Deep Spherical Manifold Gaussian Kernel for Unsupervised Domain Adaptation

Youshan Zhang Brian D. Davison
Computer Science and Engineering, Lehigh University, Bethlehem, PA, USA
{yoz217, bdd3}@lehigh.edu

Abstract

Unsupervised Domain adaptation is an effective method in addressing the domain shift issue when transferring knowledge from an existing richly labeled domain to a new domain. Existing manifold-based methods either are based on traditional models or largely rely on Grassmannian manifold via minimizing differences of single covariance matrices of two domains. In addition, existing pseudo-labeling algorithms inadequately consider the quality of pseudo labels in aligning the conditional distribution between two domains. In this work, a deep spherical manifold Gaussian kernel (DSGK) framework is proposed to map the source and target subspaces into a spherical manifold and reduce the discrepancy between them by embedding both extracted features and a Gaussian kernel. To align the conditional distributions, we further develop an easy-to-hard pseudo label refinement process to improve the quality of the pseudo labels and then reduce categorical spherical manifold Gaussian kernel geodesic loss. Extensive experimental results show that DSGK outperforms state-of-the-art methods, especially on challenging cross-domain learning tasks.

1. Introduction

Massive amounts of labeled data are a prerequisite of most existing machine learning methods. Unfortunately, such a requirement cannot be met in many real-world applications. In addition, collecting sufficient labeled data is a big investment of time and effort. Therefore, it is often necessary to transfer label knowledge from one labeled domain to an unlabeled domain. However, due to domain shift or dataset bias issue [19], it is difficult to improve performance for the unlabeled domain.

Domain adaptation (DA) is proposed to circumvent the domain shift problem. By not requiring additional annotated labels on the new domain, unsupervised DA (UDA) is attractive, as it aims to transfer knowledge learned from a label-rich source domain to a fully unlabeled target domain [14]. Before the popularity of deep features, approaches with hand-crafted features aim to map the two domains into a shared subspace and learn the invariant features [36]. Manifold learning is commonly used to identify the shared space between the source and target domains. There have been efforts made in traditional methods, including sampling geodesic flow (SGF) [6], geodesic flow kernel (GFK) [4], and geodesic sampling on manifolds (GSM) [39]. These methods focus on matching either marginal, conditional, or joint distributions between two domains to learn domain-invariant representations. However, these traditional methods cannot handle large-scale recognition tasks since they require large memory to compute singular value decomposition (SVD) on a Grassmannian manifold. Although discriminative manifold propagation (DMP) [17] proposed a Grassmannn distance to reduce the domain discrepancy, it still largely relies on the differences of covariance matrices, and it cannot avoid complex the SVD process. Further, its Log-Euclidean loss is not the closed-form solution to calculate the intrinsic distance between two domains. Recently, existing deep learning-based methods generate pseudo labels for the target domain to align the conditional distribution and learn the target discriminative representations [32, 33]. However, the credibility of these pseudo labels is unknown. Noisy labels can easily lead to poor alignment and discrimination. As a result, it is easy to cause negative transfer for the target domain.

To address the above challenges, our contributions are three-fold:

- To explicitly measure the intrinsic distance between two domains and reduce computation time, we propose a novel spherical manifold Gaussian kernel geodesic loss, which considers both latent features and discrepancy between covariance matrices.
- We develop an easy-to-hard refinement process to remove the noise labels via T times adjustment, and we then form a pseudo labeled target domain so as to jointly optimize the shared classifier with labeled examples from the source domain.
- We also enforce a categorical spherical manifold Gaussian kernel geodesic loss to reduce conditional discrepancies. Then, our model can jointly align marginal and conditional distributions between two domains.

We conduct extensive experiments on three benchmark datasets (Office-31, Office-Home, and VisDA-2017), achieving higher accuracy than state-of-the-art methods.

2. Related work

Most existing manifold-based methods are only focused on the Grassmannian manifold. The SGF model [6] generated multiple intermediate subspaces between the source and the target domain along with geodesic flow on a Grassmannian manifold. Then, GFK [4] integrated all sampled points along the geodesic as calculated in the SGF model via constructing a kernel function. Manifold embedded distribution alignment (MEDA) [29] further took the advantages of GFK to better represent source and target domain features and then dynamically aligned the marginal and conditional distributions between two domains. Later, geodesic sampling on manifolds (GSM) [39] revealed that the SGF model cannot generate correct intermediate subspaces along the true geodesic, and provided a correct way to sample the intermediate features along the correct geodesic for the general manifold, which is further extended to the sphere, Kendall's shape, and Grassmannian manifold. Luo et al. [17] proposed a discriminative manifold propagation for UDA in a deep learning framework. They proposed Grassmann distance and the Log-Euclidean loss to minimize the difference between the two domains. They did not, however, explicitly measure the intrinsic distance between two domains.

Other frequently used deep learning-based methods rely on minimizing the discrepancy between the source and target distributions by proposing different loss functions, such as Maximum Mean Discrepancy (MMD) [27], CORrelation ALignment [24], and Kullback-Leibler divergence [18]. Recently, Kang et al. [8] extended MMD to the contrastive domain discrepancy loss. Li et al. [9] proposed an Enhanced Transport Distance (ETD) to measure domain discrepancy by establishing the transport distance of attention perception as the predictive feedback of iterative learning classifiers. However, these distance-based metrics can also mix samples of different classes together. Inspired from GAN [5], adversarial learning has shown its power in learning domain invariant representations. The domain discriminator aims to distinguish the source domain from the target domain, while the feature extractor aims to learn domain-invariant representations to fool the domain discriminator [2, 26, 41]. Pseudo-labeling is another technique to address UDA and also achieves substantial performance on multiple tasks. There are also many methods that utilized pseudo-labels to consider label information in the target domain and then minimize the conditional distribution discrepancy between two domains [22, 33, 8, 37]. However, it is still difficult to remove noisy pseudo labels for the target domain. Notably, we project data into a much faster spherical manifold and propose a useful easy-to-hard refinement process.

3. Methodology

3.1. Problem

Here we discuss the unsupervised domain adaptation (UDA) problem and introduce some basic notation. Given a labeled source domain $\mathcal{D}_\mathcal{S} = \{\mathcal{X}^i_\mathcal{S}, \mathcal{Y}^i_\mathcal{S}\}^{\mathcal{N}_\mathcal{S}}_{i=1}$ with $\mathcal{N}_\mathcal{S}$ samples in C categories and an unlabeled target domain $\mathcal{D}_\mathcal{T} = \{\mathcal{X}^j_\mathcal{T}\}^{\mathcal{N}_\mathcal{T}}_{j=1}$ with $\mathcal{N}_\mathcal{T}$ samples in the same C categories ($\mathcal{Y}_\mathcal{T}$ for evaluation only), our challenge is how to get a well-trained classifier so that domain discrepancy is minimized and generalization error in the target domain is reduced.

In UDA, existing manifold-based methods are either based on traditional methods [6, 4, 39] or highly rely on the Grassmannian manifold, which requests complex singular value decomposition (SVD) of the covariance matrices [17]. In addition, to align the conditional distributions of two domains, the reliability of generated pseudo labels is uncertain. These approaches face two critical limitations: (1) the SVD needs more computation time, and minimizing covariance matrices is not equivalent to reducing marginal distribution differences between two domains. It is hence necessary to develop a faster metric to align the marginal distribution of two domains. (2) Noisy pseudo labels can deteriorate the shared classifier. The categorical condition distribution of two domains is difficult to minimize with the lower quality pseudo labels.

