

# Visual Domain Adaptation with Manifold Embedded Distribution Alignment\*

Jindong Wang, Wenjie Feng, Yiqiang Chen<sup>†</sup>  
Institute of Computing Technology, CAS, Beijing, China  
{wangjindong,yqchen}@ict.ac.cn  
fengwenjie@software.ict.ac.cn

Han Yu<sup>‡</sup>  
School of Computer Science and Engineering, NTU  
Singapore  
han.yu@ntu.edu.sg

Meiyu Huang  
Qian Xuesen Lab. of Space Technology, China Academy of  
Space Technology, Beijing, China  
huangmeiyu@qxslab.cn

Philip S. Yu<sup>§</sup>  
Department of Computer Science, UIC  
Chicago, USA  
psyu@uic.edu

## ABSTRACT

Visual domain adaptation aims to learn robust classifiers for the target domain by leveraging knowledge from a source domain. Existing methods either attempt to align the cross-domain distributions, or perform manifold subspace learning. However, there are two significant challenges: (1) *degenerated feature transformation*, which means that distribution alignment is often performed in the original feature space, where feature distortions are hard to overcome. On the other hand, subspace learning is not sufficient to reduce the distribution divergence. (2) *unevaluated distribution alignment*, which means that existing distribution alignment methods only align the marginal and conditional distributions with equal importance, while they fail to evaluate the different importance of these two distributions in real applications. In this paper, we propose a **Manifold Embedded Distribution Alignment (MEDA)** approach to address these challenges. MEDA learns a domain-invariant classifier in Grassmann manifold with structural risk minimization, while performing dynamic distribution alignment to quantitatively account for the relative importance of marginal and conditional distributions. To the best of our knowledge, MEDA is the *first* attempt to perform dynamic distribution alignment for manifold domain adaptation. Extensive experiments demonstrate that MEDA shows significant improvements in classification accuracy compared to state-of-the-art traditional and deep methods.

\*The first two authors contributed equally.

<sup>†</sup>Corr. to Y.Chen. J. Wang and Y. Chen are also affiliated with Beijing Key Lab. of Mobile Computing and Pervasive Devices. W. Feng is also with CAS Key Lab. of Network Data Science & Technology. J. Wang and W. Feng are also with University of Chinese Academy of Sciences, Beijing, China.

<sup>‡</sup>H. Yu is also with the Joint NTU-UBC Research Centre of Excellence in Active Living for the Elderly (LILY), and the Alibaba-NTU Singapore Joint Research Institute.

<sup>§</sup>P. Yu is also affiliated with Institute for Data Science, Tsinghua University, Beijing, China.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from [permissions@acm.org](mailto:permissions@acm.org).

MM '18, Oct. 22–26, 2018, Seoul, Republic of Korea

© 2018 Association for Computing Machinery.

ACM ISBN 978-1-4503-5665-7/18/10...\$15.00

<https://doi.org/10.1145/3240508.3240512>

## KEYWORDS

Domain Adaptation, Transfer Learning, Distribution Alignment, Subspace Learning

### ACM Reference format:

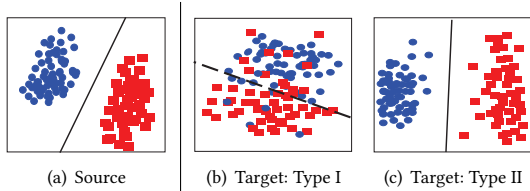
Jindong Wang, Wenjie Feng, Yiqiang Chen, Han Yu, Meiyu Huang, and Philip S. Yu. 2018. Visual Domain Adaptation with Manifold Embedded Distribution Alignment. In *Proceedings of 2018 ACM Multimedia Conference, Seoul, Republic of Korea, Oct. 22–26, 2018 (MM '18)*, 9 pages. <https://doi.org/10.1145/3240508.3240512>

## 1 INTRODUCTION

The rapid growth of online media and content sharing applications has stimulated a great demand for automatic recognition and analysis for images and other multimedia data [8, 20]. Unfortunately, it is often expensive and time-consuming to acquire sufficient labeled data to train machine learning models. Thus, it is often necessary to leverage the abundant labeled samples in some existing domains to facilitate learning in a new target domain. Domain adaptation [27, 36] has been a promising approach to solve such cross-domain learning problems.

Since the distributions of the source and target domains are different, the key to successful adaptation is to reduce the distribution divergence. To this end, existing work can be summarized into two main categories: (a) *instance reweighting* [9, 39], which reuses samples from the source domain according to some weighting technique; and (b) *feature matching*, which either performs subspace learning by exploiting the subspace geometrical structure [13, 15, 30], or distribution alignment to reduce the marginal or conditional distribution divergence between domains [23, 40]. Our focus is on feature matching methods. There are two significant challenges in existing methods, i.e. *degenerated feature transformation* and *unevaluated distribution alignment*.

*Degenerated feature transformation* means that both subspace learning and distribution alignment can only reduce, but not remove the distribution divergence [1]. Specifically, subspace learning [13, 15, 30] conducts subspace transformation to obtain better feature representations. However, feature divergence is not eliminated after subspace transformation [22] since subspace learning only utilizes the subspace or manifold structure, but fails to perform feature alignment. On the other hand, distribution alignment [23, 26, 37]



**Figure 1: Examples of two different target domains w.r.t. the same source domain during distribution adaption.**

usually reduces the distribution distance in the original feature space, where features are often distorted [3] which makes it hard to reduce the divergence between domains. Therefore, it is critical to exploit both the advantages of subspace learning and distribution alignment to further facilitate domain adaptation.

*Unevaluated distribution alignment* means that existing work [19, 23, 33, 40] only attempted to align the marginal and conditional distributions with *equal* weights. But they failed to evaluate the relative importance of these two distributions. For example, when two domains are very dissimilar (Figure 1(a)  $\rightarrow$  1(b)), the marginal distribution is more important to align. When the marginal distributions are close (Figure 1(a)  $\rightarrow$  1(c)), the conditional distribution should be given more weight. However, there is no alignment method which can *quantitatively* account for the importance of these two distributions in conjunction.

As far as we know, there has been no previous work that tackle these two challenges together. In this paper, we propose a novel **Manifold Embedded Distribution Alignment (MEDA)** method to address the challenges of both degenerated feature transformation and unevaluated distribution alignment. MEDA learns a domain-invariant classifier in Grassmann manifold with structural risk minimization, while performing dynamic distribution alignment by considering the different importance of marginal and conditional distributions. We also provide a feasible solution to quantitatively evaluate the importance of distributions. To our best knowledge, MEDA is the *first* attempt to reveal the relative importance of marginal and conditional distributions in transfer learning.

This work makes the following contributions:

1) We propose the MEDA approach for domain adaptation. MEDA is capable of addressing both the challenges of degenerated feature transformation and unevaluated distribution alignment.

2) We propose the *first* quantitative evaluation of the relative importance of marginal and conditional distributions in domain adaptation. This is significantly useful in future research on transfer learning.

3) Extensive experiments on 7 real-world image datasets demonstrate that compared to several state-of-the-art traditional and deep methods, MEDA achieves a significant improvement of 3.5% in average classification accuracy.

## 2 RELATED WORK

MEDA substantially distinguishes from existing feature matching domain adaptation methods in several aspects:

*Subspace learning.* Subspace Alignment (SA) [13] aligned the base vectors of both domains, but failed to adapt feature distributions. Subspace distribution alignment (SDA) [31] extended SA by adding the subspace variance adaptation. However, SDA did not consider

the local property of subspaces and ignored conditional distribution alignment. CORAL [30] aligned subspaces in second-order statistics, but it did not consider the distribution alignment. Scatter component analysis (SCA) [14] converted the samples into a set of subspaces (i.e. scatters) and then minimized the divergence between them. GFK [15] extended the idea of sampled points in manifold [16] and proposed to learn the geodesic flow kernel between domains. The work of [4] used a Hellinger distance to approximate the geodesic distance in Riemann space. [3] proposed to use Grassmann for domain adaptation, but they ignored the conditional distribution alignment. Different from these approaches, MEDA can learn a domain-invariant classifier in the manifold and align both marginal and conditional distributions.

