beats them by a large margin in terms of the average accuracy. Specifically, MCS ranks the first or second in 10 and 11 out of 12 tasks for VGG-FC$_6$ and VGG-FC$_7$ features, respectively.

Note that, most previous UDA methods except LDADA [45] favor VGG-FC$_6$ features because they may be not much more discriminative than VGG-FC$_7$ features, making them suitable for general feature transformation based approaches. By contrast, MCS is somewhat robust to the feature type, since the dropping rate of accuracy from VGG-FC$_6$ to VGG-FC$_7$ is relatively small.

### 4.1.4 Results on the Office-Home dataset

For the Office-Home dataset, we explore PyTorch to fine-tune the ResNet-50 model [26] pre-trained on the ImageNet dataset with the labeled source images and extract the features after the 5-th pooling layer for each task.

LDADA [45] performs quite worse with a 2.6 average accuracy, thus, we do not report its results in Table 3. Obviously, MCS again performs the best among all the methods while CDAN+E [40] and DICE [36] achieves the second

Table 3. Accuracy (%) on Office-Home with ResNet-50 features and the evaluation setup of [66, 36]. [* using VGG-FC features]

| Source | Ar | | | Cl | | | Pr | | | Re | | | Avg. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Target | Cl | Pr | Re | Ar | Pr | Re | Ar | Cl | Re | Ar | Cl | Pr | |
| 1NN | 44.1 | 61.8 | 69.2 | 49.6 | 60.6 | 63.3 | 51.6 | 43.1 | 70.6 | 63.1 | 48.9 | 76.2 | 58.5 |
| SA | 44.8 | 65.5 | 70.6 | 48.8 | 61.5 | 64.1 | 50.1 | 42.8 | 71.1 | 62.2 | 48.1 | 76.2 | 58.8 |
| JDA | 46.3 | 66.0 | 69.1 | 47.1 | 63.4 | 63.3 | 48.2 | 44.0 | 70.8 | 60.1 | 49.6 | 76.8 | 58.7 |
| CORAL | 47.1 | 67.3 | 74.8 | 52.3 | 63.6 | 66.9 | 51.0 | 41.9 | 72.6 | 62.8 | 46.8 | 77.7 | 60.4 |
| JGSA | 50.3 | 70.0 | 73.8 | 52.7 | 68.9 | 68.2 | 55.6 | 47.9 | 75.1 | 64.0 | 52.0 | 78.7 | 63.1 |
| ILS | 46.7 | 64.3 | 69.7 | 44.3 | 60.9 | 62.6 | 47.9 | 42.7 | 70.4 | 61.4 | 48.5 | 75.8 | 57.9 |
| DICE | 53.2 | 72.4 | 74.5 | 56.5 | **70.1** | 69.1 | **58.9** | **51.5** | 77.0 | 66.5 | 54.8 | 79.0 | 65.3 |
| GAKT [13]* | 34.5 | 43.6 | 55.3 | 36.1 | 52.7 | 53.2 | 31.6 | 40.6 | 61.4 | 45.6 | 44.6 | 64.9 | 47.0 |
| CDAN [40] | 49.0 | 69.3 | 74.5 | 54.4 | 66 | 68.4 | 55.6 | 48.3 | 75.9 | 68.4 | 55.4 | 80.5 | 63.8 |
| CDAN+E [40] | 50.7 | 70.6 | 76.0 | **57.6** | 70.0 | 70.0 | 57.4 | 50.9 | 77.3 | **70.9** | **56.7** | 81.6 | 65.8 |
| MCS(ours) | **55.9** | **73.8** | **79.0** | 57.5 | 69.9 | **71.3** | 58.4 | 50.3 | **78.2** | 65.9 | 53.2 | **82.2** | **66.3** |

and third best performance. Carefully comparing our MCS with CDAN+E and DICE, we find that MCS consistently performs better than CDAN+E and DICE in 7 out of 12 tasks. Checking the results of MCS again, we find that the results are somewhat low when **Cl** is the source domain. This may be because our method adopts the nearest centroid classifier which may ignore the diversity of each class in the source domain like the 'Clipart' subset.