To mitigate these shortcomings, we propose a deep spherical manifold Gaussian kernel (DSGK) model. To avoid the complex calculation of SVD, we focus on a spherical manifold. We propose a spherical manifold Gaussian kernel geodesic loss to minimize the marginal distribution, and design an easy-to-hard pseudo-label refinement process to improve the quality of the pseudo-labels in the target domain and then minimize the categorical spherical manifold Gaussian kernel geodesic loss. Therefore, our DSGK model can jointly align the marginal and conditional distributions of two domains.

3.2. Geometry of spherical manifold

Before, we discuss the contributions of this work, we first recap some important concepts on Riemannian manifold as shown in Fig. 1. The n dimensional unit sphere denoted as S^n and can be defined as $S^n = \{(x_1, x_2, \cdots, x_{n+1}) \in \mathbb{R}^{n+1} | \sum_{i=1}^{n+1} x_i^2 = 1\}$. Let p and q be two points on a sphere S^n embedded in $\mathbb{R}^{n+1}$, and the tangent space of S^n at point p is denoted as T_pS^n.

The Logarithmic (Log) map between p and q can be computed in Eq. 1.

$$\begin{aligned} v = \mathrm{Log}(p, q) = \frac{\theta \cdot L}{||L||}, \quad \theta = \arccos(\langle p, q \rangle), \\ L = (q - p \cdot \langle p, q \rangle) \end{aligned} \tag{1}$$

where $p \cdot \langle p, q \rangle$ denotes the projection of the vector q onto

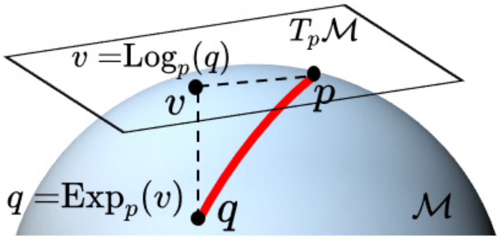


Figure 1: Some basic concepts of geometry on manifold $\mathcal{M}$. p and q are two points on $\mathcal{M}$. $T_p\mathcal{M}$ is the tangent space at point p and p is called the pole of the tangent space. The red curve γ is called the geodesic, which is shortest distance between p and q on $\mathcal{M}$. The Logarithmic map $\mathrm{Log}_p(\cdot)$ projects the point p into the tangent space and the Exponential map $\mathrm{Exp}_p(\cdot)$ projects the element of tangent space v back to the manifold, such that $v = \mathrm{Log}_p(q)$ and $\mathrm{Exp}_p(v) = \gamma(1) = q$.

p. The norm of v is usually a constant, *i.e.* $||v|| = const.$, which is the distance between point p and q.

Given point p, and its tangent vector v from Eq. 1 and t, the Exponential (Exp) map is defined as:

$$\mathrm{Exp}(p, vt) = \cos\theta \cdot p + \frac{\sin\theta}{\theta} \cdot vt, \ \theta = ||vt||. \tag{2}$$

Additional details of Log map and Exp map on the spherical manifold can be found in [31, 40, 34].

3.3. Initial source classifier

Given feature activation function Φ from one backbone network, it maps data into a d dimensional latent space, *i.e.,* $\Phi(\mathcal{X}_\mathcal{Z}) \in \mathbb{R}^{\mathcal{N}_\mathcal{Z} \times d}$, where $\mathcal{Z}$ can be either source domain $\mathcal{S}$ or target domain $\mathcal{T}$. The task in the source domain is trained using the typical cross-entropy loss as follows:

$$\mathcal{L}_\mathcal{S} = -\frac{1}{\mathcal{N}_\mathcal{S}} \sum_{i=1}^{\mathcal{N}_\mathcal{S}} \sum_{c=1}^{C} \mathcal{Y}^i_{\mathcal{S}_c} \log(G_c(\Phi(\mathcal{X}^i_\mathcal{S}))), \tag{3}$$

where $\mathcal{Y}^i_{\mathcal{S}_c} \in [0,1]^C$ is the binary indicator of each class c in true label for observation $\Phi(\mathcal{X}^i_\mathcal{S})$, and $G_c(\Phi(\mathcal{X}^i_\mathcal{S}))$ is the predicted probability of class c (using the softmax function as shown in Fig. 2).

3.4. Kernel for Gaussian distribution

We assume that the batch-wise features vector $B_\mathcal{Z}$ (which is one batch data of $G(\Phi(\mathcal{X}_\mathcal{Z}))$) follows a Gaussian distribution $\mathcal{N}(\mu_\mathcal{Z}, \Sigma_\mathcal{Z})$, where $\mathcal{Z}$ can be either the source or target domain, and $\mu_\mathcal{Z}$ and $\Sigma_\mathcal{Z}$ are the data mean and covariance respectively:

$$\mu_\mathcal{Z} = \frac{1}{\mathcal{N}_B} \sum_{z=1}^{\mathcal{N}_B} B^z_\mathcal{Z} \tag{4}$$

$$\Sigma_\mathcal{Z} = \frac{1}{\mathcal{N}_B} \sum_{z=1}^{\mathcal{N}_B} (B^z_\mathcal{Z} - \mu_\mathcal{Z})(B^z_\mathcal{Z} - \mu_\mathcal{Z})^T \tag{5}$$

Therefore, we can calculate the batch-wise covariance matrix of source $\Sigma_\mathcal{S}$ and target $\Sigma_\mathcal{T}$ domain, respectively. The Gaussian jointly considers the first-order statistic mean and second-order statistic covariance in one single model. Then a form of RBF kernel can be denoted as:

$$\mathcal{K} = exp(-\kappa||\Sigma_\mathcal{S} - \Sigma_\mathcal{T}||^2_F), \tag{6}$$

where $||\cdot||^2_F$ is the Frobenius norm. This differs from previous work [24], which only minimizes the difference between covariance matrices between two domains and reduces Log-Euclidean distance loss. These existing loss functions are largely based on the covariance matrices while ignoring original data. To alleviate this issue, we define the spherical manifold Gaussian kernel geodesic loss to incorporate both covariance matrices and original data.

3.5. Spherical manifold Gaussian kernel geodesic loss

By combining the defined Gaussian kernel (Eq. 6) and batch-wise feature vectors ($B_\mathcal{Z}$), we can measure the intrinsic distance between two domains on one underlying Riemannian spherical manifold. Therefore, we first project two subspaces into a spherical manifold as follows.

$$\begin{aligned} p &= \phi_{Proj.}(B_\mathcal{S} \times \mathcal{K}) \\ q &= \phi_{Proj.}(B_\mathcal{T} \times \mathcal{K}), \end{aligned} \tag{7}$$

where $\phi_{Proj.}$ projects data into a $|C|^2$ dimensional spherical manifold and is defined as $\phi_{Proj.}(x) = x.\mathrm{reshape}(-1)/norm(x)$. It first reshapes data into a $|C|^2$ dimensional space ($|C|$ is the number of categories) and then projects data into a unit $|C|^2$ dimensional sphere as shown in Fig. 2 (for better visualization, we only show a 3D sphere). Therefore, we can define the spherical manifold Gaussian kernel geodesic loss as:

$$\mathcal{L}_\mathcal{K} = ||\,\mathrm{Log}_p(q)||^2_F. \tag{8}$$

$\mathcal{L}_\mathcal{K}$ can estimate the true geodesic distance between two points on the sphere with the closed-form solution in Eq. 1. During the training, minimizing this loss function directly leads to a small distance between the source and target domains, which is equivalent to minimizing the marginal distribution between two domains.