*Distribution alignment.* MEDA substantially differs from existing work that only align marginal or conditional distribution [26]. Joint distribution adaptation (JDA) [23] proposed to match both distributions with equal weights. Others extended JDA by adding regularization [22], sparse representation [38], structural consistency [19], domain invariant clustering [33], and label propagation [40]. The main differences between MEDA and these methods are: 1) These work treats the two distributions equally. However, when there is a greater discrepancy between both distributions, they cannot evaluate their relative importance and thus lead to undermined performance. Our work is capable of evaluating the quantitative importance of each distribution via considering their different effects. 2) These methods are designed only in the original space, where feature distortion will hinder the performance. MEDA can align the distributions in the manifold to overcome the feature distortions.

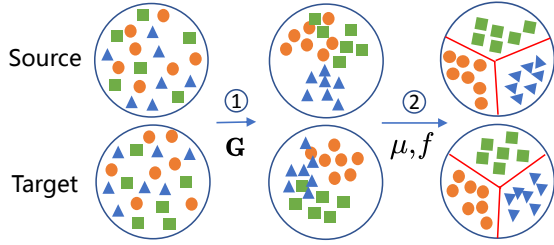
*Domain-invariant classifier learning.* The recent work of ARTL [22], DIP [2, 3], and DMM [7] also aimed to build a domain-invariant classifier. However, ARTL and DMM can be undermined by feature distortion in original space, and they failed to leverage the different importance of distributions. DIP mainly focused on feature transformation and only aligned marginal distributions. MEDA is able to avoid the feature distortion and quantitatively evaluate the importance of marginal and conditional distribution alignment.

## 3 MANIFOLD EMBEDDED DISTRIBUTION ALIGNMENT

### 3.1 Problem Definition

Given a labeled source domain  $\mathcal{D}_s = \{\mathbf{x}_s, y_s\}_{i=1}^n$  and an unlabeled target domain  $\mathcal{D}_t = \{\mathbf{x}_t\}_{j=n+1}^{n+m}$ , assume the feature space  $\mathcal{X}_s = \mathcal{X}_t$ , label space  $\mathcal{Y}_s = \mathcal{Y}_t$ , but marginal probability  $P_s(\mathbf{x}_s) \neq P_t(\mathbf{x}_t)$  with conditional probability  $Q_s(y_s|\mathbf{x}_s) \neq Q_t(y_t|\mathbf{x}_t)$ . The goal of domain adaptation is to learn a classifier  $f: \mathbf{x}_t \mapsto y_t$  to predict the labels  $y_t \in \mathcal{Y}_t$  for the target domain  $\mathcal{D}_t$  using labeled source domain  $\mathcal{D}_s$ .

According to the structural risk minimization (SRM) [35],  $f = \arg \min_{f \in \mathcal{H}_K} \ell(f(\mathbf{x}), y) + R(f)$ , where the first term indicates the loss on data samples, the second term denotes the regularization term, and  $\mathcal{H}_K$  is the Hilbert space induced by kernel function  $K(\cdot, \cdot)$ . Since there is no labels on  $\mathcal{D}_t$ , we can only perform SRM on  $\mathcal{D}_s$ . Moreover, due to the different distributions between  $\mathcal{D}_s$  and  $\mathcal{D}_t$ , it is necessary to add other constraints to maximize the distribution consistency while learning  $f$ .



**Figure 2: The main idea of MEDA.** ① Features in the original space are transformed into manifold space by learning the manifold kernel  $G$ . ② Dynamic distribution alignment (by learning  $\mu$ ) with SRM is performed in manifold to learn the final domain-invariant classifier  $f$ .

### 3.2 Main Idea

MEDA consists of two fundamental steps. Firstly, MEDA performs *manifold feature learning* to address the challenge of degenerated feature transformation. Secondly, MEDA performs *dynamic distribution alignment* to quantitatively account for the relative importance of marginal and conditional distributions to address the challenge of unevaluated distribution alignment. Eventually, a domain-invariant classifier  $f$  can be learned by summarizing these two steps with the principle of SRM. Figure 2 presents the main idea of the proposed MEDA approach.

Formally, if we denote  $g(\cdot)$  the manifold feature learning functional, then  $f$  can be represented as

$$f = \arg \min_{f \in \sum_{i=1}^n \mathcal{H}_K} \ell(f(g(\mathbf{x}_i)), y_i) + \eta \|f\|_K^2 + \lambda \overline{D}_f(\mathcal{D}_s, \mathcal{D}_t) + \rho R_f(\mathcal{D}_s, \mathcal{D}_t) \quad (1)$$

where  $\|f\|_K^2$  is the squared norm of  $f$ . The term  $\overline{D}_f(\cdot, \cdot)$  represents the proposed dynamic distribution alignment. Additionally, we introduce  $R_f(\cdot, \cdot)$  as a Laplacian regularization to further exploit the similar geometrical property of nearest points in manifold  $\mathbb{G}$  [5].  $\eta$ ,  $\lambda$ , and  $\rho$  are regularization parameters accordingly.

The overall learning process of MEDA is in Algorithm 1. In next sections, we first introduce manifold feature learning (learn  $g(\cdot)$ ). Then, we present the dynamic distribution alignment (learn  $\overline{D}_f(\cdot, \cdot)$ ). Eventually, we articulate the learning of  $f$ .

### 3.3 Manifold Feature Learning

Manifold feature learning serves as the preprocessing step to eliminate the threat of *degenerated feature transformation*. MEDA learns  $g(\cdot)$  in the *Grassmann* manifold  $\mathbb{G}(d)$  [18] since features in the manifold have some geometrical structures [5, 18] that can avoid distortion in the original space. And  $\mathbb{G}$  can facilitate classifier learning by treating the original  $d$ -dimensional subspace (i.e. feature vector) as its basic element [4]. Additionally, feature transformation and distribution alignment often have efficient numerical forms and can thus facilitate domain adaptation on  $\mathbb{G}(d)$  [18]. There are several approaches to transform the features into  $\mathbb{G}$  [4, 16], among which we embed Geodesic Flow Kernel (GFK) [15] to learn  $g(\cdot)$  for its computational efficiency. We only introduce the main idea of GFK and the details can be found in its original paper.

When learning manifold features, MEDA tries to model the domains with  $d$ -dimensional subspaces and then embed them into

$\mathbb{G}$ . Let  $\mathcal{S}_s$  and  $\mathcal{S}_t$  denote the PCA subspaces for the source and target domain, respectively.  $\mathbb{G}$  can thus be regarded as a collection of all  $d$ -dimensional subspaces. Each original subspace can be seen as a point in  $\mathbb{G}$ . Therefore, the geodesic flow  $\{\Phi(t) : 0 \leq t \leq 1\}$  between two points can draw a path for the two subspaces. If we let  $\mathcal{S}_s = \Phi(0)$  and  $\mathcal{S}_t = \Phi(1)$ , then finding a geodesic flow from  $\Phi(0)$  to  $\Phi(1)$  equals to transforming the original features into an infinite-dimensional feature space, which eventually eliminates the domain shift. This kind of approach can be seen as an incremental way of ‘walking’ from  $\Phi(0)$  to  $\Phi(1)$ . Specifically, the new features can be represented as  $\mathbf{z} = g(\mathbf{x}) = \Phi(t)^T \mathbf{x}$ . From [15], the inner product of transformed features  $\mathbf{z}_i$  and  $\mathbf{z}_j$  gives rise to a positive semidefinite geodesic flow kernel:

$$\langle \mathbf{z}_i, \mathbf{z}_j \rangle = \int_0^1 (\Phi(t)^T \mathbf{x}_i)^T (\Phi(t)^T \mathbf{x}_j) dt = \mathbf{x}_i^T \mathbf{G} \mathbf{x}_j \quad (2)$$

Thus, the feature in original space can be transformed into Grassmann manifold with  $\mathbf{z} = g(\mathbf{x}) = \sqrt{\mathbf{G}}\mathbf{x}$ .  $\mathbf{G}$  can be computed efficiently by singular value decomposition [15]. Note that  $\sqrt{\mathbf{G}}$  is only an expression form and cannot be computed directly, while its square root is calculated by Denman-Beavers algorithm [10].

### 3.4 Dynamic Distribution Alignment

The purpose of dynamic distribution alignment is to *quantitatively* evaluate the importance of aligning marginal ( $P$ ) and conditional ( $Q$ ) distributions in domain adaptation. Existing methods [23, 40] failed in this evaluation by only assuming that both distributions are equally important. However, this assumption may not be realistic for real applications. For instance, when transferring from Figure 1(a) to 1(b), there is a large difference between datasets. Therefore, the divergence between  $P_s$  and  $P_t$  is more dominant. In contrast, from Figure 1(a) to 1(c), the datasets are similar. Therefore, the distribution divergence in each class ( $Q_s$  and  $Q_t$ ) is more dominant.

