Transductive Learning Via Improved Geodesic Sampling

Youshan Zhang
Sihong Xie
Brian D. Davison
{yoz217,six316,bdd3}@lehigh.edu

Computer Science and Engineering
Lehigh University
Bethlehem, PA, USA

Abstract

Transductive learning exploits the connection between training and test data to improve classification performance, and the geometry of the manifold underlying the training and the test data is essential to make this connection explicit. Existing approaches primarily focus on Grassmannian manifolds, while much less is known regarding other manifolds, which can potentially bring increased computational and learning performance. In this paper, we close the gap and formulate a novel and more general geodesic sampling approach on Riemannian manifolds (GSM) that encompasses Sphere, Kendall, and Grassmannian manifolds. To provide practical guidance for classification, we explore extensive hyperparameter settings and baselines, including deep transfer learning models. The results show that the new method can enable more accurate and less computationally expensive geodesic sampling on the sphere manifold, which is not possible to achieve using the existing Grassmannian manifold.

1 Introduction

Transductive learning takes advantage of unlabeled test data during training and can outperform inductive learning that has to train a model without test data. Prior transductive learning approaches need to make a strong assumption that the distributions of training and test data are the same. On one hand, traditional transductive learning paradigms assume that the two distributions are the same, so that information embedded in the test data will be useful in augmenting the training data to aid the learning task. For example, transductive SVM and semi-supervised learning on graphs exploit the data density estimated from a large amount of test data to train a more accurate SVM [9, 12]. On the other hand, it is also assumed that test data can be different from training data in their posteriors $p(y|\mathbf{x})$ or marginals $p(\mathbf{x})$. For example, to learn a latent feature space in which both domains have similar distributions, structural correspondence learning approaches have been proposed [4, 6, 7, 8, 37]. For semi-supervised domain adaptation, Daume and Marcu [14] designed a model to find similar data distributions of both source and target domains. Subsequent work includes approaches that combine co-training and domain adaptation using labels from either domain [15], semi-supervised learning based on EM algorithms [13], co-regularization [29] and projecting data to kernel Hilbert space [38]. There has also been theoretical work addressing the classification error across domains [3].

We focus on transductive learning using manifolds that smoothly connect the potentially different geometries of the training and test data to facilitate transductive transfer learning. Simard et al. [42] proposed tangent distance and tangent propagation algorithms using a Lie group (a special manifold) for image recognition. Gopalan et al. [22] proposed sampling geodesic flow (SGF) to learn the intermediate features between the source and the target domain via Grassmannian manifolds, and then using all intermediate feature vectors to train the classifiers and evaluate on test data. (A geodesic is a curve representing the shortest path between two points on a manifold.) However, SGF has high time complexity making sampling slow when the sample size is large. Gong et al. [21] proposed geodesic flow kernel (GFK) method to address the limitations in SGF. However, the reduced dimensionality of data is an important parameter of the GFK model; it needs to calculate the optimal dimensionality. In addition, it has the constraint that the size of dimensionality should be less than half of the minimum dimension of training and testing data, which is $d < \frac{1}{2}\min(l(\text{train}), l(\text{test}))$, where l refers to the number of features. In addition, GFK works best only if the dimensionality is significantly larger than the sample size. Wang and Mahadevan [49] aligned the source and target domains by preserving the 'neighborhood structure' of the data points. Wang et al. proposed a manifold embedding distribution alignment method to align both the degenerate feature transformation and the unknown distribution of both domains [51]. However, all prior manifold-based approaches only focus on Grassmannian manifolds and cannot be generalized to other manifolds that are potentially useful for transductive learning.

To address these challenges, we propose an improved Geodesic Sampling model (GSM) to generalize prior work to include Riemannian manifolds that can be potentially more useful for transductive learning. More specifically, we present geodesic sampling formulations and algorithms to apply GSM to sphere and Kendall's shape manifolds. For highest absolute performance, we also create two updated datasets (Office + Caltech 10, Office 31 and Office-Home) for domain adaptation whose features are extracted using a pre-trained Xception deep neural network [11]. Our empirical results find that the much simpler sphere manifold often works better than the more commonly adopted Grassmannian manifolds and encourages more efficient computation. Extensive experiments demonstrate significant improvements in classification accuracy over state-of-the-art methods.