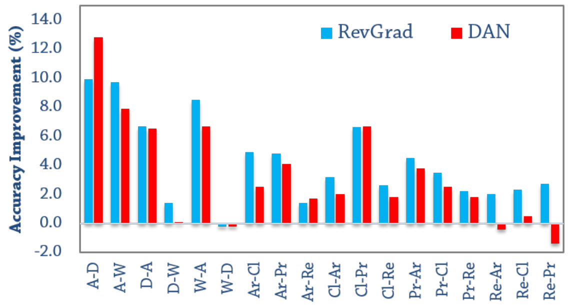


Figure 3. Accuracy improvement (%) of RevGrad [17] and DAN [39] when integrated with MCS(ours) on Office31 and Office-Home via ResNet-50. (A-D: A is Source, D is Target)

### 4.1.5 Parameter Sensitivity Analysis

To investigate the sensitivity of dimensionality $k$, parameters $\lambda$ and $\alpha$ in our method, we exploit four UDA tasks on the Office31 and Office-Home datasets, *i.e.*, A→D, A→W, Ar→Cl, and Ar→Pr. We respectively display all these parameter sensitivity analysis results in Figures 2(a)∼2(c), with a wide range of $d \in [100, 110, \cdots, 200]$ and $\lambda, \alpha \in \{0.01, 0.02, 0.05, 0.1, 0.2, 0.5, 1, 2, 5\}$.

Table 4. Domain generalization performance accuracy (%) on the VLCS dataset with the evaluation setup of CIDG [35].

| Source | Target | 1NN | KPCA | DICA | Undo-bias | SCA | CIDG | MCS$_A$(ours) | MCS$_B$(ours) |
|---|---|---|---|---|---|---|---|---|---|
| L,C,S | V | 53.27 ± 1.52 | 58.62 ± 1.44 | 58.29 ± 1.51 | 57.73 ± 1.02 | 57.48 ± 1.78 | 65.65 ± 0.52 | **65.74 ± 1.26** | 65.17 ± 0.44 |
| V,C,S | L | 50.35 ± 0.94 | 53.80 ± 1.78 | 50.35 ± 1.45 | 58.16 ± 2.13 | 52.07 ± 0.86 | **60.43 ± 1.57** | 57.42 ± 0.48 | 59.95 ± 0.40 |
| V,L,S | C | 76.82 ± 1.56 | 85.84 ± 1.64 | 73.32 ± 4.13 | 82.18 ± 1.77 | 70.39 ± 1.42 | 91.12 ± 1.62 | **92.40 ± 0.62** | 89.30 ± 0.37 |
| V,L,C | S | 51.78 ± 2.07 | 53.23 ± 0.62 | 54.97 ± 0.61 | 55.02 ± 2.53 | 54.46 ± 2.71 | 60.85 ± 1.05 | 62.07 ± 0.82 | **65.16 ± 0.68** |
| C,S | V,L | 52.44 ± 1.87 | 55.74 ± 1.01 | 53.76 ± 0.96 | 56.83 ± 0.67 | 56.05 ± 0.98 | 59.25 ± 1.21 | 58.24 ± 0.44 | **60.42 ± 0.40** |
| L,S | V,C | 57.09 ± 1.43 | 58.50 ± 3.84 | 44.09 ± 0.58 | 51.16 ± 3.52 | 49.98 ± 1.84 | 55.65 ± 3.57 | **68.46 ± 2.66** | 67.19 ± 0.74 |
| L,C | V,S | 45.04 ± 2.49 | 45.13 ± 3.01 | 44.81 ± 1.62 | 52.16 ± 0.80 | 48.97 ± 1.04 | 54.04 ± 0.91 | 57.77 ± 1.29 | **59.73 ± 1.23** |
| V,S | L,C | 58.39 ± 0.78 | 64.56 ± 0.99 | 60.68 ± 1.36 | 68.58 ± 1.62 | 63.29 ± 1.34 | **70.44 ± 1.43** | 70.02 ± 0.32 | 70.43 ± 0.38 |
| V,C | L,S | 47.09 ± 2.49 | 55.79 ± 1.57 | 49.81 ± 1.40 | 59.00 ± 2.49 | 53.47 ± 0.71 | 61.61 ± 0.67 | **64.51 ± 0.54** | 61.46 ± 0.51 |
| V,L | C,S | 59.21 ± 1.84 | 63.88 ± 0.36 | 61.22 ± 0.95 | 64.26 ± 2.77 | 66.68 ± 1.09 | 70.89 ± 1.31 | 70.88 ± 0.51 | **72.12 ± 0.53** |