#### The adaptive factor:

In view of this phenomenon, we introduce an *adaptive factor* to dynamically leverage the importance of these two distributions. Formally, the dynamic distribution alignment  $\overline{D}_f$  is defined as

$$\overline{D}_f(\mathcal{D}_s, \mathcal{D}_t) = (1 - \mu) D_f(P_s, P_t) + \mu \sum_{c=1}^C D_f^{(c)}(Q_s, Q_t) \quad (3)$$

where  $\mu \in [0, 1]$  is the adaptive factor and  $c \in \{1, \dots, C\}$  is the class indicator.  $D_f(P_s, P_t)$  denotes the marginal distribution alignment, and  $D_f^{(c)}(Q_s, Q_t)$  denotes the conditional distribution alignment for class  $c$ .

When  $\mu \rightarrow 0$ , it means that the distribution distance between the source and the target domains is large. Thus, marginal distribution alignment is more important (Figure 1(a)  $\rightarrow$  1(b)). When  $\mu \rightarrow 1$ , it means that feature distribution between domains is relatively small, so the distribution of each class is dominant. Thus, the conditional distribution alignment is more important (Figure 1(a)  $\rightarrow$  1(c)). When  $\mu = 0.5$ , both distributions are treated equally as in existing methods [23, 40]. Hence, the existing methods can be regarded as the special cases of MEDA. By learning the optimal adaptive factor  $\mu_{opt}$  (which we will discuss later), MEDA can be applied to different domain adaptation problems.

We use the *maximum mean discrepancy* (MMD) [6] to empirically calculate the distribution divergence between domains. As a nonparametric measurement, MMD has been widely applied in many existing methods [14, 26, 40], and its theoretical effectiveness has been verified in [17]. The MMD distance between distributions  $p$  and  $q$  is defined as  $d^2(p, q) = (\mathbb{E}_p[\phi(z_s)] - \mathbb{E}_q[\phi(z_t)])_{\mathcal{H}_K}^2$  where  $\mathcal{H}_K$  is the reproducing kernel Hilbert space (RKHS) induced by feature map  $\phi(\cdot)$ . Here,  $\mathbb{E}[\cdot]$  denotes the mean of the embedded samples. In order to compute an MMD associated with  $f$ , we adopt *projected MMD* [28] and compute the marginal distribution alignment as  $D_f(P_s, P_t) = \|\mathbb{E}[f(z_s)] - \mathbb{E}[f(z_t)]\|_{\mathcal{H}_K}^2$ . Similarly, the conditional distribution alignment is  $D_f^{(c)}(Q_s, Q_t) = \|\mathbb{E}[f(z_s^{(c)})] - \mathbb{E}[f(z_t^{(c)})]\|_{\mathcal{H}_K}^2$ . Then, dynamic distribution alignment can be expressed as

$$\begin{aligned} \overline{D}_f(\mathcal{D}_s, \mathcal{D}_t) = & (1 - \mu) \|\mathbb{E}[f(z_s)] - \mathbb{E}[f(z_t)]\|_{\mathcal{H}_K}^2 \\ & + \mu \sum_{c=1}^C \|\mathbb{E}[f(z_s^{(c)})] - \mathbb{E}[f(z_t^{(c)})]\|_{\mathcal{H}_K}^2 \end{aligned} \quad (4)$$

Note that since  $\mathcal{D}_t$  has no labels, it is not feasible to evaluate the conditional distribution  $Q_t = Q_t(y_t | z_t)$ . Instead, we follow the idea in [37] and use the class conditional distribution  $Q_t(z_t | y_t)$  to approximate  $Q_t$ . In order to evaluate  $Q_t(z_t | y_t)$ , we apply prediction to  $\mathcal{D}_t$  using a base classifier trained on  $\mathcal{D}_s$  to obtain soft labels for  $\mathcal{D}_t$ . The soft labels may be less reliable, so we *iteratively* refine the prediction. Note that we *only* use the base classifier in the first iteration. After that, MEDA can *automatically* refine the labels for  $\mathcal{D}_t$  using results from previous iteration.

#### The quantitative evaluation of the adaptive factor $\mu$ :

We can treat  $\mu$  as a parameter and tune its value by cross-validation techniques. However, there is no labels for the target domain in unsupervised domain adaptation problems. It is extremely hard to calculate the value of  $\mu$ . In this work, we made the *first* attempt towards calculating  $\mu$  (i.e.  $\hat{\mu}$ ) by exploiting the global and local structure of domains. We adopted the  $\mathcal{A}$ -distance [6] as the basic measurement. The  $\mathcal{A}$ -distance is defined as the error of building a linear classifier to discriminate two domains (i.e. a binary classification). Formally, we denote  $\epsilon(h)$  the error of a linear classifier  $h$  discriminating the two domains  $\mathcal{D}_s$  and  $\mathcal{D}_t$ . Then, the  $\mathcal{A}$ -distance can be defined as

$$d_A(\mathcal{D}_s, \mathcal{D}_t) = 2(1 - 2\epsilon(h)) \quad (5)$$

We can directly compute the marginal  $\mathcal{A}$ -distance using above equation, which is denoted as  $d_M$ . For the  $\mathcal{A}$ -distance between conditional distributions, we denote  $d_c$  as the  $\mathcal{A}$ -distance for the  $c$ th class. It can be calculated as  $d_c = d_A(\mathcal{D}_s^{(c)}, \mathcal{D}_t^{(c)})$ , where  $\mathcal{D}_s^{(c)}$  and  $\mathcal{D}_t^{(c)}$  denote samples from class  $c$  in  $\mathcal{D}_s$  and  $\mathcal{D}_t$ , respectively. Eventually,  $\mu$  can be estimated as

$$\hat{\mu} \approx 1 - \frac{d_M}{d_M + \sum_{c=1}^C d_c} \quad (6)$$

This estimation has to be conducted at every iteration of the dynamic distribution adaptation, since the feature distribution may vary after evaluating the conditional distribution each time. To be noticed, this is the *first* solution to quantitatively estimate the relative importance of each distribution. In fact, this estimation can

be of significant help in future research on transfer learning and domain adaptation.

### 3.5 Learning Classifier $f$

After manifold feature learning and dynamic distribution alignment,  $f$  can be learned by summarizing SRM over  $\mathcal{D}_s$  and distribution alignment. Adopting the square loss  $l_2$ ,  $f$  can be represented as

$$\begin{aligned} f = \arg \min_{f \in \mathcal{H}_K} \sum_{i=1}^n (y_i - f(z_i))^2 + \eta \|f\|_K^2 \\ + \lambda \overline{D}_f(\mathcal{D}_s, \mathcal{D}_t) + \rho R_f(\mathcal{D}_s, \mathcal{D}_t) \end{aligned} \quad (7)$$

In order to perform efficient learning, we now reformulate each term in detail.

**SRM on the Source Domain:** Using the representer theorem [5],  $f$  admits the expansion

$$f(z) = \sum_{i=1}^{n+m} \beta_i K(z_i, z) \quad (8)$$

where  $\boldsymbol{\beta} = (\beta_1, \beta_2, \dots)^T \in \mathbb{R}^{(n+m) \times 1}$  is the coefficients vector and  $K$  is a kernel. Then, SRM on  $\mathcal{D}_s$  can be

$$\begin{aligned} \sum_{i=1}^n (y_i - f(z_i))^2 + \eta \|f\|_K^2 \\ = \sum_{i=1}^{n+m} \mathbf{A}_{ii} (y_i - f(z_i))^2 + \eta \|f\|_K^2 \\ = \|(\mathbf{Y} - \boldsymbol{\beta}^T \mathbf{K}) \mathbf{A}\|_F^2 + \eta \text{tr}(\boldsymbol{\beta}^T \mathbf{K} \boldsymbol{\beta}) \end{aligned} \quad (9)$$

where  $\|\cdot\|_F$  is the Frobenious norm.  $\mathbf{K} \in \mathbb{R}^{(n+m) \times (n+m)}$  is the kernel matrix with  $\mathbf{K}_{ij} = K(z_i, z_j)$ , and  $\mathbf{A} \in \mathbb{R}^{(n+m) \times (n+m)}$  is a diagonal domain indicator matrix with  $\mathbf{A}_{ii} = 1$  if  $i \in \mathcal{D}_s$ , otherwise  $\mathbf{A}_{ii} = 0$ .  $\mathbf{Y} = [y_1, \dots, y_{n+m}]$  is the label matrix from source and the target domains.  $\text{tr}(\cdot)$  denotes the trace operation. Although the labels for  $\mathcal{D}_t$  are unavailable, they can be filtered out by the indicator matrix  $\mathbf{A}$ .