2 Background

Existing manifold-based methods only learn features between the source and target domains on Grassmannian manifolds [21, 22]. They also fail to generate correct samples along their "geodesic path" which is from the source to the target domain. Classification accuracy is also not high using these incorrect samples. In contrast, we formulate a generalized model that can be applied in any Riemannian manifold, and which generates sample intermediate points that are along the true geodesic path. Specifically, we learn the geodesic between two domains by using the geometry of data without making any assumptions about the domain shift transformation. Description of the problem is given in the following section.

Here we discuss the general sampling problem on manifolds and then apply sampling strategies to the problem of classification. Given training data (source data): $X'_S = \{x_i\}_{i=1}^{N_1}$ (has the size of $N_1 \times d$, where d is the number of features of the source data), with the labels $Y_S = \{y_i\}_{i=1}^{N_1}, y_i \in \{1,2,3,\cdots,C\}, \forall i$ denoting one of C categories, and the test data (target data): $X'_T = \{x_j\}_{j=1}^{N_2}$ (has the size of $N_2 \times d$), with the labels $Y_t = \{y_i\}_{i=1}^{N_3}$ and $N_3 \leq N_2$, which means we might not have all labels for testing data. If $N_3 = N_2$, which means we have labels

for X'_T, then we want to build a model that can predict labels of X'_T with accuracy as high as possible. If $N_3 < N_2$, we not only want to get a high enough predictive accuracy, but also predict the labels for the unlabeled data. Typically, one constraint of sampling on manifolds is that it requires the same dimensions of X'_S and X'_T [21, 22]. While the number of features (d) of examples in X'_S and X'_T should be the same, the number of examples N_1 can be different from N_2, and usually $N_1 >> N_2$, if we want to get a high predictive accuracy on test data. Therefore, we need to construct a low-dimensional sub-space representation of the train and test data. There are several methods for constructing low-dimensional representations (e.g., principal component analysis (PCA) [26], Laplacian eigenmaps [2], and principal geodesic analysis (PGA) [17]). We use PCA to realize the dimensionality agreement in subspace. After computing the new subspace X_S and X_T representations of X'_S and X'_T, respectively, we have two questions to answer: (i) how to obtain correct and meaningful intermediate samples with $0 \leq t \leq 1$ from X_S to X_T (where t represents time); and, (ii) how to predict the test labels by using the samples of intermediate sub-spaces.

3 Our Approach: Geodesic Sampling on Manifold (GSM)

We first review the geometry of three manifolds (Sphere, Kendall's shape and Grassmannian manifold), and then present detailed algorithms to calculate samples.

3.1 Riemannian Geometry

In this section, we briefly review three basic concepts (geodesic, exponential and logarithmic map) of Riemannian geometry (while more details are provided by others [15, 18, 39]).

Geodesic. Let (M,g) be a Riemannian manifold, where g is a Riemannian metric on the manifold M. A curve $\gamma(t) : [0,1] \to M$ and let $\gamma'(t) = d\gamma/dt$ to be its velocity. The operation $D \cdot /dt$ is called a *covariant derivative* (also called a connection on M), which is denoted as $\nabla_{\gamma'(t)}$ or $\nabla_{\gamma'}$. A vector field $V(t)$ along γ is parallel if $\frac{DV(t)}{dt} = \nabla_{\gamma'} V = 0$. We call γ a geodesic if $\gamma'(t)$ is parallel along γ, that is: $\gamma'' = \frac{D\gamma'}{dt} = \nabla_{\gamma'}\gamma' = 0$, which means the acceleration vector (directional derivative) γ'' is normal to $T_{\gamma(t)}M$ (the tangent space of M at $\gamma(t)$). A geodesic is also a curve $\gamma \in M$ that locally minimizes $E(\gamma) = \int_0^1 ||\gamma'(t)||^2 dt$. Here $||\cdot||$ is called *Riemannian norm*, for any points $X_S \in M$, and $v \in T_{X_S}M$, $||v||$ is defined by: $||v|| = \sqrt{g_{X_S}(v,v)}$. $g_{X_S}(u,v)$ is called *Riemannian inner product* of two tangent vectors $u, v \in T_{X_S}M$, which can also be denoted by $\langle u,v\rangle_{X_S}$ or simply $\langle u,v\rangle$. The norm of velocity in a geodesic γ is constant, that is: $||\gamma'(t)|| = c$ [18]. In Fig. 1, the red curve is the geodesic given the base point X_S and the initial velocity v. Note that geodesics are straight lines in Euclidean space ($\mathbb{R}^n$).