Table 5. Average accuracy (%) on the Office31 dataset via ResNet-50 with the evaluation setup of [44, 53, 40].

| Methods before 2018 | RevGrad [17] | DAN [39] | ADDA [64] | JAN-A [44] | RevGrad [17] +MCS(ours) | DAN [39] +MCS(ours) |
|---|---|---|---|---|---|---|
| | 81.8 | 81.7 | 82.9 | 85.3 | **87.8** | 87.4 |
| Methods after 2018 | GTA [56] | SimNet [53] | iCAN [75] | TEM [29] | CDAN [40] | CDAN+E [40] |
| | 86.5 | 86.2 | 87.2 | 87.2 | 86.6 | 87.7 |

It can be observed that MCS is robust with regard to different values of $k$ in Figure 2(a). For high-dimensional features via AlexNet and ResNet-50, $k = 150$ is an optimal choice. When $\lambda \to 0$, the optimization problem is ill-defined. When $\lambda \to \infty$, the minimum centroid shift is not performed, and MCS cannot construct robust representation for cross-domain classification. Concerning the sensitivity of regularization parameter $\lambda$, we plot the accuracies in Figure 2(b), which indicates that $\lambda \in$[1,2] is an optimal choice.

Theoretically, smaller values of $\alpha$ can make self-learning in the target domain (*i.e.*, using the class mean itself) more important in MCS. Observing the accuracy in Figure 2(c), we can discover that $\alpha \in$[0.1,1.0] is an optimal choice under which both the source centroids and the pseudo target class means are effectively considered. As expected, larger values of $\alpha$ can degenerate the adaptation performance due to the essential heterogeneity.

Finally, we plot the accuracy w.r.t. the number of iterations in Figure 2(d) where we consider the inner loop as well as the outer loop. Obviously, the accuracy always grows gradually until convergence within 4 outer iterations.

### 4.1.6 Combination with Deep UDA Methods

The comparisons with recent state-of-the-art shallow methods demonstrate the effectiveness of our method. However, we still doubt that whether deep adaptation methods can be enhanced with our simple method. Here we investigate this problem by exploiting two popular deep domain adaptation methods , *i.e.*, RevGrad [17] and DAN [39] for cross-domain object recognition and GTA [56] for cross-domain digit recognition. Regarding these deep UDA methods [17, 39, 56], we extract the intermediate features for both domains and take them as input for our method MCS.

As can be seen from Figure 3, once MCS is integrated with deep UDA methods, *i.e.*, RevGrad and DAN on the Office31 and Office-Home datasets, the results of most

Table 6. Accuracy (%) cross-domain recognition tasks on three digit based datasets (each domain using the entire training set [64, 56]). M: MNIST (60,000), U: USPS (7,291), S: SVHN (73,257).

| Method | M→U | U→M | S→M | Avg. |
|---|---|---|---|---|
| Source-only [56] | 84.6 | 68.9 | 60.9 | 71.5 |
| Source-only [56]+MCS(ours) | 91.5 | 94.5 | 82.3 | 88.4 |
| GTA [56] | 96.5 | 98.1 | 89.7 | 94.8 |
| GTA [56]+MCS(ours) | **97.8** | **98.2** | **91.7** | **95.9** |
| DRCN [19] | - | 73.7 | 82.0 | - |
| ADDA [64] | - | 90.1 | 76.0 | - |
| PixelDA [2] | 95.9 | - | - | - |
| SBADA-GAN [54] | 97.6 | 95.0 | 76.1 | 89.6 |
| S→M: LeNet (56.8), DICE$_{\text{MV-SVM}}$ [36] (80.9), MCS (ours, **82.6**) | | | | |

cases are improved in 17 and 15 out of 18 tasks, respectively. Checking the average accuracy (%) in Table 5, with the help of MCS, both the results of RevGrad and DAN are significantly improved, from 81.8 to 87.8 and 81.7 to 87.4. For generic cross-domain object recognition, we explore the pre-trained ResNet-50 model as our basic network and compare it with recent state-of-the-art methods [56, 53, 75, 29, 40] in Table 5. Specifically, RevGrad+MCS obtains the best average accuracy, which even beats the best result reported in CDAN+E [40]. It can be expected that MCS can achieve much higher results when combined with better adaptation methods like TEM.