**Dynamic distribution alignment:** Using the representer theorem and kernel tricks, dynamic distribution alignment in equation (4) becomes

$$\overline{D}_f(\mathcal{D}_s, \mathcal{D}_t) = \text{tr}(\boldsymbol{\beta}^T \mathbf{K} \mathbf{M} \mathbf{K} \boldsymbol{\beta}) \quad (10)$$

where  $\mathbf{M} = (1 - \mu) \mathbf{M}_0 + \mu \sum_{c=1}^C \mathbf{M}_c$  is the MMD matrix with its element calculated by

$$(\mathbf{M}_0)_{ij} = \begin{cases} \frac{1}{n^2}, & \mathbf{z}_i, \mathbf{z}_j \in \mathcal{D}_s \\ \frac{1}{m^2}, & \mathbf{z}_i, \mathbf{z}_j \in \mathcal{D}_t \\ -\frac{1}{mn}, & \text{otherwise} \end{cases} \quad (11)$$

$$(\mathbf{M}_c)_{ij} = \begin{cases} \frac{1}{n_c^2}, & \mathbf{z}_i, \mathbf{z}_j \in \mathcal{D}_s^{(c)} \\ \frac{1}{m_c^2}, & \mathbf{z}_i, \mathbf{z}_j \in \mathcal{D}_t^{(c)} \\ -\frac{1}{m_c n_c}, & \begin{cases} \mathbf{z}_i \in \mathcal{D}_s^{(c)}, \mathbf{z}_j \in \mathcal{D}_t^{(c)} \\ \mathbf{z}_i \in \mathcal{D}_t^{(c)}, \mathbf{z}_j \in \mathcal{D}_s^{(c)} \end{cases} \\ 0, & \text{otherwise} \end{cases} \quad (12)$$

where  $n_c = |\mathcal{D}_s^{(c)}|$  and  $m_c = |\mathcal{D}_t^{(c)}|$ .

---

**Algorithm 1** Manifold Embedded Distribution Alignment

---

**Input:** Data matrix  $\mathbf{X} = [\mathbf{X}_s, \mathbf{X}_t]$ , source domain labels  $\mathbf{y}_s$ , manifold subspace dimension  $d$ , regularization parameters  $\lambda, \eta, \rho$ , and #neighbor  $p$ .

**Output:** Classifier  $f$ .

- 1: Learn manifold feature transformation kernel  $\mathbf{G}$  via equation (2), and get manifold feature  $\mathbf{Z} = \sqrt{\mathbf{G}}\mathbf{X}$ .
  - 2: Train a base classifier using  $\mathcal{D}_s$ , then apply prediction on  $\mathcal{D}_t$  to get its soft labels  $\hat{\mathbf{y}}_t$ .
  - 3: Construct kernel  $\mathbf{K}$  using transformed features  $\mathbf{Z}_s = \mathbf{Z}_{1:n,:}$  and  $\mathbf{Z}_t = \mathbf{Z}_{n+1:n+m,:}$ .
  - 4: **repeat**
  - 5: Calculate the adaptive factor  $\hat{\mu}$  using equation (6), and compute  $\mathbf{M}_0$  and  $\mathbf{M}_c$  by equations (11) and (12).
  - 6: Compute  $\beta^*$  by solving equation (16) and obtain  $f$  via the representer theorem in equation (8).
  - 7: Update the soft labels of  $\mathcal{D}_t$ :  $\hat{\mathbf{y}}_t = f(\mathbf{Z}_t)$ .
  - 8: **until** Convergence
  - 9: **return** Classifier  $f$ .
- 

**Laplacian Regularization:** Additionally, we add a Laplacian regularization term to further exploit the similar geometrical property of nearest points in manifold  $\mathbb{G}$  [5]. We denote the pair-wise affinity matrix as

$$\mathbf{W}_{ij} = \begin{cases} \text{sim}(\mathbf{z}_i, \mathbf{z}_j), & \mathbf{z}_i \in \mathcal{N}_p(\mathbf{z}_j) \text{ or } \mathbf{z}_j \in \mathcal{N}_p(\mathbf{z}_i) \\ 0, & \text{otherwise} \end{cases} \quad (13)$$

where  $\text{sim}(\cdot, \cdot)$  is a similarity function (such as cosine distance) to measure the distance between two points.  $\mathcal{N}_p(\mathbf{z}_i)$  denotes the set of  $p$ -nearest neighbors to point  $\mathbf{z}_i$ .  $p$  is a free parameter and must be set in the method. By introducing Laplacian matrix  $\mathbf{L} = \mathbf{D} - \mathbf{W}$  with diagonal matrix  $\mathbf{D}_{ii} = \sum_{j=1}^{n+m} \mathbf{W}_{ij}$ , the final regularization can be expressed by

$$\begin{aligned} R_f(\mathcal{D}_s, \mathcal{D}_t) &= \sum_{i,j=1}^{n+m} \mathbf{W}_{ij} (f(\mathbf{z}_i) - f(\mathbf{z}_j))^2 \\ &= \sum_{i,j=1}^{n+m} f(\mathbf{z}_i) \mathbf{L}_{ij} f(\mathbf{z}_j) \\ &= \text{tr}(\beta^T \mathbf{K} \mathbf{L} \mathbf{K} \beta) \end{aligned} \quad (14)$$

**Overall Reformulation:** Substituting with equations (9), (10) and (14),  $f$  in equation (7) can be reformulated as

$$\begin{aligned} f &= \arg \min_{f \in \mathcal{H}_K} \|(Y - \beta^T \mathbf{K}) \mathbf{A}\|_F^2 + \eta \text{tr}(\beta^T \mathbf{K} \beta) \\ &\quad + \text{tr}(\beta^T \mathbf{K} (\lambda \mathbf{M} + \rho \mathbf{L}) \mathbf{K} \beta) \end{aligned} \quad (15)$$

Setting derivative  $\partial f / \partial \beta = 0$ , we obtain the solution

$$\beta^* = ((\mathbf{A} + \lambda \mathbf{M} + \rho \mathbf{L}) \mathbf{K} + \eta \mathbf{I})^{-1} \mathbf{A} \mathbf{Y}^T \quad (16)$$

MEDA has a nice property: it can learn the cross-domain function directly *without* the need of explicit classifier training. This makes it significantly different from most existing work such as JGSA [40] and CORAL [30] that further needs to learn a certain classifier.

**Table 1: Statistics of the seven benchmark datasets.**

| Dataset    | #Sample | #Feature    | #Class | Domain       |
|------------|---------|-------------|--------|--------------|
| Office-10  | 1,410   | 800 (4,096) | 10     | A, W, D      |
| Caltech-10 | 1,123   | 800 (4,096) | 10     | C            |
| Office-31  | 4,652   | 4,096       | 31     | A, W, D      |
| USPS       | 1,800   | 256         | 10     | USPS (U)     |
| MNIST      | 2,000   | 256         | 10     | MNIST (M)    |
| ImageNet   | 7,341   | 4,096       | 5      | ImageNet (I) |
| VOC2007    | 3,376   | 4,096       | 5      | VOC (V)      |

## 4 EXPERIMENTS AND EVALUATIONS

In this section, we evaluate the performance of MEDA through extensive experiments on large-scale public datasets. The source code for MEDA is available at <http://transferlearning.xyz/>.

### 4.1 Data Preparation

We adopted seven publicly image datasets: Office+Caltech10, USPS + MNIST, ImageNet + VOC2007, and Office-31. These datasets are popular for benchmarking domain adaptation algorithms and have been widely adopted in most existing work such as [15, 22, 40, 41]. Table 1 lists the statistics of the seven datasets.

**Office-31** [29] consists of three real-world object domains: **Amazon** (A), **Webcam** (W) and **DSLRL** (D). It has 4,652 images with 31 categories. **Caltech-256** (C) contains 30,607 images and 256 categories. Since the objects in Office and Caltech follow different distributions, domain adaptation can help to perform cross-domain recognition. There are 10 common classes in the two datasets. For our experiments, we adopted the **Office+Caltech10** datasets from [15] which contains 12 tasks:  $A \rightarrow D$ ,  $A \rightarrow C, \dots$ ,  $C \rightarrow W$ . In the rest of the paper, we use  $A \rightarrow B$  to denote the knowledge transfer from source domain  $A$  to the target domain  $B$ .