3.6. Conditional distribution alignment

Since there are no labels in the target domain, we first generate the pseudo labels for the target domain. To mitigate the detrimental effects of bad pseudo-labels, we employ a T times easy-to-hard pseudo-label refinement process to improve the quality of the pseudo-labels in the

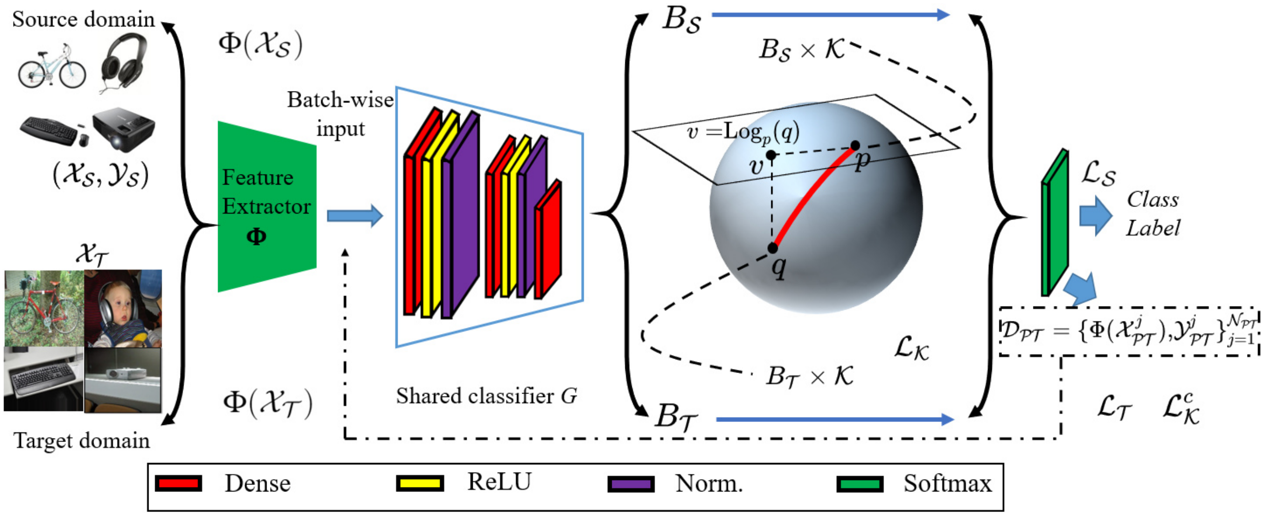

Source domain
$\Phi(\mathcal{X}_\mathcal{S})$
$(\mathcal{X}_\mathcal{S}, \mathcal{Y}_\mathcal{S})$
$\mathcal{X}_\mathcal{T}$
Target domain
$\Phi(\mathcal{X}_\mathcal{T})$
Feature Extractor
$\mathbf{\Phi}$
Batch-wise input
Shared classifier G
$B_\mathcal{S}$
$B_\mathcal{S} \times \mathcal{K}$
$v = \mathrm{Log}_p(q)$
v
p
q
$\mathcal{L}_\mathcal{K}$
$B_\mathcal{T} \times \mathcal{K}$
$B_\mathcal{T}$
$\mathcal{L}_\mathcal{S}$
Class Label
$\mathcal{D}_{\mathcal{PT}} = \{\Phi(\mathcal{X}^j_{\mathcal{PT}}), \mathcal{Y}^j_{\mathcal{PT}}\}_{j=1}^{\mathcal{N}_{\mathcal{PT}}}$
$\mathcal{L}_\mathcal{T}$
$\mathcal{L}^c_\mathcal{K}$
🟥 Dense
🟨 ReLU
🟪 Norm.
🟩 Softmax

Figure 2: Architecture of the DSGK model. We first extract features $\Phi(\mathcal{X}_\mathcal{Z})$ for both source and target domains via Φ using a pre-trained model, and then train the shared classifier G. $\mathcal{L}_\mathcal{S}$ is source classification loss. Spherical manifold Gaussian kernel geodesic loss $\mathcal{L}_\mathcal{K}$ minimize the marginal distribution difference of two domains. The dash-dot line is the generated pseudo labeled target domain using an easy-to-hard refinement process, which will optimize the shared classifier G in T times. $\mathcal{L}_\mathcal{T}$ is the pseudo labeled target domain classification loss, and $\mathcal{L}^c_\mathcal{K}$ is categorical spherical manifold Gaussian kernel geodesic loss to minimize the conditional distribution. Norm.: BatchNormalization layer. Best viewed in color.

target domain. Given the trained classifier G in Eq. 3, we can get predicted probability for each target sample as $\mathrm{Softmax}(G(\Phi(\mathcal{X}^j_\mathcal{T})))$, and the dominate class label is $\mathcal{Y}^j_{\mathcal{PT}} = \max \mathrm{Softmax}(G(\Phi(\mathcal{X}^j_\mathcal{T})))$. Easy samples are those whose dominant predicted class probability is bigger than a certain threshold P_t. Therefore, one easy pseudo labeled target example can be defined as:

$$(\Phi(\mathcal{X}^j_{\mathcal{PT}}), \mathcal{Y}^j_{\mathcal{PT}}) \text{ if } \max(\mathrm{Softmax}(G(\Phi(\mathcal{X}^j_\mathcal{T})))) > P_t \quad (9)$$

During T times easy-to-hard pseudo-label refinement process, for easy examples, P_t has a higher value and for hard examples, P_t has a lower value, hence $P_1 > P_2 > \cdots > P_\mathcal{T}$. We define the pseudo labeled target domain as: $\mathcal{D}_{\mathcal{PT}} = \{\Phi(\mathcal{X}^j_{\mathcal{PT}}), \mathcal{Y}^j_{\mathcal{PT}}\}_{j=1}^{\mathcal{N}_{\mathcal{PT}}}$ with $\mathcal{N}_{\mathcal{PT}}$ samples.