Regarding the digit recognition task, we explore 3 popular digit datasets, *i.e.*, MNIST [33], USPS [28] and SVHN [50]. Specifically, we follow the standard protocol that uses the corresponding entire training sets as domains, and the testing sets for validation. All the images are rescaled to 32×32. In fact, we exploit the Source-only and GTA [56] models as a baseline network, and compare them with DRCN [19], PixelDA [2], ADDA [64], and SBADA-GAN[54] in Table 6. We easily observe that Source-only+MCS and GTA+MCS are much better than Source-only and GTA, respectively. Compared with recent state-of-the-art results [19, 2, 64, 54], GTA+MCS obtains the best results. Besides, MCS beats DICE$_{\text{MV-SVM}}$ with the features provide in [36]. Regarding the per-class classification accuracy (%) in M→U and U→M, MCS scores 97.5 and 98.2 that are higher than 96.4 and 95.6 in SimNet [53].

## 4.2. Domain Generalization

Table 7. Multi-source domain adaptation performance accuracy (%) on the PIE dataset with the evaluation setup of StP [37].

| Source | 1NN | StP | $MCS_C$ | $MCS_D$ | 1NN | StP | $MCS_C$ | $MCS_D$ | 1NN | StP | $MCS_C$ | $MCS_D$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| C05,C07 | Target=C09 | | | | Target=C27 | | | | Target=C29 | | | |
| | 45.4 | 53.1 | 63.8 | **72.8** | 60.7 | 43.8 | 90.7 | **92.9** | 31.2 | 60.0 | 49.6 | **61.8** |
| C05,C09 | Target=C07 | | | | Target=C27 | | | | Target=C29 | | | |
| | 45.2 | 44.9 | 66.4 | **70.3** | 59.1 | 57.4 | 90.5 | **90.7** | 36.9 | **67.5** | 55.7 | 63.1 |
| C05,C27 | Target=C07 | | | | Target=C09 | | | | Target=C29 | | | |
| | 69.1 | 58.3 | **82.9** | 81.1 | 71.3 | 45.4 | 73.0 | **74.6** | 45.6 | 58.2 | 58.8 | **69.3** |
| C05,C29 | Target=C07 | | | | Target=C09 | | | | Target=C27 | | | |
| | 38.5 | 66.8 | **67.3** | 63.9 | 35.8 | 71.6 | 71.0 | **75.9** | 50.3 | 71.0 | 89.1 | **89.3** |
| C07,C09 | Target=C05 | | | | Target=27 | | | | Target=C29 | | | |
| | 40.8 | 51.8 | 71.0 | **72.3** | 62.0 | 55.7 | 92.2 | **92.2** | 35.3 | 50.1 | **70.2** | 69.2 |
| C07,C27 | Target=C05 | | | | Target=C09 | | | | Target=C29 | | | |
| | 55.3 | 46.7 | **87.4** | 86.6 | 74.6 | 51.0 | 71.3 | **80.0** | 43.0 | 51.0 | 68.5 | **73.5** |
| C07,C29 | Target=C05 | | | | Target=C09 | | | | Target=C27 | | | |
| | 42.4 | 40.1 | 68.4 | **76.4** | 51.6 | 51.0 | 73.4 | **77.0** | 60.0 | 50.3 | 89.4 | **89.5** |
| C09,C27 | Target=C05 | | | | Target=C07 | | | | Target=C29 | | | |
| | 52.9 | 51.5 | **83.8** | 82.0 | 70.3 | 55.3 | 84.0 | **86.0** | 41.6 | 71.7 | 71.4 | **72.3** |
| C09,C29 | Target=C05 | | | | Target=C07 | | | | Target=C27 | | | |
| | 36.4 | 43.1 | 64.0 | **70.0** | 47.5 | 38.5 | **70.1** | 65.9 | 54.2 | 58.4 | 89.1 | **92.8** |
| C27,C29 | Target=C05 | | | | Target=C07 | | | | Target=C09 | | | |
| | 53.2 | 47.2 | 83.3 | **84.9** | 70.3 | 57.1 | **84.6** | 81.0 | 73.4 | 58.0 | 74.5 | **83.6** |
| Average : 1NN (51.8) StP (54.2) $MCS_C$ (ours, 75.2) $MCS_D$ (ours, **78.0**) | | | | | | | | | | | | |