**USPS** (U) and **MNIST** (M) are standard digit recognition datasets containing handwritten digits from 0-9. Since the same digits across two datasets follow different distributions, it is necessary to perform domain adaptation. USPS consists of 7,291 training images and 2,007 test images of size  $16 \times 16$ . MNIST consists of 60,000 training images and 10,000 test images of size  $28 \times 28$ . We construct two tasks:  $U \rightarrow M$  and  $M \rightarrow U$ .

**ImageNet** (I) and **VOC2007** (V) are large standard image recognition datasets. Each dataset can be treated as one domain. The images from the same classes of two domains follow different distributions. In our experiments, we adopt the sub-datasets presented in [12] to construct cross-domain tasks. Five common classes are extracted from both datasets: *bird*, *cat*, *chair*, *dog*, and *person*. Eventually, we have two tasks:  $I \rightarrow V$  and  $V \rightarrow I$ .

### 4.2 State-of-the-art Comparison Methods

We compared the performance of MEDA with several state-of-the-art traditional and deep domain adaptation approaches.

Traditional learning methods:

- 1NN, SVM, and PCA
- Transfer Component Analysis (**TCA**) [26], which performs marginal distribution alignment
- Geodesic Flow Kernel (**GFK**) [15], which performs manifold feature learning
- Joint distribution alignment (**JDA**) [23], which adapts both marginal and conditional distribution

**Table 2: Accuracy (%) on Office+Caltech10 datasets using SURF features.**

| Task    | 1NN  | SVM  | PCA  | TCA  | GFK  | JDA         | TJM  | CORAL | SCA  | ARTL | JGSA        | MEDA        |
|---------|------|------|------|------|------|-------------|------|-------|------|------|-------------|-------------|
| C → A   | 23.7 | 53.1 | 39.5 | 45.6 | 46.0 | 43.1        | 46.8 | 52.1  | 45.6 | 44.1 | 51.5        | <b>56.5</b> |
| C → W   | 25.8 | 41.7 | 34.6 | 39.3 | 37.0 | 39.3        | 39.0 | 46.4  | 40.0 | 31.5 | 45.4        | <b>53.9</b> |
| C → D   | 25.5 | 47.8 | 44.6 | 45.9 | 40.8 | 49.0        | 44.6 | 45.9  | 47.1 | 39.5 | 45.9        | <b>50.3</b> |
| A → C   | 26.0 | 41.7 | 39.0 | 42.0 | 40.7 | 40.9        | 39.5 | 45.1  | 39.7 | 36.1 | 41.5        | <b>43.9</b> |
| A → W   | 29.8 | 31.9 | 35.9 | 40.0 | 37.0 | 38.0        | 42.0 | 44.4  | 34.9 | 33.6 | 45.8        | <b>53.2</b> |
| A → D   | 25.5 | 44.6 | 33.8 | 35.7 | 40.1 | 42.0        | 45.2 | 39.5  | 39.5 | 36.9 | 47.1        | <b>45.9</b> |
| W → C   | 19.9 | 28.8 | 28.2 | 31.5 | 24.8 | 33.0        | 30.2 | 33.7  | 31.1 | 29.7 | 33.2        | <b>34.0</b> |
| W → A   | 23.0 | 27.6 | 29.1 | 30.5 | 27.6 | 29.8        | 30.0 | 36.0  | 30.0 | 38.3 | 39.9        | <b>42.7</b> |
| W → D   | 59.2 | 78.3 | 89.2 | 91.1 | 85.4 | <b>92.4</b> | 89.2 | 86.6  | 87.3 | 87.9 | 90.5        | 88.5        |
| D → C   | 26.3 | 26.4 | 29.7 | 33.0 | 29.3 | 31.2        | 31.4 | 33.8  | 30.7 | 30.5 | 29.9        | <b>34.9</b> |
| D → A   | 28.5 | 26.2 | 33.2 | 32.8 | 28.7 | 33.4        | 32.8 | 37.7  | 31.6 | 34.9 | 38.0        | <b>41.2</b> |
| D → W   | 63.4 | 52.5 | 86.1 | 87.5 | 80.3 | 89.2        | 85.4 | 84.7  | 84.4 | 88.5 | <b>91.9</b> | 87.5        |
| Average | 31.4 | 41.1 | 43.6 | 46.2 | 43.1 | 46.8        | 46.3 | 48.8  | 45.2 | 44.3 | 50.0        | <b>52.7</b> |

**Table 3: Accuracy (%) on USPS+MNIST and ImageNet+VOC2007 datasets.**

| Task    | 1NN  | SVM  | PCA  | TCA  | GFK  | JDA  | TJM  | CORAL | SCA  | ARTL | JGSA | MEDA        |
|---------|------|------|------|------|------|------|------|-------|------|------|------|-------------|
| U → M   | 44.7 | 62.2 | 45.0 | 51.2 | 46.5 | 59.7 | 52.3 | 30.5  | 48.0 | 67.7 | 68.2 | <b>72.1</b> |
| M → U   | 65.9 | 68.2 | 66.2 | 56.3 | 61.2 | 67.3 | 63.3 | 49.2  | 65.1 | 88.8 | 80.4 | <b>89.5</b> |
| I → V   | 50.8 | 52.4 | 58.4 | 63.7 | 59.5 | 63.4 | 63.7 | 59.6  | -    | 62.4 | 52.3 | <b>67.3</b> |
| V → I   | 38.2 | 42.7 | 65.1 | 64.9 | 73.8 | 70.2 | 73.0 | 70.3  | -    | 72.2 | 70.6 | <b>74.7</b> |
| Average | 49.9 | 56.3 | 58.7 | 59.0 | 60.2 | 65.1 | 63.1 | 52.4  | -    | 72.8 | 67.9 | <b>75.9</b> |

**Table 4: Accuracy (%) on Office+Caltech10 datasets using DeCaf6 features.**

| Task    | Traditional Methods |       |      |       |       |       |       |       |              |             |       | Deep Methods |         |       |             |        | MEDA |             |
|---------|---------------------|-------|------|-------|-------|-------|-------|-------|--------------|-------------|-------|--------------|---------|-------|-------------|--------|------|-------------|
|         | 1NN                 | SVM   | PCA  | TCA   | GFK   | JDA   | TJM   | SCA   | ARTL         | JGSA        | CORAL | DMM          | AlexNet | DDC   | DAN         | DCORAL |      | DUCEA       |
| C → A   | 87.3                | 91.6  | 88.1 | 89.8  | 88.2  | 89.6  | 88.8  | 89.5  | 92.4         | 91.4        | 92.0  | 92.4         | 91.9    | 91.9  | 92.0        | 92.4   | 92.8 | <b>93.4</b> |
| C → W   | 72.5                | 80.7  | 83.4 | 78.3  | 77.6  | 85.1  | 81.4  | 85.4  | 87.8         | 86.8        | 80.0  | 87.5         | 83.7    | 85.4  | 90.6        | 91.1   | 91.6 | <b>95.6</b> |
| C → D   | 79.6                | 86.0  | 84.1 | 85.4  | 86.6  | 89.8  | 84.7  | 87.9  | 86.6         | <b>93.6</b> | 84.7  | 90.4         | 87.1    | 88.8  | 89.3        | 91.4   | 91.7 | 91.1        |
| A → C   | 71.7                | 82.2  | 79.3 | 82.6  | 79.2  | 83.6  | 84.3  | 78.8  | 87.4         | 84.9        | 83.2  | 84.8         | 83.0    | 85.0  | 84.1        | 84.7   | 84.8 | <b>87.4</b> |
| A → W   | 68.1                | 71.9  | 70.9 | 74.2  | 70.9  | 78.3  | 71.9  | 75.9  | 88.5         | 81.0        | 74.6  | 84.7         | 79.5    | 86.1  | <b>91.8</b> | -      | -    | 88.1        |
| A → D   | 74.5                | 80.9  | 82.2 | 81.5  | 82.2  | 80.3  | 76.4  | 85.4  | 85.4         | 88.5        | 84.1  | <b>92.4</b>  | 87.4    | 89.0  | 91.7        | -      | -    | 88.1        |
| W → C   | 55.3                | 67.9  | 70.3 | 80.4  | 69.8  | 84.8  | 83.0  | 74.8  | 88.2         | 85.0        | 75.5  | 81.7         | 73.0    | 78.0  | 81.2        | 79.3   | 80.2 | <b>93.2</b> |
| W → A   | 62.6                | 73.4  | 73.5 | 84.1  | 76.8  | 90.3  | 87.6  | 86.1  | 92.3         | 90.7        | 81.2  | 86.5         | 83.8    | 84.9  | 92.1        | -      | -    | <b>99.4</b> |
| W → D   | 98.1                | 100.0 | 99.4 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0        | 100.0       | 100.0 | 98.7         | 100.0   | 100.0 | 100.0       | -      | -    | 99.4        |
| D → C   | 42.1                | 72.8  | 71.7 | 82.3  | 71.4  | 85.5  | 83.8  | 78.1  | 87.3         | 86.2        | 76.8  | 83.3         | 79.0    | 81.1  | 80.3        | 82.8   | 82.5 | <b>87.5</b> |
| D → A   | 50.0                | 78.7  | 79.2 | 89.1  | 76.3  | 91.7  | 90.3  | 90.0  | 92.7         | 92.0        | 85.5  | 90.7         | 87.1    | 89.5  | 90.0        | -      | -    | <b>93.2</b> |
| D → W   | 91.5                | 98.3  | 98.0 | 99.7  | 99.3  | 99.7  | 99.3  | 98.6  | <b>100.0</b> | 99.7        | 99.3  | 99.3         | 97.7    | 98.2  | 98.5        | -      | -    | 97.6        |
| Average | 71.1                | 82.0  | 81.7 | 85.6  | 81.5  | 88.2  | 86.0  | 85.9  | 90.7         | 90.0        | 84.7  | 89.4         | 86.1    | 88.2  | 90.1        | -      | -    | <b>92.8</b> |