Exponential Map. For any point $X_S \in M$ and its tangent vector v, let $\mathcal{D}(X_S)$ be the open subset of $T_{X_S}M$ defined by: $\mathcal{D}(X_S) = \{v \in T_{X_S}M | \gamma(1) \text{ is defined}\}$, where γ is the unique geodesic with initial conditions $\gamma(0) = X_S$ and $\gamma'(0) = v$. The *exponential map* is the map $\mathrm{Exp}_{X_S} : \mathcal{D}(X_S) \to M$ defined by: $\mathrm{Exp}_{X_S}(vt_{=1}) = \gamma(1)$, which means the exponential map returns the points at $\gamma(1)$ when $t = 1$. If $\omega \in \mathcal{D}(X_S)$, then the line segment $\{t\omega | 0 \leq t \leq 1\}$ is constrained to be in $\mathcal{D}(X_S)$. We then define: $\mathrm{Exp}_{X_S}(vt) = \gamma(t)$, where $0 \leq t \leq 1$.

Intuitively, Exp generates points on the geodesic as a function of the starting point, tangent vector, and t. Fig. 1 calculates the geodesic given the base point X_S, and initial velocity

v. For reference, in Euclidean space the exponential map is the addition operation ($\mathbb{R}^n$): $\text{Exp}_{X_S}(vt) = X_S + vt$ [18, 39].

Logarithmic Map. Given two points X_S and $X_T \in M$, the *logarithmic map* takes the point pair (X_S, X_T) and maps them into the tangent space $T_{X_S}M$, and it is an inverse of the exponential map: $\text{Log}(X_S, X_T) \rightarrow T_{X_S}M$. $\text{Log}(X_S, X_T)$ can also be denoted as: $\text{Log}_{X_S} X_T$. Because Log is an inverse of the exponential map, we can also write: $X_T = \text{Exp}(X_S, \text{Log}(X_S, X_T))$. The *Riemannian distance* is defined as $d(X_S, X_T) = ||\text{Log}_{X_S}(X_T)||$. In Euclidean space ($\mathbb{R}^n$), the logarithmic map is the subtraction operation: $\text{Log}_{X_S}(X_T) = X_T - X_S$ [53].

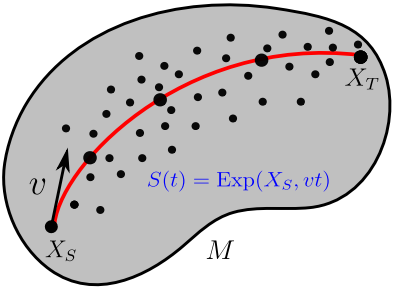

Figure 1: Illustration of the Exp map of our model. X_S is a base point of tangent space, v is the velocity at point X_S, t is the time, the red curve is the geodesic line with the increase of t giving the base points X_S and v. X_S is the sample at $t = 0$, and X_T is the sample at $t = 1$. S_t are sampled points (black points on the red curve) on the geodesic.

3.2 Methods

Given source data X_S, and target data X_T, we could get samples S between X_S and X_T. Eq. 1 is our generalized GSM model to sample points on Riemannian manifolds, which, unlike prior work, is applicable to any manifold.

$$S_t = \text{Exp}(X_S, v \times t), \tag{1}$$

where $0 \leq t \leq 1$, with $S_0 = X_S$ and $S_1 = X_T$, and velocity (v) is calculated from X_S to X_T by the Log map, which is defined as $v = \text{Log}(X_S, X_T)$.