After obtaining the pseudo labeled target domain, we can first optimize the shared classifier G with pseudo labeled target samples in Eq. 10.

$$\mathcal{L}_\mathcal{T} = -\frac{1}{\mathcal{N}_{\mathcal{PT}}} \sum_{j=1}^{\mathcal{N}_{\mathcal{PT}}} \sum_{c=1}^{C} \mathcal{Y}^j_{\mathcal{PT}_c} \log(G_c(\Phi(\mathcal{X}^j_{\mathcal{PT}}))) \quad (10)$$

For the conditional distribution alignment, differing from Sec. 3.4, we use categorical batch-wise feature vectors $\mathcal{C}(B_\mathcal{Z})$ instead of $B_\mathcal{Z}$ since we have labels for both the source and the target domain. $\mathcal{C}(B_\mathcal{Z})$ is the categorical data, which consists of all data in same categories. For example, if the category is 1, then $\mathcal{C}(B_\mathcal{S}) = B_\mathcal{S}[B_{\mathcal{Y}_\mathcal{S}} == 1]$, and $\mathcal{C}(B_\mathcal{T}) = B_\mathcal{T}[B_{\mathcal{Y}_{\mathcal{PT}}} == 1]$. We again assume that $\mathcal{C}(B_\mathcal{Z})$ follows a Gaussian distribution $\mathcal{N}(\mathcal{C}(\mu_\mathcal{Z}), \mathcal{C}(\Sigma_\mathcal{Z}))$, where $\mathcal{Z}$ can be either source or the target domain, and $\mathcal{C}(\mu_\mathcal{Z})$ and $\mathcal{C}(\Sigma_\mathcal{Z})$ are the mean and covariance of $\mathcal{C}(B_\mathcal{Z})$:

$$\mathcal{C}(\mu_\mathcal{Z}) = \frac{1}{\mathcal{N}_{\mathcal{C}(B)}} \sum_{z=1}^{\mathcal{N}_{\mathcal{C}(B)}} \mathcal{C}(B^z_\mathcal{Z}) \quad (11)$$

$$\mathcal{C}(\Sigma_\mathcal{Z}) = \frac{1}{\mathcal{N}_{\mathcal{C}(B)}} \sum_{z=1}^{\mathcal{N}_{\mathcal{C}(B)}} (\mathcal{C}(B^z_\mathcal{Z}) - \mathcal{C}(\mu_\mathcal{Z}))(\mathcal{C}(B^z_\mathcal{Z}) - \mathcal{C}(\mu_\mathcal{Z}))^T \quad (12)$$

Therefore, the categorical RBF kernel can be denoted as:

$$\mathcal{C}(\mathcal{K}) = exp(-\kappa||\mathcal{C}(\Sigma_\mathcal{S}) - \mathcal{C}(\Sigma_\mathcal{T})||^2_F), \quad (13)$$

The categorical spherical manifold Gaussian kernel geodesic loss is defined as:

$$\mathcal{L}^c_\mathcal{K} = ||\,\mathrm{Log}_{\phi_{Proj.}(\mathcal{C}(B_\mathcal{S}) \times \mathcal{C}(\mathcal{K}))}(\phi_{Proj.}(\mathcal{C}(B_\mathcal{T}) \times \mathcal{C}(\mathcal{K})))||^2_F. \quad (14)$$

During the training, minimizing $\mathcal{L}^c_\mathcal{K}$ can lead to categorical features between source and target domains to be close to each other. We hence can align the conditional distribution between two domains.

3.7. DSGK model

The framework of our proposed DSGK model is depicted in Fig. 2. Taken altogether, our model minimizes the following objective function:

$$\arg\min\,(\mathcal{L}_\mathcal{S} + \mathcal{L}_\mathcal{T} + \alpha \mathcal{L}_\mathcal{K} + \beta \frac{1}{C} \sum_{c=1}^{C} \mathcal{L}^c_\mathcal{K}) \quad (15)$$

where $\mathcal{L}_\mathcal{S}$ is the source classification loss and $\mathcal{L}_\mathcal{K}$ minimizes marginal discrepancy between two domains. $\mathcal{L}_\mathcal{T}$ is the pseudo labeled target domain classification loss, and $\mathcal{L}^c_\mathcal{K}$ is the loss function reducing the conditional categorical discrepancy between two domains. $\mathcal{L}_\mathcal{T}$ is equally important as $\mathcal{L}_\mathcal{S}$ since we treat pseudo labels as real target labels. The overall training algorithm is shown in Alg. 1.

Algorithm 1 Deep Spherical Manifold Gaussian Kernel Network. $B_\mathcal{S}$ and $B_\mathcal{T}$ denote the mini-batch training sets, I is number of iterations. T is the number refinement steps.

```
Input: labeled source samples $\mathcal{D}_\mathcal{S} = \{\mathcal{X}^i_\mathcal{S}, \mathcal{Y}^i_\mathcal{S}\}_{i=1}^{\mathcal{N}_\mathcal{S}}$ and
    unlabeled target samples $\mathcal{D}_\mathcal{T} = \{\mathcal{X}^j_\mathcal{T}\}_{j=1}^{\mathcal{N}_\mathcal{T}}$
Output: predicted target domain labels
repeat
  Derive $B_\mathcal{S}$ and $B_\mathcal{T}$ sampled from $\mathcal{D}_\mathcal{S}$ and $\mathcal{D}_\mathcal{T}$
  Initialize $\Phi$ and $G$ using Eqs. 3 and 8, output: $G$
  for $t = 1$ to $T$ do
    for $i = 1$ to $I$ do
      Generate pseudo-labels $\mathcal{Y}_{\mathcal{T}_\mathcal{P}}$ using $G$
      Derive $\hat{B_\mathcal{T}}$ sampled from pseudo-labeled
          $\mathcal{D}_{\mathcal{PT}} = \{\Phi(\mathcal{X}^j_{\mathcal{PT}}), \mathcal{Y}^j_{\mathcal{PT}}\}_{j=1}^{\mathcal{N}_{\mathcal{PT}}}$
      Perform conditional distribution alignment using
          Eqs. 10 and 14
      Refine $G$ using Eq. 15
    end for
  end for
until converged
```

3.8. Theoretical Analysis

In this section, we theoretically show the error bound of the target domain for our proposed DSGK model with the domain adaptation theory [1] in Theorem 1.

Theorem 1 *Let $\mathcal{H}$ be a hypothesis space. Given two domains $\mathcal{D}_\mathcal{S}$ and $\mathcal{D}_\mathcal{T}$, we have*

$$\forall h \in \mathcal{H},\ R_\mathcal{T}(h) \leq R_\mathcal{S}(h) + d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_\mathcal{S}, \mathcal{D}_\mathcal{T}) + \beta,$$

where $R_\mathcal{S}(h)$ and $R_\mathcal{T}(h)$ represent the source and target domain risk, respectively. $d_{\mathcal{H}\Delta\mathcal{H}}$ is the discrepancy distance between two distributions $\mathcal{D}_\mathcal{S}$ and $\mathcal{D}_\mathcal{T}$ (including both marginal and conditional distributions). $\beta = \arg\min_{h\in\mathcal{H}} R_\mathcal{S}(h^*, f_\mathcal{S}) + R_\mathcal{T}(h^*, f_\mathcal{T})$ where $f_\mathcal{S}$ and $f_\mathcal{T}$ are the label functions of two domains, which can be determined by $\mathcal{Y}_\mathcal{S}$ and pseudo target domain labels. h^* is the ideal hypothesis and β is the shared error and is expected to be negligibly small and can be disregarded.

In our DSGK model, the first term $R_\mathcal{S}(h)$ can be small by training the labeled source domain in Eq. 3. During the training, the domain discrepancy distance $d_{\mathcal{H}\Delta\mathcal{H}}$ can be minimized by reducing the divergence between the marginal and target distributions of latent feature space of two domains. Specifically, $d_{\mathcal{H}\Delta\mathcal{H}} \approx \mathcal{L}_\mathcal{K} + \frac{1}{C}\sum_{c=1}^{C}\mathcal{L}^c_\mathcal{K}$. Ideally, the domain discrepancy distance will be perfectly removed if $\mathcal{L}_\mathcal{K} + \frac{1}{C}\sum_{c=1}^{C}\mathcal{L}^c_\mathcal{K}$ is close to 0. However, it can be achieved if and only if $\mathcal{D}_\mathcal{S} = \mathcal{D}_\mathcal{T}$. Therefore, minimizing $d_{\mathcal{H}\Delta\mathcal{H}}$ is equivalent to minimizing $\mathcal{L}_\mathcal{K} + \frac{1}{C}\sum_{c=1}^{C}\mathcal{L}^c_\mathcal{K}$.