- Transfer Joint Matching (**TJM**) [24], which adapts marginal distribution with source sample selection
- Adaptation Regularization (**ARTL**) [22], which learns domain classifier in original space
- CORrelation Alignment (**CORAL**) [30], which performs second-order subspace alignment
- Scatter Component Analysis (**SCA**) [14], which adapts scatterers in subspace
- Joint Geometrical and Statistical Alignment (**JGSA**) [40], which aligns marginal & conditional distributions with label propagation
- Distribution Matching Machine (**DMM**) [7], which learns a transfer SVM to align distributions

And deep domain adaptation methods:

- **AlexNet** [21], which is a standard convnet
- Deep Domain Confusion (**DDC**) [34], which is a single-layer deep adaptation method with MMD loss
- Deep Adaptation Network (**DAN**) [25], which is a multi-layer adaptation method with multiple kernel MMD

- Deep CORAL (**DCORAL**) [32], which is a deep neural network with CORAL loss
- Deep Unsupervised Convolutional Domain Adaptation (**DUCEA**) [41], which is based on attention and CORAL loss

### 4.3 Experimental Setup

For fair comparison, we follow the same protocols as [37, 40, 41] to adopt the extracted features for MEDA and other traditional methods. To be specific, 256 SURF features are used for USPS+MNIST datasets; for Office+Caltech10 datasets, both 800 SURF and 4,096 DeCaf6 [11] features are used; for Office-31 dataset, 4,096 DeCaf6 features are used; for ImageNet+VOC datasets, 4,096 DeCaf6 features are used. Deep methods can be used to the original images.

Parameter setting: The optimal parameters of all comparison methods are set according to their original papers. As for MEDA, we set the manifold feature dimension  $d = 20, 30, 40$  for Office+Caltech10, USPS+MNIST, and ImageNet+VOC datasets, respectively. The iteration number are set to  $T = 10$ . We use the RBF kernel with the bandwidth set to be the variance of inputs. The regularization parameters are set as  $p = 10, \lambda = 10, \eta = 0.1$ , and  $\rho = 1$ . The

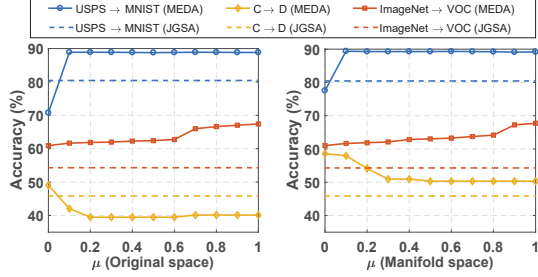


Figure 3: Accuracy in original (left) and manifold space (right) with different  $\mu$ . Dashed lines are best baseline.

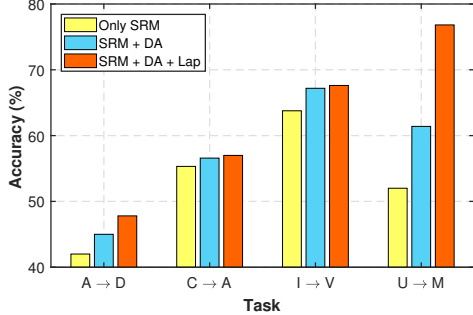


Figure 4: Evaluation of each component.

approach of setting these parameters are in the supplementary file. Additionally, the experiments on parameter sensitivity and convergence analysis in later experiments (Section 4.6 and 4.7) indicate that MEDA stays robust with a wide range of parameter choices.

We adopt classification Accuracy on  $\mathcal{D}_t$  as the evaluation metric, which is widely used in existing literatures [15, 26, 37]:  $Accuracy = \frac{|\{x: x \in \mathcal{D}_t \wedge \hat{y}(x) = y(x)\}|}{|\{x: x \in \mathcal{D}_t\}|}$ , where  $y(x)$  and  $\hat{y}(x)$  are the truth and predicted labels for target domain, respectively.

#### 4.4 Experimental Results and Analysis

The classification accuracy results on the aforementioned datasets are shown in Tables 2, 3, and 4, respectively<sup>1</sup>. From those results, we can make several observations as follows.

Firstly, MEDA outperformed all other traditional and deep comparison methods in most tasks (21/28 tasks). The average classification accuracy of MEDA on 28 tasks was 73.2%. Compared to the best baseline method JGSA (69.7%), the average performance improvement was 3.5%, which showed a significant average error reduction of 11.6%. Note that the results on Office-31 dataset were in the supplementary file<sup>2</sup> due to space constraints, and the observations are the same. Since these results were obtained from a wide range of image datasets, it demonstrates that MEDA is capable of significantly reducing the distribution divergence in domain adaptation problems.

Secondly, the performances of distribution alignment methods (TCA, JDA, ARTL, TJM, JGSA, and DMM) and subspace learning methods (GFK, CORAL, and SCA) were generally worse than MEDA. Each kind of methods has its limitations and cannot handle domain adaptation in specific tasks. This indicates the disadvantages of those methods to cope with degenerated feature transformation

<sup>1</sup>Symbol ‘-’ denotes the result is not available since there is no code or results.

<sup>2</sup>Supplementary file is at <https://www.jianguoyn.com/p/DRuWOFkQjKnsBRjkr2E>.

Table 5: Mean and standard deviation of accuracy in feature learning in both original and manifold space.

| Task                      | Original Space | Manifold Space | Improvement    |
|---------------------------|----------------|----------------|----------------|
| C $\rightarrow$ A         | 44.9 (2.1)     | 56.5 (0.5)     | 25.8% (-76.9%) |
| C $\rightarrow$ W         | 33.5 (4.5)     | 54.0 (0.4)     | 61.4% (-90.9%) |
| C $\rightarrow$ A (DeCaf) | 92.5 (0.2)     | 93.4 (0.1)     | 1.0% (-58.3%)  |
| C $\rightarrow$ W (DeCaf) | 88.4 (1.7)     | 95.5 (0.3)     | 8.1% (-82.6%)  |
| U $\rightarrow$ M         | 64.1 (9.2)     | 71.2 (4.2)     | 11.1% (-54.5%) |
| I $\rightarrow$ V         | 63.0 (2.5)     | 63.7 (2.2)     | 1.1% (-13.2%)  |

Table 6: Performance comparison between  $\mu_{opt}$  and  $\hat{\mu}$ .

| Task                  | C $\rightarrow$ A | W $\rightarrow$ D | C $\rightarrow$ A (DeCaf) | W $\rightarrow$ C (DeCaf) | M $\rightarrow$ U | I $\rightarrow$ V |
|-----------------------|-------------------|-------------------|---------------------------|---------------------------|-------------------|-------------------|
| $\mu_{opt}$           | 57.0              | 89.2              | 93.4                      | 88.0                      | 89.4              | 67.6              |
| $\hat{\mu}$           | 56.5              | 88.5              | 93.4                      | 93.2                      | 89.5              | 67.3              |
| Performance Variation | -0.9%             | -0.8%             | 0                         | +5.9%                     | +0.1%             | -0.4%             |

and unevaluated distribution alignment. After manifold or subspace learning, there still exists large domain shift [3]; while feature distortion will undermine the distribution alignment methods.