In the following sub-sections, we give formulations of Log and Exp map to calculate the samples on three manifolds. After calculating the Log map and the Exp map on each manifold, we use Alg. 1 to sample points between X_S and X_T.

Algorithm 1 Geodesic Sampling on Riemannian Manifolds

Input: X_S, X_T, Sample size: N
1: Calculate v according to $v = \text{Log}(X_S, X_T)$
2: **For** $t = 0 : (1/(N-1)) : 1$
3: Calculate S_t according to Eq. (1)
4: **end**

3.3 Sphere manifold

One of the well-known spherical manifolds is the 3D sphere (2D surface embedding in 3D space). Let r be the radius of the sphere, u the azimuth angle and v the zenith angle. Then any points on the 3D sphere can be expressed by: $X = (r\sin(u)\sin(v), r\cos(u)\sin(v), r\cos(v))$. The generalized $n-1$ dimensional hyper-sphere embedded in $\mathbb{R}^n$ Euclidean space $(x_1, x_2, \cdots, x_n)$ has the constraint of: $\sum_{i=1}^{n} x_i^2 = r^2$, where r is the radius of such a hyper-sphere (we set $r = 1$). Let X_S and X_T be such points on on a sphere embedded in $\mathbb{R}^n$, and let v be a tangent vector at X_S.

The Log map between X_S and X_T can be computed as follows:

$$v = \text{Log}(X_S, X_T) = \frac{\theta \cdot L}{||L||}, \quad \theta = \arccos(\langle X_S, X_T \rangle), \quad L = (X_T - X_S \cdot \langle X_S, X_T \rangle) \tag{2}$$

where $X_S \cdot \langle X_S, X_T \rangle$ denotes the projection of the vector X_T onto X_S. $||L||$ is the *Riemannian norm* as defined in Section 3.1.

Given base point X_S, and its estimated tangent vector v from Eq. 2 and t, we can compute the Exp map as:

$$\text{Exp}(X_S, vt) = \cos\theta \cdot X_S + \frac{\sin\theta}{\theta} \cdot vt,\ \theta = ||vt||. \tag{3}$$

This explanation is based on that of Wilson and Hancock [52], where additional details of Log map and Exp map on the Sphere manifold can be found.

3.4 Kendall's shape manifold

A more complex manifold was studied by David G. Kendall [27]. Kendall's shape can provide a geometric setting for analyzing arbitrary sets of landmarks. The landmark points in Kendall's space are a collection of points in Euclidean space. Before calculating the Log and Exp maps, we use some transformations (scale and rotation) to get the pre-shape space representation of data.

Given two shapes $X_S, X_T \in V$ with $d \times n$ matrix (d is the dimension of the shape, n is the number of points), to construct Kendall's shape space, first, we remove the translation and scale of the shape. To get the prep-shape space representation of an object, we eliminate the centroid (subtract the row means from each row) and scale it into unit norm (divide by the Frobenius norm). Then, we remove the rotation of the shape using Orthogonal Procrustes Analysis (OPA) [23]. OPA solves the problem of finding the rotation R^* that can minimize the distance between X_S and X_T: $R^* = \arg\min_{R \in SO(d)} ||RX_S - X_T||$, where SO means a special orthogonal group. OPA performs singular value decomposition of $X_S \cdot X_T^T$; let $[U,S,V] = \text{SVD}(X_S \cdot X_T^T)$, then $R^* = UV^T$. Similar to the sphere manifold, the Log map between two shapes X_S, X_T of Kendall's shape manifold is given by finding the rotation between X_S and X_T first. To find the rotation of X_T, we calculate the singular value decomposition of $X_S \cdot X_T^T$; then we find the rotation of X_T by $X_{T(Rot)} = R^* \cdot X_T$. Then the Log map is given by:

$$v = \text{Log}(X_S, X_T) = \frac{\theta \cdot L}{||L||}, \quad \theta = \arccos(\langle X_S, X_{T(Rot)} \rangle),\ L = (X_{T(Rot)} - X_S \cdot \langle X_S, X_{T(Rot)} \rangle), \tag{4}$$

where $X_S \cdot \langle X_S, X_{T(Rot)} \rangle$ denotes the projection of the vector $X_{T(Rot)}$ onto X_S.