4. Experiments

4.1. Setup

Datasets. We test our model using three image datasets: Office-31, Office-Home, and VisDA-2017. **Office-31** [21] has 4,110 images from three domains: Amazon (A), Webcam (W), and DSLR (D) in 31 classes. In experiments, A→W represents transferring knowledge from domain A to domain W. **Office-Home** [28] dataset contains 15,588 images from four domains: Art (Ar), Clipart (Cl), Product (Pr), and Real-World (Rw) in 65 classes. Fig. 3 shows some example images of four domains. **VisDA-2017** [20] is a challenging dataset due to the big domain-shift between the synthetic images (152,397 images from VisDA) and the real images (55,388 images from COCO) in 12 classes. We test our model on the setting of synthetic-to-real as the source-to-target domain and report the accuracy of each category.

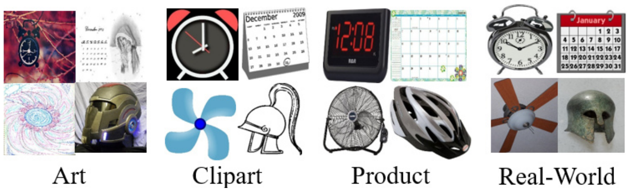


Figure 3: Sample images from four domains of the Office-Home dataset. We only show images from four categories.

Implementation details. We implement our approach using PyTorch and extract features for the three datasets from finely tuned ResNet50 (Office-31, Office-Home) and ResNet101 (VisDA-2017) networks [7]. The 1,000 features are then extracted from the last fully connected layer for the source and target features. The output of three Linear layers are 512, 256 and $|C|$, respectively. Parameters in recurrent pseudo labeling are $T = 5$ and $P_\mathcal{T} = [0.9, 0.8, 0.7, 0.6, 0.5]$. Learning rate ($\epsilon = 0.001$), batch size (64), $\kappa = 0.1$, $\alpha = 0.1$, $\beta = 0.01$ and number of epochs (9) are determined by performance on the source domain. We compare our results with 20 state-of-the-art methods[1]. Experiments are performed with a GeForce 1080 Ti.

4.2. Results

The performance on Office-Home, VisDA-2017, and Office-31 are shown in Tables 1-3. Our DSGK model outperforms all state-of-the-art methods in terms of average

[1] Source code is available at: https://github.com/YoushanZhang/Transfer-Learning/tree/main/Code/Deep/DSGK.

Table 1: Accuracy (%) on Office-Home dataset (based on ResNet50)

Task	Ar→Cl	Ar→Pr	Ar→Rw	Cl→Ar	Cl→Pr	Cl→Rw	Pr→Ar	Pr→Cl	Pr→Rw	Rw→Ar	Rw→Cl	Rw→Pr	**Ave.**
ResNet-50 [7]	34.9	50.0	58.0	37.4	41.9	46.2	38.5	31.2	60.4	53.9	41.2	59.9	46.1
DAN [12]	43.6	57.0	67.9	45.8	56.5	60.4	44.0	43.6	67.7	63.1	51.5	74.3	56.3
DANN [3]	45.6	59.3	70.1	47.0	58.5	60.9	46.1	43.7	68.5	63.2	51.8	76.8	57.6
JAN [15]	45.9	61.2	68.9	50.4	59.7	61.0	45.8	43.4	70.3	63.9	52.4	76.8	58.3
CDAN-M [13]	50.6	65.9	73.4	55.7	62.7	64.2	51.8	49.1	74.5	68.2	56.9	80.7	62.8
TAT [11]	51.6	69.5	75.4	59.4	69.5	68.6	59.5	50.5	76.8	70.9	56.6	81.6	65.8
ETD [9]	51.3	71.9	85.7	57.6	69.2	73.7	57.8	51.2	79.3	70.2	57.5	82.1	67.3
TADA [30]	53.1	72.3	77.2	59.1	71.2	72.1	59.7	53.1	78.4	72.4	**60.0**	82.9	67.6
SymNets [38]	47.7	72.9	78.5	64.2	71.3	74.2	64.2	48.8	79.5	74.5	52.6	82.7	67.6
DMP [17]	52.3	73.0	77.3	64.3	72.0	71.8	63.6	52.7	78.5	72.0	57.7	81.6	68.1
DCAN [10]	54.5	75.7	81.2	67.4	74.0	76.3	67.4	52.7	80.6	74.1	59.1	83.5	70.5
DSGK	**55.9**	**78.4**	**81.3**	**69.1**	**81.9**	**80.2**	**70.1**	**55.7**	**82.1**	**75.1**	58.4	**84.9**	**72.8**

Table 2: Accuracy (%) on VisDA-2017 dataset (based on ResNet101)

Task	plane	bcycl	bus	car	horse	knife	mcycl	person	plant	sktbrd	train	truck	**Ave.**
Source-only [7]	55.1	53.3	61.9	59.1	80.6	17.9	79.7	31.2	81.0	26.5	73.5	8.5	52.4
DANN [3]	81.9	77.7	82.8	44.3	81.2	29.5	65.1	28.6	51.9	54.6	82.8	7.8	57.4
DAN [12]	87.1	63.0	76.5	42.0	90.3	42.9	85.9	53.1	49.7	36.3	85.8	20.7	61.1
JAN [15]	75.7	18.7	82.3	86.3	70.2	56.9	80.5	53.8	92.5	32.2	84.5	54.5	65.7
MCD [23]	87.0	60.9	83.7	64.0	88.9	79.6	84.7	76.9	88.6	40.3	83.0	25.8	71.9
DMP [17]	92.1	75.0	78.9	75.5	91.2	81.9	89.0	77.2	93.3	77.4	84.8	35.1	79.3
DADA [25]	92.9	74.2	82.5	65.0	90.9	93.8	87.2	74.2	89.9	71.5	86.5	48.7	79.8
STAR [16]	95.0	84.0	84.6	73.0	91.6	91.8	85.9	78.4	94.4	84.7	87.0	42.2	82.7
DSGK	**95.7**	**86.3**	**85.8**	**77.3**	**92.3**	**94.9**	**88.5**	**82.9**	**94.9**	**86.5**	**88.1**	**46.8**	**85.0**

Table 3: Accuracy (%) on Office-31 dataset (based on ResNet50)