Thirdly, MEDA also outperformed the deep methods (AlexNet, DDC, DAN, DCORAL, and DUCDA) on Office+Caltech10 datasets. Deep methods often have to tune a lot of hyperparameters before obtaining the optimal results. Compared to them, MEDA only involves several parameters that can easily be set by human experience or cross-validation. This implies the accuracy and efficiency of MEDA in domain adaptation problems over other deep methods.

#### 4.5 Effectiveness Analysis

**4.5.1 Manifold Feature Learning.** We investigate the effectiveness of manifold feature learning in handling the degenerated feature transformation challenge. To this end, we ran MEDA with and without manifold feature learning on randomly selected tasks. Table 5 showed the mean, standard deviation, and performance improvement of classification accuracy with  $\mu \in \{0, 0.1, \dots, 1\}$ . For instance, the improvement of mean accuracy on task C  $\rightarrow$  A was:  $(56.5 - 44.9)/44.9 \times 100\% = 25.8\%$ . From these results, we can observe that: 1) The performance of all the tasks were improved with manifold feature learning, indicating that transforming features into the manifold alleviates domain shift to some extent and facilitates distribution alignment; 2) The standard deviation of methods that adopted manifold learning with different  $\mu$  could be dramatically reduced. 3) MEDA can also reach a comparable performance without manifold learning, while adding manifold learning would produce better results. This reveals the effectiveness of manifold feature learning to alleviate degenerated feature transformation.

**4.5.2 Dynamic Distribution Alignment.** We verify the effectiveness of dynamic distribution alignment in handling the *unevaluated distribution alignment* challenge. We ran MEDA by searching  $\mu \in \{0, 0.1, \dots, 0.9, 1.0\}$  and compared the performances with the best baseline method (JGSA). From the results in Figure 3, we can clearly observe that the classification accuracy varied with different choice of  $\mu$ . This indicates the *necessity* to consider the different effects between marginal and conditional distributions. We can also observe that the optimal  $\mu$  value varied on different tasks ( $\mu = 0.2, 0, 1$  for three tasks, respectively). Thus, it is necessary to dynamically adjust the distribution alignment between domains according to different tasks. Moreover, the optimal value of  $\mu$  is not unique on certain task. The classification results may be the same even for different  $\mu$ .

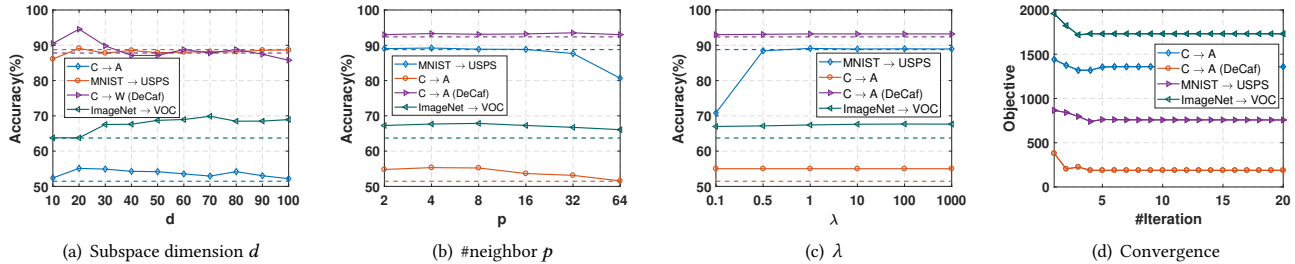


Figure 5: (a)~(c): classification accuracy w.r.t.  $d$ ,  $p$ , and  $\lambda$ , respectively. (d) convergence analysis.

The estimation of  $\mu$ : We evaluate our solution of estimating  $\mu$  (equation (6)). Since the optimal  $\mu$  is not unique, we can not directly compare the value of  $\mu_{opt}$  and  $\hat{\mu}$  to evaluate our solution. Instead, we compare the performances (accuracy values) achieved by  $\mu_{opt}$  and  $\hat{\mu}$ . The results in Table 6 indicated that the performance of estimated  $\hat{\mu}$  was very close to  $\mu_{opt}$ , and sometimes it is better than grid search (M  $\rightarrow$  U). For instance, the performance variation of C  $\rightarrow$  A was  $(57.0 - 56.5)/57.0 \times 100\% = 0.9\%$ . This demonstrates that the effectiveness in estimating  $\mu$ . This estimation solution can be directly applied to future research.

4.5.3 Evaluation of Each Component. When learning the final classifier  $f$ , MEDA involves three components: the structural risk minimization (SRM), the dynamic distribution alignment (DA), and Laplacian regularization (Lap). We empirically evaluated the importance of each component. We randomly selected several tasks and reported the results in Figure 4. Note that we did not run this experiment on the Decaf features of Office+Caltech10 dataset since its results are already satisfied.

Those results clearly indicated that each component is important in MEDA, and they are indispensable. Moreover, we observe that in all tasks, it is more important to align the *distributions*. The reason is that there exists large distribution divergence between two domains. The results also suggests that adding Laplacian regularization is more beneficial in capturing the manifold structure. Additionally, combining the effectiveness of manifold feature learning (Section 4.5.1), it is clear that all components are important for improving the accuracy in domain adaptation tasks.

## 4.6 Parameter Sensitivity

As with other state-of-the-art domain adaptation algorithms [14, 22, 40], MEDA also involves several parameters. In this section, we evaluate the parameter sensitivity.

4.6.1 Subspace Dimension and #neighbor. We investigated the sensitivity of manifold subspace dimension  $d$  and #neighbor  $p$  through experiments with a wide range of  $d \in \{10, 20, \dots, 100\}$  and  $p \in \{2, 4, \dots, 64\}$  on randomly selected tasks. From the results in Figure 5(a) and 5(b), it can be observed that MEDA was robust with regard to different values of  $d$  and  $p$ . Therefore, they can be selected without knowledge in real applications.

4.6.2 Regularization Parameters. We ran MEDA with a wide range of values for regularization parameters  $\lambda$ ,  $\eta$ , and  $\rho$  on several random tasks and compare its performance with the best baseline method. For the lack of space, we only report the results of  $\lambda$  in

Table 7: Running time (s) of ARTL, JGSA, and MEDA.

| Task              | #Sample $\times$ #Feature | ARTL    | JGSA     | MEDA    |
|-------------------|---------------------------|---------|----------|---------|
| C $\rightarrow$ A | 2,081 $\times$ 800        | 29.2    | 95.2     | 32.3    |
| M $\rightarrow$ U | 3,800 $\times$ 256        | 29.1    | 14.6     | 31.4    |
| I $\rightarrow$ V | 10,717 $\times$ 4,096     | 2,648.8 | > 10,000 | 2,931.7 |

Figure 5(c), and the results of  $\rho$  and  $\eta$  can be found in the supplementary file. We observed that MEDA can achieve a robust performance with regard to a wide range of parameter values. Specifically, the best choices of these parameters are:  $\lambda \in [0.5, 1, 1000]$ ,  $\eta \in [0.01, 1]$ , and  $\rho \in [0.01, 5]$ . To sum up, the performance of MEDA stays robust with a wide range of regularization parameter choice.

## 4.7 Convergence and Time Complexity

We validated the convergence of MEDA through empirical analysis. From the results in Figure 5(d), it can be observed that MEDA can reach a steady performance in only a few ( $T < 10$ ) iterations. It indicates the training advantage of MEDA in cross-domain tasks.

We also empirically checked the time complexity of MEDA and compared it with other top two baselines ARTL and JGSA on different tasks. The environment was an Intel Core i7-4790 CPU with 24 GB memory. Note that the time complexity of deep methods are not comparable with MEDA since they require a lot of backpropagations. The results in Table 7 reveal that except its superiority in classification accuracy, MEDA also achieved a running time complexity comparable to top two best baseline methods.

## 5 CONCLUSIONS

In this paper, we propose a novel Manifold Embedded Distribution Alignment (MEDA) approach for visual domain adaptation. Compared to existing work, MEDA is the first attempt to handle the challenges of both degenerated feature transformation and unevaluated distribution alignment. MEDA can learn the domain-invariant classifier with the principle of structural risk minimization while performing dynamic distribution alignment. We also provide a feasible solution to quantitatively calculate the adaptive factor. We conducted extensive experiments on several large-scale publicly available image classification datasets. The results demonstrate the superiority of MEDA against other state-of-the-art traditional and deep domain adaptation methods.