The Exp map on Kendall's shape manifold is the same as the sphere manifold (Eq. 3). Please refer to Kendall [27] to see additional details of Log map and Exp map on Kendall's shape manifold.

3.5 Grassmannian manifold

Before we introduce the Exp and Log map of the Grassmannian manifold, we should get the subspace of the data, since the Grassmannian manifold $\mathbb{G}_{N,d}$ is defined as a $d-$dimension subspace. Therefore, it is necessary to use PCA to get the subspace (submanifold) of Grassmannian manifold. Please refer to Fletcher et al. [17] for more details.

Suppose that we get the subspaces from PCA of the data (that is X_S and X_T) and they have the same dimensionality. Again to get the sampling from X_S to X_T, we need to calculate the Log map and Exp map. Please refer to Edelman et al. [16] and Absil et al. [1] to see the details of geometry on the Grassmannian manifold. Unlike Gallivan et al. [19] in which the calculation of Exp and Log map is complex and time-consuming (because of additional QR

(a) Progression of samples using GSM (b) Progression of samples using SGF

Figure 2: The comparison of sampling results between the two images (square and circle) with $t = 0, 0.05, 0.5, 0.95, 1$. Obviously, the SGF model does not generate a correct sample in (b), but our GSM model can create the correct sample in (a). For reference, the source image is the far left at $t = 0$ in Fig. 2(a), and the target image is the far right at $t = 1$ in Fig. 2(a).

decomposition), we use simpler calculations. The Log map is given by:

$$\begin{aligned} X &= (I - X_S \cdot X_S^T) \cdot X_T \cdot (X_S^T \cdot X_T)^{-1}, \;\; [U, s, V] = \text{SVD}(X), \\ v &= \text{Log}(X_S, X_T) = U \cdot \theta \cdot V^T, \;\; \theta = \arctan s, \end{aligned} \tag{5}$$

where I is the identity matrix. Here θ is the principal angle between subspace X_S and S_T.

The exponential map is computed from base point X_S and the estimated initial velocity v:

$$[U, \theta t, V] = \text{SVD}(vt), \;\; \text{Exp}(X_S, vt) = X_S \cdot V \cdot \cos(\theta t) + U \cdot \sin(\theta t) \cdot V^T, \tag{6}$$

However, one limitation of calculating Exp and Log map is that the number of columns should be minimized. If the column dimensionality is larger than one, the error will increase ($Error = norm(\text{Exp}(X_s, \text{Log}(X_s, X_t)) - X_t)$).[1] To address this concern, we reshape the data ($D \times N$) into a single column dimension ($(D \times N) \times 1$) since that will minimize the error.[2] After calculating Exp and Log map, we then reshape the data into $D \times N$ to reconstitute the original data dimensionality.

4 Experiments

We first show the disadvantages of the SGF model using an image sampling progression task, and then conduct extensive experiments in transductive learning for image recognition tasks.

4.1 Defects of the SGF model

Given two images, we calculate the progress of changing from source to target image. For example, as shown in Fig. 2(a), the source image is a square (the leftmost object of Fig. 2(a)) and the target image is a circle (the rightmost object of Fig. 2(a)); while the progress of sampled images of GSM and SGF model are shown in Figures 2(a) and 2(b). To evaluate the quality of samples, there are two criteria. The sample should be similar to the source image when $t = 0$, and the sample should be similar to the target image when $t = 1$. The sampled images of GSM model when $t = 0.05$ and $t = 0.95$ in Fig. 2(a) are almost the same as the true square and circle, respectively. However, the sample image of the SGF model is far from the source and target image when $t = 0.05$ and $t = 0.95$.

There are two issues in the SGF sampled target image: first, its background is dark; this is caused by the Log map not being correctly calculated in Gopalan et al. [22] (as there are

[1] See Fig. 1 in the supplementary file for a graph of this error.