Here we conduct experiments on a real world image classification dataset VLCS like [35], including four domains: VOC2007 (**V**), LabelMe (**L**), Caltech-101 (**C**) and SUN09 (**S**). These datasets share 5 object categories: bird, car, chair, dog, and person. The datasets from source domains are split into two parts: 70% for training and 30% for validation, following [35]. The whole target domain is used for testing. We repeat the random selection 10 times and report the mean classification accuracy and standard deviation.

For the domain generalization task, we compute all the total scatter matrices for each source domain and obtain $S_t$ by adding them up. Similarly, we compute all the within-class scatter matrices for each source domain and obtain $S_w$ via adding them up, where each $\Delta_r$ is a fixed variable by measuring the difference between different source class means. *To this end, we obtain the optimal projection matrix $W$ via Eq. (7). Note that there is no need to learn $\hat{Y}_t$ and $\Delta_r$, making this problem to be a one-pass algorithm. For MCS$_A$, each class has multiple centroids instead of one, then we estimate $\hat{Y}_t$ via Eq. (9). MCS$_B$ needs to know all the source instances instead of the estimated distributional parameters $\hat{\mu}_r, \hat{\Sigma}_r$, it follows DICE$_{SVM}$ [36] by training a linear SVM classifier on the projected source instances.*

We compare MCS$_A$ and MCS$_B$ with KPCA [57], DICA [48], Undo-bias [31], SCA [18] and CIDG [35]. As can be seen from Table 4, both MCS$_A$ and MCS$_B$ win 4 out of 10 tasks, while the best performing baseline CIDG merely win 2 out of 10 tasks. Besides, the standard deviations are much smaller than CIDG, which indicates that our methods are somewhat robust.

### 4.3. Multi-source Domain Adaptation

To address the multi-source domain adaptation task, we develop two different methods MCS$_C$ and MCS$_D$. Specifically, MCS$_C$ *naively combines multiple source domains into one source domain* and becomes a vanilla domain adaptation task. Similar to MCS$_A$, MCS$_D$ sums up all the within-class scatter matrices and total matrices and adopt them in Eq. (7), and *assumes the optimal centroid on the target to be close to the mean of different source class means*.

Table 8. Multi-source domain adaptation performance accuracy (%) on the Office31 and Office-Caltech datasets with the evaluation setups of DLD [47] and StP [37], respectively.

| Method | Dataset | A,D→W | A,W→D | D,W→A | Avg. | |
|---|---|---|---|---|---|---|
| DCTN [69] | Office31 | 96.9 | **99.6** | 54.9 | 83.8 | |
| DLD [47] | | 94.6 | 93.7 | 62.6 | 83.6 | |
| MsDA [47] | | 95.8 | 94.8 | **62.9** | 84.5 | |
| MCS$_C$(ours) | | 96.5 | 98.2 | 62.0 | 85.6 | |
| MCS$_D$(ours) | | **97.2** | 99.4 | 61.3 | **86.0** | |
| Method | Office-Caltech | A,C,D→W | A,C,W→D | A,D,W→C | C,D,W→A | Avg. |
| StP [37] | | 94.9 | 96.2 | 88.7 | **94.5** | 93.6 |
| MCS$_C$(ours) | | 97.6 | 96.8 | **89.2** | 93.5 | 94.3 |
| MCS$_D$(ours) | | **98.6** | **100.0** | 88.3 | 92.4 | **94.8** |

We utilize the PIE dataset that includes facial images of 68 people with various pose, illumination, and expression changes. Following [41, 37], we select 5 out of 13 poses, i.e., C05 (left), C07 (upward), C09 (downward), C27 (frontal) and C29 (right). These images are cropped to the size 32×32, constituting 1,024-dimensional features. We compare MCS$_C$ and MCS$_D$ with StP [37] in Table 7. Generally, MCS$_C$ and MCS$_D$ achieve the best results in 7 and 19 out of 27 tasks, respectively. They significantly outperform StP for almost all tasks except (C05, C09)→C29, and MCS$_D$ performs slightly better than MCS$_C$. This may be because that MCS$_C$ ignores the size of different source domains, which mainly relies on the larger source.