## ACKNOWLEDGMENTS

This work is supported by National Key R & D Program of China (2016YFB1001200), NSFC (61572471, 61702520, 61672313), NSF through grants IIS-1526499, IIS-1763325, CNS-1626432, and Nanyang Assistant Professorship (NAP) of Nanyang Technological University.



## REFERENCES

- [1] Rahaf Aljundi, Rémi Emonet, Damien Muselet, and Marc Sebban. 2015. Landmarks-based kernelized subspace alignment for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 56–63.
- [2] Mahsa Baktashmotlagh, Mehrtash Harandi, and Mathieu Salzmann. 2016. Distribution-matching embedding for visual domain adaptation. *The Journal of Machine Learning Research* 17, 1 (2016), 3760–3789.
- [3] Mahsa Baktashmotlagh, Mehrtash T Harandi, Brian C Lovell, and Mathieu Salzmann. 2013. Unsupervised domain adaptation by domain invariant projection. In *Proceedings of the IEEE International Conference on Computer Vision*. 769–776.
- [4] Mahsa Baktashmotlagh, Mehrtash T Harandi, Brian C Lovell, and Mathieu Salzmann. 2014. Domain adaptation on the statistical manifold. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2481–2488.
- [5] Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. 2006. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of machine learning research* 7, Nov (2006), 2399–2434.
- [6] Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. 2007. Analysis of representations for domain adaptation. In *Advances in neural information processing systems*. 137–144.
- [7] Yue Cao, Mingsheng Long, and Jianmin Wang. 2018. Unsupervised Domain Adaptation with Distribution Matching Machines. In *Proceedings of the 2018 AAAI International Conference on Artificial Intelligence*.
- [8] Long Chen, Hanwang Zhang, Jun Xiao, Wei Liu, and Shih-Fu Chang. 2018. Zero-Shot Visual Recognition using Semantics-Preserving Adversarial Embedding Network. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- [9] Wenyan Dai, Qiang Yang, Gui-Rong Xue, and Yong Yu. 2007. Boosting for transfer learning. In *Proceedings of the 24th international conference on Machine learning (ICML)*. ACM, 193–200.
- [10] Eugene D Denman and Alex N Beavers Jr. 1976. The matrix sign function and computations in systems. *Applied mathematics and Computation* 2, 1 (1976), 63–94.
- [11] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. 2014. Decaf: A deep convolutional activation feature for generic visual recognition. In *International conference on machine learning*. 647–655.
- [12] Chen Fang, Ye Xu, and Daniel N Rockmore. 2013. Unbiased metric learning: On the utilization of multiple datasets and web images for softening bias. In *Proceedings of the IEEE International Conference on Computer Vision*. 1657–1664.
- [13] Basura Fernando, Amaury Habrard, Marc Sebban, and Tinne Tuytelaars. 2013. Unsupervised visual domain adaptation using subspace alignment. In *Proceedings of the IEEE international conference on computer vision*. 2960–2967.
- [14] Muhammad Ghifary, David Balduzzi, W Bastiaan Kleijn, and Mengjie Zhang. 2017. Scatter component analysis: A unified framework for domain adaptation and domain generalization. *IEEE transactions on pattern analysis and machine intelligence* 39, 7 (2017), 1414–1430.
- [15] Boqing Gong, Yuan Shi, Fei Sha, and Kristen Grauman. 2012. Geodesic flow kernel for unsupervised domain adaptation. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2066–2073.
- [16] Raghuraman Gopalan, Ruonan Li, and Rama Chellappa. 2011. Domain adaptation for object recognition: An unsupervised approach. In *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 999–1006.
- [17] Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. 2012. A kernel two-sample test. *Journal of Machine Learning Research* 13, Mar (2012), 723–773.
- [18] Jihun Hamm and Daniel D Lee. 2008. Grassmann discriminant analysis: a unifying view on subspace-based learning. In *Proceedings of the 25th international conference on Machine learning*. ACM, 376–383.
- [19] Cheng-An Hou, Yao-Hung Hubert Tsai, Yi-Ren Yeh, and Yu-Chiang Frank Wang. 2016. Unsupervised Domain Adaptation With Label and Structural Consistency. *IEEE Transactions on Image Processing* 25, 12 (2016), 5552–5562.
- [20] Bogdan Ionescu, Mihai Lupu, Maia Rohm, Alexandru Lucian Gînsca, and Henning Müller. 2018. Datasets column: diversity and credibility for social images and image retrieval. *ACM SIGMultimedia Records* 9, 3 (2018), 7.
- [21] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.
- [22] Mingsheng Long, Jianmin Wang, Guiguang Ding, Sinno Jialin Pan, and S Yu Philip. 2014. Adaptation regularization: A general framework for transfer learning. *IEEE Transactions on Knowledge and Data Engineering* 26, 5 (2014), 1076–1089.
- [23] Mingsheng Long, Jianmin Wang, Guiguang Ding, Jiaguang Sun, and Philip S Yu. 2013. Transfer feature learning with joint distribution adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*. 2200–2207.
- [24] Mingsheng Long, Jianmin Wang, Guiguang Ding, Jiaguang Sun, and Philip S Yu. 2014. Transfer joint matching for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1410–1417.
- [25] Mingsheng Long, Jianmin Wang, Jiaguang Sun, and S Yu Philip. 2015. Domain invariant transfer kernel learning. *IEEE Transactions on Knowledge and Data Engineering* 27, 6 (2015), 1519–1532.
- [26] Sinno Jialin Pan, Ivor W Tsang, James T Kwok, and Qiang Yang. 2011. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks* 22, 2 (2011), 199–210.
- [27] Sinno Jialin Pan and Qiang Yang. 2010. A survey on transfer learning. *Knowledge and Data Engineering, IEEE Transactions on* 22, 10 (2010), 1345–1359.
- [28] Brian Quanz and Jun Huan. 2009. Large margin transductive transfer learning. In *Proceedings of the 18th ACM conference on Information and knowledge management*. ACM, 1327–1336.
- [29] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. 2010. Adapting visual category models to new domains. In *European conference on computer vision*. Springer, 213–226.
- [30] Baochen Sun, Jiashi Feng, and Kate Saenko. 2016. Return of Frustratingly Easy Domain Adaptation. In *AAAI*, Vol. 6. 8.
- [31] Baochen Sun and Kate Saenko. 2015. Subspace Distribution Alignment for Unsupervised Domain Adaptation. In *BMVC*. 24–41.
- [32] Baochen Sun and Kate Saenko. 2016. Deep coral: Correlation alignment for deep domain adaptation. In *European Conference on Computer Vision*. Springer, 443–450.
- [33] Jafar Tahmoresnezhad and Sattar Hashemi. 2016. Visual domain adaptation via transfer feature learning. *Knowledge and Information Systems* (2016), 1–21.
- [34] Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. 2014. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474* (2014).
- [35] Vladimir Naumovich Vapnik and Vladimir Vapnik. 1998. *Statistical learning theory*. Vol. 1. Wiley New York.
- [36] Jindong Wang et al. 2018. Everything about Transfer Learning and Domain Adaption. <http://transferlearning.xyz>. (2018).
- [37] Jindong Wang, Yiqiang Chen, Shuji Hao, Wenjie Feng, and Zhiqi Shen. 2017. Balanced distribution adaptation for transfer learning. In *Data Mining (ICDM), 2017 IEEE International Conference on*. IEEE, 1129–1134.
- [38] Yong Xu, Xiaozhao Fang, Jian Wu, Xuelong Li, and David Zhang. 2016. Discriminative transfer subspace learning via low-rank and sparse representation. *IEEE Transactions on Image Processing* 25, 2 (2016), 850–863.
- [39] Yonghui Xu, Sinno Jialin Pan, Hui Xiong, Qingyao Wu, Ronghua Luo, Huaqing Min, and Hengjie Song. 2017. A Unified Framework for Metric Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering* (2017).
- [40] Jing Zhang, Wanqing Li, and Philip Ogunbona. 2017. Joint Geometrical and Statistical Alignment for Visual Domain Adaptation. In *CVPR*.
- [41] Junbao Zhuo, Shuhui Wang, Weigang Zhang, and Qingming Huang. 2017. Deep Unsupervised Convolutional Domain Adaptation. In *Proceedings of the 2017 ACM on Multimedia Conference*. ACM, 261–269.