Task	A→W	A→D	W→A	W→D	D→A	D→W	**Ave.**
ResNet50 [7]	68.4	68.9	60.7	99.3	62.5	96.7	76.1
RTN [14]	84.5	77.5	64.8	99.4	66.2	96.8	81.6
ADDA [26]	86.2	77.8	68.9	98.4	69.5	96.2	82.9
GSM [39]	84.8	82.7	73.5	96.6	70.9	95.0	83.9
JAN [15]	85.4	84.7	70.0	99.8	68.6	97.4	84.3
ETD [9]	92.1	88.0	67.8	**100**	71.0	**100**	86.2
DMP [17]	93.0	91.0	70.2	**100**	71.4	99.0	87.4
TADA [30]	94.3	91.6	73.0	99.8	72.9	98.7	88.4
SymNets [38]	90.8	93.9	72.5	**100**	74.6	98.8	88.4
TAT [11]	92.5	93.2	73.1	**100**	73.1	99.3	88.5
MDA [35]	94.0	92.6	77.6	99.2	78.7	96.9	89.8
CAN [8]	94.5	95.0	77.0	99.8	78.0	99.1	90.6
DSGK	**95.7**	**96.9**	**80.3**	99.6	**80.3**	99.2	**92.0**

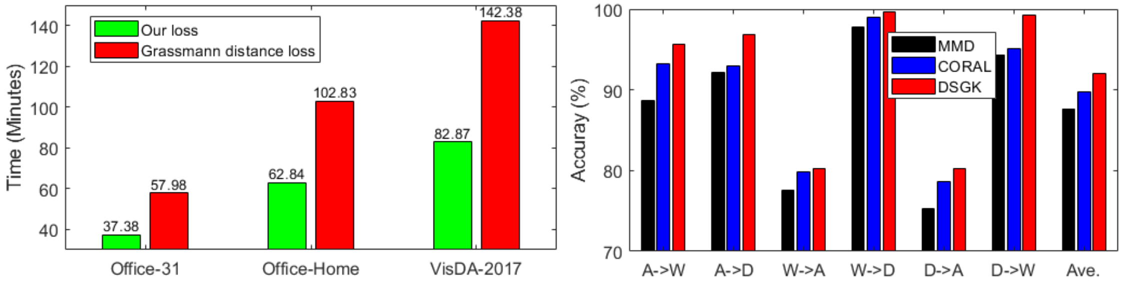
accuracy (especially in the VisDA-2017 and Office-Home datasets). The DSGK model substantially improves classification accuracy on difficult adaptation tasks (e.g., W→A task in the Office-31 dataset and the challenging VisDA-2017 and Office-Home datasets, which have a larger number of categories and different domains are visually dissimilar).

In the Office-31 dataset, the mean accuracy is 92.0%, compared with the best baseline (CAN), our DSGK model provides a 1.4% improvement. Although the improvement is not significant, we have an obvious improvement in some difficult tasks (W→A and D→A). The mean accuracy on the Office-Home dataset is increased from 70.5% (DCAN) to 72.8%. We also notice that accuracy is obviously improved across all tasks except Pr→Cl. In the VisDA-2017 dataset, the DSGK model has a 2.3% improvement over the best baseline (STAR), and it achieves the highest performance in all tasks. Therefore, our proposed spherical manifold Gaussian kernel loss is useful, and the easy-to-hard refinement process is effective in improving the classification accuracy.

In addition, we compare the computation time of our proposed DSGK model with Grassmann distance in DMP, which relies on the SVD of the covariance matrices in Fig. 4a. DSGK model requires relatively less computation time for all three datasets. Our loss functions are (1.5, 1.6, and 1.7 times) faster than Grassmann distance loss in the

Table 4: Ablation experiments on Office-31 dataset

Task	A→W	A→D	W→A	W→D	D→A	D→W	**Ave.**
DSGK−K/T/C	85.2	89.2	75.0	97.6	75.4	94.7	86.2
DSGK−C/T	85.9	90.1	75.2	98.0	76.0	96.1	86.9
DSGK−K/T	91.6	90.7	78.0	98.1	77.8	96.3	88.8
DSGK−K/T	93.4	91.6	78.0	98.3	78.3	96.9	89.4
DSGK−C	94.5	94.3	78.7	99.0	78.6	97.0	90.4
DSGK−T	94.9	95.3	79.0	99.0	79.0	97.4	90.8
DSGK−K	95.3	96.4	79.5	99.0	79.2	98.0	91.2
DSGK	**95.7**	**96.9**	**80.3**	**99.6**	**80.3**	**99.2**	**92.0**



(a) Time of three datasets

(b) Accuracy comparison of different loss functions

Figure 4: Computation time and different loss functions comparison. (a) is the total computation time of all transfer tasks in three datasets (six for Office-31, twelve for Office-Home and one for VisDA-2017 dataset). On average, our DSGK model is approximately 1.6 times faster than Grassmann distance loss. (b) compares the DSGK model with the other two loss functions in each task of Office-31. Our loss function achieves a higher accuracy than the other two.

three datasets since Grassmann distance loss requires the calculation-intensive SVD. Therefore, our proposed loss function is much faster than Grassmann distance loss. To show the effectiveness of the proposed spherical manifold Gaussian kernel geodesic loss, we also compare the results of well-used loss functions: CORAL loss [24] and MMD [12] loss; that is, we replace Eq. 8 and Eq. 14 with these two loss functions. As shown in Fig. 4, our proposed loss achieves higher accuracy than CORAL and MMD loss functions. Therefore, our DSGK model is effective and accurate in UDA tasks.

4.3. Ablation study

To demonstrate the effects of different loss functions on final classification accuracy, we present an ablation study in Tab. 4, in which K represents spherical manifold Gaussian kernel geodesic loss $\mathcal{L}_{\mathcal{K}}$, T is the $\mathcal{L}_{\mathcal{T}}$ and C denotes the $\mathcal{L}_{\mathcal{K}}^c$. “DSGK−K/T/C” is implemented without $\mathcal{L}_{\mathcal{K}}$, $\mathcal{L}_{\mathcal{T}}$, and $\mathcal{L}_{\mathcal{K}}^c$. It is a simple model, which only reduces the source risk without minimizing the domain discrepancy using $\mathcal{L}_{\mathcal{S}}$. “DSGK−C/T” only aligns the marginal distribution between two domains. “DSGK−C” reports results without performing the additional categorical conditional distribution alignment. We observe that with the increasing number of loss functions, the robustness of our model keeps improving. The usefulness of loss functions is ordered as $\mathcal{L}_{\mathcal{K}} < \mathcal{L}_{\mathcal{T}} < \mathcal{L}_{\mathcal{K}}^c$. Therefore, the proposed spherical manifold Gaussian kernel geodesic loss and easy-to-hard learning approach are effective in improving performance, and different loss functions are helpful and important in minimizing target domain risk.