In addition, we consider the Office31 (AlexNet features [59]) and Office-Caltech (DeCAF features [74, 36]) datasets with the protocols in DLD [47] and StP [37], respectively. As can be seen from Table 8, we discover that MCS$_D$ is always superior to MCS$_C$ and both of them outperform other methods in terms of the average accuracy.

## 5. Conclusion

In this paper, we have proposed a simple, efficient, yet effective approach for visual domain adaptation. The key idea is to seek a subspace where the target centroids are moderately shifted from those in the source domain. Then a unified objective is designed to derive several sub-problems with closed-form solutions to subspace discovery and target pseudo-labeling, and the alternating minimization algorithm is guaranteed to converge. Note that, our method only acquires some class-wise distribution estimators from source data as distant supervisions, hence it also provides a privacy-preserving way for source domain data. Besides, it can be easily extended for domain generalization and multi-source domain adaptation problems. Extensive experiments on several visual benchmarks demonstrate the superiority of the proposed method over many existing state-of-the-art methods. Generally, our method is impressively simple and efficient, hence, it can be considered as a promising baseline for domain adaptation and generalization tasks.

## References

[1] S. Baker and M. Bsat. The cmu pose, illumination, and expression database. *IEEE Trans. Pattern Anal. Mach. Intell.*, 25(12):1615, 2003.

[2] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. In *Proc. CVPR*, pages 3722–3731, 2017.

[3] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan. Domain separation networks. In *Proc. NIPS*, pages 343–351, 2016.

[4] P. P. Busto and J. Gall. Open set domain adaptation. In *Proc. ICCV*, pages 754–763, 2017.

[5] M. Chen, Z. Xu, K. Q. Weinberger, and F. Sha. Marginalized denoising autoencoders for domain adaptation. In *Proc. ICML*, pages 1627–1634, 2012.

[6] B. Chidlovskii, S. Clinchant, and G. Csurka. Domain adaptation in the absence of source domain data. In *Proc. ACM SIGKDD*, pages 451–460, 2016.

[7] W.-S. Chu, F. De la Torre, and J. F. Cohn. Selective transfer machine for personalized facial expression analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(3):529–545, 2017.

[8] N. Courty, R. Flamary, D. Tuia, and A. Rakotomamonjy. Optimal transport for domain adaptation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(9):1853–1865, 2017.

[9] G. Csurka. Domain adaptation for visual applications: A comprehensive survey. *arXiv preprint arXiv:1702.05374*, 2017.

[10] G. Csurka, B. Chidlovskii, and S. Clinchant. Adapted domain specific class means. In *Proc. ICCV Workshop*, pages 7–11, 2015.

[11] G. Csurka, B. Chidlovskii, and F. Perronnin. Domain adaptation with a domain specific class means classifier. In *Proc. ECCV Workshop*, pages 32–46, 2014.

[12] C. Ding and T. Li. Adaptive dimension reduction using discriminant analysis and k-means clustering. In *Proc. ICML*, pages 521–528, 2007.

[13] Z. Ding, S. Li, M. Shao, and Y. Fu. Graph adaptive knowledge transfer for unsupervised domain adaptation. In *Proc. ECCV*, pages 36–52, 2018.

[14] T. Evgeniou and M. Pontil. Regularized multi-task learning. In *Proc. ACM SIGKDD*, pages 109–117, 2004.

[15] B. Fernando, A. Habrard, M. Sebban, and T. Tuytelaars. Unsupervised visual domain adaptation using subspace alignment. In *Proc. ICCV*, pages 2960–2967, 2013.

[16] K. Fukunaga. *Introduction to statistical pattern recognition*. Academic Press Professional, Inc., 1990.

[17] Y. Ganin and V. Lempitsky. Unsupervised domain adaptation by backpropagation. In *Proc. ICML*, pages 1180–1189, 2015.

[18] M. Ghifary, D. Balduzzi, W. B. Kleijn, and M. Zhang. Scatter component analysis: A unified framework for domain adaptation and domain generalization. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(7):1414–1430, 2017.

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

[20] B. Gholami, O. Rudovic, and V. Pavlovic. Punda: Probabilistic unsupervised domain adaptation for knowledge transfer across visual categories. In *Proc. ICCV*, pages 3601–3610, 2017.

[21] B. Gong, Y. Shi, F. Sha, and K. Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *Proc. CVPR*, pages 2066–2073, 2012.

[22] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Proc. NIPS*, pages 2672–2680, 2014.

[23] R. Gopalan, R. Li, and R. Chellappa. Domain adaptation for object recognition: An unsupervised approach. In *Proc. ICCV*, pages 999–1006, 2011.

[24] A. Gretton, K. M. Borgwardt, M. Rasch, B. Schölkopf, and A. J. Smola. A kernel method for the two-sample-problem. In *Proc. NIPS*, pages 513–520, 2007.

[25] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical report 7694, 2007.

[26] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proc. CVPR*, pages 770–778, 2016.

[27] S. Herath, M. T. Harandi, and F. Porikli. Learning an invariant hilbert space for domain adaptation. In *Proc. CVPR*, pages 3956–3965, 2017.

[28] J. J. Hull. A database for handwritten text recognition research. *IEEE Trans. Pattern Anal. Mach. Intell.*, 16(5):550–554, 1994.

[29] G. Kang, L. Zheng, Y. Yan, and Y. Yang. Deep adversarial attention alignment for unsupervised domain adaptation: the benefit of target expectation maximization. In *Proc. ECCV*, pages 420–436, 2018.

[30] Z. Kang, K. Grauman, and F. Sha. Learning with whom to share in multi-task feature learning. In *Proc. ICML*, pages 521–528, 2011.

[31] A. Khosla, T. Zhou, T. Malisiewicz, A. A. Efros, and A. Torralba. Undoing the damage of dataset bias. In *Proc. ECCV*, pages 158–171, 2012.

[32] P. Koniusz, Y. Tas, and F. Porikli. Domain adaptation by mixture of alignments of second-or higher-order scatter tensors. In *Proc. CVPR*, pages 7139–7148, 2017.

[33] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proc. IEEE*, 86(11):2278–2324, 1998.

[34] W. Li, L. Chen, D. Xu, and L. Van Gool. Visual recognition in rgb images and videos by learning from rgb-d data. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(8):2030–2036, 2018.

[35] Y. Li, M. Gong, X. Tian, T. Liu, and D. Tao. Domain generalization via conditional invariant representations. In *Proc. AAAI*, pages 3579–3587, 2018.

[36] J. Liang, R. He, Z. Sun, and T. Tan. Aggregating randomized clustering-promoting invariant projections for domain adaptation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 41(5):1027–1042, 2019.

[37] H. Liu, M. Shao, Z. Ding, and Y. Fu. Structure-preserved unsupervised domain adaptation. *IEEE Trans. Knowl. Data Eng.*, 31(4):799–812, 2019.

[38] M.-Y. Liu and O. Tuzel. Coupled generative adversarial networks. In *Proc. NIPS*, pages 469–477, 2016.

[39] M. Long, Y. Cao, J. Wang, and M. I. Jordan. Learning transferable features with deep adaptation networks. In *Proc. ICML*, pages 97–105, 2015.

[40] M. Long, Z. Cao, J. Wang, and M. I. Jordan. Conditional adversarial domain adaptation. In *Proc. NeurIPS*, pages 1640–1650, 2018.

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

[42] M. Long, J. Wang, G. Ding, J. Sun, and P. S. Yu. Transfer joint matching for unsupervised domain adaptation. In *Proc. CVPR*, pages 1410–1417, 2014.

[43] M. Long, H. Zhu, J. Wang, and M. I. Jordan. Unsupervised domain adaptation with residual transfer networks. In *Proc. NIPS*, pages 136–144, 2016.

[44] M. Long, H. Zhu, J. Wang, and M. I. Jordan. Deep transfer learning with joint adaptation networks. In *Proc. ICML*, pages 2208–2217, 2017.

[45] H. Lu, C. Shen, Z. Cao, Y. Xiao, and A. van den Hengel. An embarrassingly simple approach to visual domain adaptation. *IEEE Trans. Image Process.*, 27(7):3403–3417, 2018.

[46] D. Luo, C. Ding, and H. Huang. Linear discriminant analysis: new formulations and overfit analysis. In *Proc. AAAI*, pages 417–422, 2011.