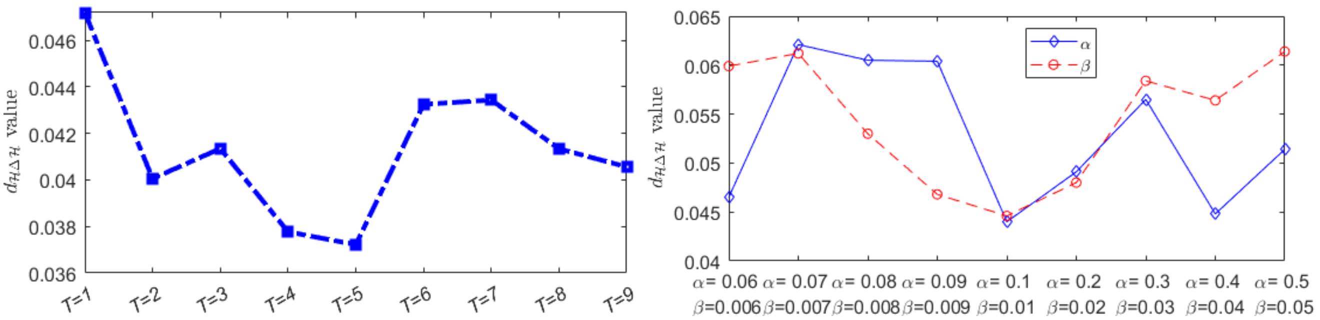
4.4. Parameter Analysis

There are four hyperparameters T, P_t, α and β in DSGK that can affect the final accuracy. To get the optimal parameters, we randomly select the task W→A in Office-31 dataset and run a set of experiments regarding different values of each parameter. Notice that it is inappropriate to tune parameters using the target domain accuracy since we do not have any labels in the target domain. Therefore, we report the $\mathcal{H}$ divergence between two domains, as stated in Sec. 3.8. Since $\mathcal{H}$ divergence can be assessed by $d_{\mathcal{H}\Delta\mathcal{H}} \approx \mathcal{L}_{\mathcal{K}} + \frac{1}{C}\sum_{c=1}^{C}\mathcal{L}_{\mathcal{K}}^c$, we can calculate $\mathcal{L}_{\mathcal{K}} + \frac{1}{C}\sum_{c=1}^{C}\mathcal{L}_{\mathcal{K}}^c$ and select the parameters if they achieve the minimal value. α is selected from $\{0.06, 0.07, 0.08, 0.09, 0.1, 0.2, 0.3, 0.4, 0.5\}$, β is selected from $\{0.006, 0.007, 0.008, 0.009, 0.01, 0.02, 0.03, 0.04, 0.05\}$, T is selected from $\{1, 2, 3, 4, 5, 6, 7, 8, 9\}$, and P_t is selected from $\{0.9, 0.8, 0.7, 0.6, 0.5, 0.5, 0.3, 0.2, 0.1\}$, we vary one parameter and fix the others at a time. For T and P_t, if $T = 9$, then $P_t = [0.9, 0.8, 0.7, 0.6, 0.5, 0.5, 0.3, 0.2, 0.1]$ and if $T = 1$, then $P_t = [0.1]$.

Fig. 5 demonstrates that our DSGK model is not very sensitive to a wide range of parameter values since the $\mathcal{H}$ divergence ($d_{\mathcal{H}\Delta\mathcal{H}}$) is not significantly changed. We first determine the T and P_t as shown in Fig. 5a. We can find that when $T = 5$, that is $P_t = [0.9, 0.8, 0.7, 0.6, 0.5]$, $d_{\mathcal{H}\Delta\mathcal{H}}$ achieves the minimum value. Hence, $P_t = [0.9, 0.8, 0.7, 0.6, 0.5]$ is the best parameter for our DSGK model. In Fig. 5, a large T can have a larger $d_{\mathcal{H}\Delta\mathcal{H}}$ (e.g., $T = 6$ in Fig. 5a and $T = 7$ in Fig. 5a) since a larger T brings hard examples into the shared classifier G, which leads to lower perforamce in the target domain. Therefore, the parameter analysis is useful in finding the best hyperparameters for our DSGK model. After fixing T and P_t, in Fig. 5b, we combine the parameter tuning results for α and β together. If it is α, then the x-axis is from 0.1 to 0.9, and if it is β, then the x-axis is from 0.01 to 0.09. It shows that when $\alpha = 0.1$ and $\beta = 0.01$ achieves the minimum number. Therefore, the hyperparameter $\alpha = 0.1$ and $\beta = 0.01$ is the best since the discrepancy between two domains is minimized.

4.5. Feature Visualization

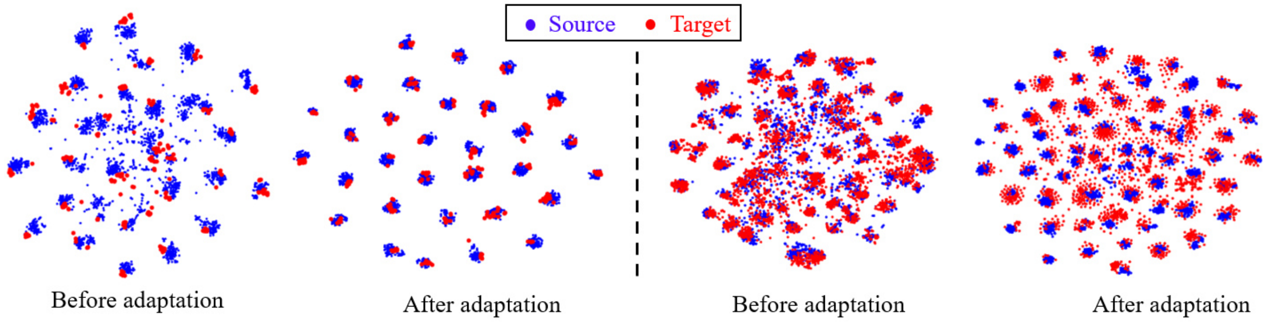
To intuitively present adaptation performance during the transition from the source domain to the target domain, we utilize-SNE to visualize the deep features of network activations in 2D space before and after distribution adaptation.



(a) Effect of different T on $d_{\mathcal{H}\Delta\mathcal{H}}$

(b) Effect of different α and β on $d_{\mathcal{H}\Delta\mathcal{H}}$

Figure 5: Parameter analysis. In (a), $d_{\mathcal{H}\Delta\mathcal{H}}$ is minimum when $T = 5$. In (b), the x-axis denotes different α and β, $d_{\mathcal{H}\Delta\mathcal{H}}$ is minimum when $\alpha = 0.1$ and $\beta = 0.01$.



(a) Office-31 (A→W)

(b) Office-Home (Pr→ Ar)

Figure 6: Feature visualization using a 2D t-SNE view of task A→W in Office-31 dataset and Pr→Ar in Office-Home dataset. Our DSGK model improves the discriminative representations across domains. (blue color: source domain, red color: target domain). Best viewed in color.

Fig. 6 visualizes embeddings of the task A→W in the Office-31 dataset and Pr→Ar in the Office-Home dataset. We can observe that the representation becomes more discriminative after adaptation, while many categories are mixed in the feature space before adaptation. Therefore, DSGK can learn more discriminative representations, which can significantly increase inter-class dispersion and intra-class compactness.

5. Discussion

In these experiments, DSGK always achieves the highest average accuracy. Therefore, the quality of our model exceeds that of SOTA methods and is better than existing loss functions. There are two compelling reasons. First, the proposed spherical manifold Gaussian kernel geodesic loss can project data into a spherical manifold and calculate the intrinsic distance between two domains. It not only avoids the complex SVD calculation to reduce computation time as in the Grassmannian manifold but also considers both the batch-wise features $B_{\mathcal{Z}}$ and the Gaussian kernel $\mathcal{K}$ between two domains. Secondly, to minimize the conditional distribution discrepancy, we develop an easy-to-hard refinement process by keeping reducing the predicted probability of the target domain. This strategy can push the shared classifier G towards the target domain. Hence, the easy-to-hard refinement process is useful in updating the network parameters, which further reduces the domain discrepancy.