[47] M. Mancini, L. Porzi, S. Rota Bulò, B. Caputo, and E. Ricci. Boosting domain adaptation by discovering latent domains. In *Proc. CVPR*, pages 3771–3780, 2018.

[48] K. Muandet, D. Balduzzi, and B. Schölkopf. Domain generalization via invariant feature representation. In *Proc. ICML*, pages 10–18, 2013.

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

[50] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. In *Proc. NIPS Workshop*, 2011.

[51] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang. Domain adaptation via transfer component analysis. *IEEE Trans. Neural Netw.*, 22(2):199–210, 2011.

[52] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.*, 22(10):1345–1359, 2010.

[53] P. O. Pinheiro. Unsupervised domain adaptation with similarity learning. In *Proc. CVPR*, pages 8004–8013, 2018.

[54] P. Russo, F. M. Carlucci, T. Tommasi, and B. Caputo. From source to target and back: symmetric bi-directional adaptive gan. In *Proc. CVPR*, pages 8099–8108, 2018.

[55] K. Saenko, B. Kulis, M. Fritz, and T. Darrell. Adapting visual category models to new domains. In *Proc. ECCV*, pages 213–226, 2010.

[56] S. Sankaranarayanan and Y. Balaji. Generate to adapt: Aligning domains using generative adversarial networks. In *Proc. CVPR*, pages 8503–8512, 2018.

[57] B. Schölkopf, A. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998.

[58] M. Sugiyama, S. Nakajima, H. Kashima, P. V. Buenau, and M. Kawanabe. Direct importance estimation with model selection and its application to covariate shift adaptation. In *Proc. NIPS*, pages 1433–1440, 2008.

[59] B. Sun, J. Feng, and K. Saenko. Return of frustratingly easy domain adaptation. In *Proc. AAAI*, pages 1–8, 2016.

[60] B. Sun and K. Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *Proc. ECCV Workshop*, pages 443–450, 2016.

[61] T. Tommasi and B. Caputo. Frustratingly easy nbnn domain adaptation. In *Proc. ICCV*, pages 897–904, 2013.

[62] A. Torralba and A. A. Efros. Unbiased look at dataset bias. In *Proc. CVPR*, pages 1521–1528, 2011.

[63] E. Tzeng, J. Hoffman, T. Darrell, and K. Saenko. Simultaneous deep transfer across domains and tasks. In *Proc. ICCV*, pages 4068–4076, 2015.

[64] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell. Adversarial discriminative domain adaptation. In *Proc. CVPR*, pages 7167–7266, 2017.

[65] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474*, 2014.

[66] H. Venkateswara, S. Chakraborty, and S. Panchanathan. Deep-learning systems for domain adaptation in computer vision: Learning transferable feature representations. *IEEE Signal Process. Mag.*, 34(6):117–129, 2017.

[67] J. Wang, W. Feng, Y. Chen, H. Yu, M. Huang, and P. S. Yu. Visual domain adaptation with manifold embedded distribution alignment. In *Proc. ACM MM*, pages 402–410, 2018.

[68] S. Xie, Z. Zheng, L. Chen, and C. Chen. Learning semantic representations for unsupervised domain adaptation. In *Proc. ICML*, pages 5423–5432, 2018.

[69] R. Xu, Z. Chen, W. Zuo, J. Yan, and L. Lin. Deep cocktail network: Multi-source unsupervised domain adaptation with category shift. In *Proc. CVPR*, pages 3964–3973, 2018.

[70] Y. Xu, S. J. Pan, H. Xiong, Q. Wu, R. Luo, H. Min, and H. Song. A unified framework for metric transfer learning. *IEEE Trans. Knowl. Data Eng.*, 29(6):1158–1171, 2017.

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

[72] B. Zadrozny. Learning and evaluating classifiers under sample selection bias. In *Proc. ICML*, 2004.

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

[74] J. Zhang, W. Li, and P. Ogunbona. Joint geometrical and statistical alignment for visual domain adaptation. In *Proc. CVPR*, pages 5150–5158, 2017.

[75] W. Zhang, W. Ouyang, W. Li, and D. Xu. Collaborative and adversarial network for unsupervised domain adaptation. In *Proc. CVPR*, pages 3801–3809, 2018.

[76] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proc. ICCV*, pages 2242–2251, 2017.