One limitation of DSGK is the loss $\mathcal{L}_{\mathcal{K}}^{c}$ needs to calculate the difference between two domains of each category. If the number of categories ($|C|$) is substantially larger, it may need more computation time. However, it will still be faster than existing Grassmann distance loss function, as shown in Fig. 4a.

6. Conclusion

In this paper, we propose a novel deep spherical manifold Gaussian kernel (DSGK) model for unsupervised domain adaptation. To align the marginal distribution between the source and the target domain, we develop a spherical manifold Gaussian kernel geodesic loss to minimize the intrinsic distance between two domains. We also employ an easy-to-hard refinement process to remove the noisy pseudo labels and reduce the categorical spherical manifold Gaussian kernel geodesic loss to align the conditional distribution of two domains. Extensive experiments demonstrate that the proposed DSGK model achieves higher accuracy than state-of-the-art domain adaptation methods.

References

[1] S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira. Analysis of representations for domain adaptation. In *Advances in Neural Information Processing Systems*, pages 137–144, 2007. 5

[2] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030, 2016. 2

[3] M. Ghifary, W. B. Kleijn, and M. Zhang. Domain adaptive neural networks for object recognition. In *Proceedings of the Pacific Rim International Conference on Artificial Intelligence*, pages 898–904. Springer, 2014. 6

[4] B. Gong, Y. Shi, F. Sha, and K. Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2066–2073. IEEE, 2012. 1, 2

[5] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014. 2

[6] R. Gopalan, R. Li, and R. Chellappa. Domain adaptation for object recognition: An unsupervised approach. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 999–1006. IEEE, 2011. 1, 2

[7] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 5, 6

[8] G. Kang, L. Jiang, Y. Yang, and A. G. Hauptmann. Contrastive adaptation network for unsupervised domain adaptation. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4893–4902, 2019. 2, 6

[9] M. Li, Y. Zhai, Y. Luo, P. Ge, and C. Ren. Enhanced transport distance for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13936–13944, 2020. 2, 6

[10] S. Li, C. H. Liu, Q. Lin, B. Xie, Z. Ding, G. Huang, and J. Tang. Domain conditioned adaptation network. In *AAAI*, pages 11386–11393, 2020. 6

[11] H. Liu, M. Long, J. Wang, and M. Jordan. Transferable adversarial training: A general approach to adapting deep classifiers. In *International Conference on Machine Learning*, pages 4013–4022, 2019. 6

[12] M. Long, Y. Cao, J. Wang, and M. I. Jordan. Learning transferable features with deep adaptation networks. *arXiv preprint arXiv:1502.02791*, 2015. 6, 7

[13] M. Long, Z. Cao, J. Wang, and M. I. Jordan. Conditional adversarial domain adaptation. In *Advances in Neural Information Processing Systems*, pages 1647–1657, 2018. 6

[14] M. Long, H. Zhu, J. Wang, and M. I. Jordan. Unsupervised domain adaptation with residual transfer networks. In *Advances in Neural Information Processing Systems*, pages 136–144, 2016. 1, 6

[15] M. Long, H. Zhu, J. Wang, and M. I. Jordan. Deep transfer learning with joint adaptation networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pages 2208–2217. JMLR.org, 2017. 6

[16] Z. Lu, Y. Yang, X. Zhu, C. Liu, Y. Song, and T. Xiang. Stochastic classifiers for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9111–9120, 2020. 6

[17] Y. Luo, C. Ren, D. Dao-Qing, and H. Yan. Unsupervised domain adaptation via discriminative manifold propagation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. 1, 2, 6

[18] Z. Meng, J. Li, Y. Gong, and B. Juang. Adversarial teacher-student learning for unsupervised domain adaptation. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5949–5953. IEEE, 2018. 2

[19] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010. 1

[20] X. Peng, B. Usman, N. Kaushik, J. Hoffman, D. Wang, and K. Saenko. Visda: The visual domain adaptation challenge. *arXiv preprint arXiv:1710.06924*, 2017. 5

[21] K. Saenko, B. Kulis, M. Fritz, and T. Darrell. Adapting visual category models to new domains. In *Proceedings of the European Conference on Computer Vision*, pages 213–226. Springer, 2010. 5

[22] K. Saito, Y. Ushiku, and T. Harada. Asymmetric tri-training for unsupervised domain adaptation. *arXiv preprint arXiv:1702.08400*, 2017. 2

[23] K. Saito, K. Watanabe, Y. Ushiku, and T. Harada. Maximum classifier discrepancy for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3723–3732, 2018. 6

[24] B. Sun and K. Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *Proc. of European Conference on Computer Vision*, pages 443–450. Springer, 2016. 2, 3, 7

[25] H. Tang and K. Jia. Discriminative adversarial domain adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5940–5947, 2020. 6

[26] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7167–7176, 2017. 2, 6

[27] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474*, 2014. 2

[28] H. Venkateswara, J. Eusebio, S. Chakraborty, and S. Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5018–5027, 2017. 5

[29] J. Wang, W. Feng, Y. Chen, H. Yu, M. Huang, and P. S. Yu. Visual domain adaptation with manifold embedded distribution alignment. In *Proceedings of the 26th ACM International Conference on Multimedia*, MM '18, pages 402–410, 2018. 2

[30] X. Wang, L. Li, W. Ye, M. Long, and J. Wang. Transferable attention for domain adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5345–5352, 2019. 6

[31] R. C. Wilson and E. R. Hancock. Spherical embedding and classification. In *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*, pages 589–599. Springer, 2010. 3

[32] S. Xie, Z. Zheng, L. Chen, and C. Chen. Learning semantic representations for unsupervised domain adaptation. In *International Conference on Machine Learning*, pages 5423–5432, 2018. 1

[33] W. Zhang, W. Ouyang, W. Li, and D. Xu. Collaborative and adversarial network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3801–3809, 2018. 1, 2

[34] Y. Zhang. K-means principal geodesic analysis on riemannian manifolds. In *Proceedings of the Future Technologies Conference*, pages 578–589. Springer, 2019. 3

[35] Y. Zhang and B. D. Davison. Modified distribution alignment for domain adaptation with pre-trained Inception ResNet. *arXiv preprint arXiv:1904.02322*, 2019. 6

[36] Y. Zhang and B. D. Davison. Domain adaptation for object recognition using subspace sampling demons. *Multimedia Tools and Applications*, pages 1–20, 2020. 1

[37] Y. Zhang and B. D. Davison. Adversarial continuous learning in unsupervised domain adaptation. In *Pattern Recognition. ICPR International Workshops and Challenges: Virtual Event, January 10–15, 2021, Proceedings, Part II*, pages 672–687. Springer International Publishing, 2021. 2

[38] Y. Zhang, H. Tang, K. Jia, and M. Tan. Domain-symmetric networks for adversarial domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5031–5040, 2019. 6

[39] Y. Zhang, S. Xie, and B. D. Davison. Transductive learning via improved geodesic sampling. In *Proceedings of the 30th British Machine Vision Conference*, 2019. 1, 2, 6

[40] Y. Zhang, J. Xing, and M. Zhang. Mixture probabilistic principal geodesic analysis. In *Multimodal Brain Image Analysis and Mathematical Foundations of Computational Anatomy*, pages 196–208. Springer, 2019. 3

[41] Y. Zhang, H. Ye, and B. D. Davison. Adversarial reinforcement learning for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 635–644, 2021. 2