[2] If original data size is 400×100, we reshape it into 40000×1, which can be easily changed using the *reshape* function in MATLAB.

Algorithm 2 Classification using GSM

Input: X_S, Y_S, X_T, Y_T, Sample size: N
Output: Accuracy of $predict_{Y_T}$

1: Generate sample (S_t) N times between X_S and X_T using Alg. 1.
2: $New\_X_S = X_S' \times [S_0, \cdots, S_t, \cdots, S_1]$
$New\_X_T = X_T' \times [S_0, \cdots, S_t, \cdots, S_1]$
3: Train a classifier using New_X_S and Y_S, then predict the labels of New_X_T using trained classifier, and calculate the accuracy of $predict_{Y_T}$.

Figure 3: The comparison of our model GSM and SGF. Two black points are the given points; the solid cyan curve illustrates the true geodesic points; the blue dashed curve on top is the sampling results of GSM, and the yellow lower curve is the sampling results of SGF.

some negations of the estimated velocity v between the source image and target image). The second is that the shape is never unified, and this is because the Exp does not approach the target at $t = 1$ [22].

To validate the GSM method in the Grassmannian manifold, we test the model with two data points and compare the results with the technique mentioned by Gopalan et al. [22]. Fig. 3 shows a good overlaying of correct geodesic points by GSM, while the geodesic curve of SGF does not show an accurate recovery of original points.

4.2 Classification tasks

4.2.1 Dataset descriptions

We test our model using three standard public image datasets: Office10, Caltech10, Office 31 and Office-Home [41, 48, 51]. The features for Office + Caltech 10, Office 31 and Office-Home datasets are extracted from Xception pre-trained network [11][3]. These datasets are widely used in many publications (e.g., [21, 22, 51]), and are benchmark data for evaluating the performance of domain adaptation algorithms. We also use two additional text datasets: 20ng and S-F (student-faculty). The 20ng is newsgroups data, and we want to classify the different categories of articles [30]. The S-F is student and faculty data, in which we want to discriminate between student and faculty.[4] Table 1 lists statistics of these datasets. We then solve and evaluate classification tasks using Alg. 2.

Table 1: Statistics of benchmark datasets

Dataset	# Samples	# Features	# Classes	Domain(s)
Office-10	1410	1000	10	A, W, D
Caltech-10	1123	1000	10	C
Office-31	1330	1000	31	A, W, D
Office-Home	15588	1000	10	A, W, D
20ng	1952	707	2	Tr1, Te1
S-F	300	5	10	Tr2, Te2

[3] Xception is a well-trained deep neural network using Imagenet datasets.
[4] https://github.com/heaventian93/Data

Table 2: Accuracy and timing of three manifolds

Task	Sphere		Kendall		Grassmannian	
	accuracy	time	accuracy	time	accuracy	time
C	**95.8%**	**4.8s**	95.5%	5.4s	95.5%	5.0s
A	**97.0%**	**3.6s**	97.0%	3.8s	96.9%	4.2s
W	**99.7%**	**1.8s**	99.3%	1.9s	99.3%	2.6s
D	**99.4%**	**1.6s**	98.7%	1.8s	98.7%	2.2s
Tr1 $\longrightarrow$ Te1	**85.7%**	**4.4s**	83.6%	6.2s	**85.7%**	11.6s
Tr2 $\longrightarrow$ Te2	**82.0%**	**0.06s**	80.0%	0.07s	**82.0%**	0.08s
Average	**93.3%**	**2.7s**	92.4%	3.2s	93.0%	4.3s

4.2.2 Choosing a manifold

To choose the best manifold for classification tasks, we compare the accuracy and time of three manifolds. The sample size $N = 200$, and the dimensionality $d = 20$ (for S-F dataset, $d = 4$). For the Office + Caltech 10 dataset, we choose the best manifold based on the mean of ten-fold cross validation, and we learn from existing domain training (Tr1(2)), and transferring knowledge to classify domain testing (Te1(2)) in 20ng and S-F datasets. As shown in Table 2, GSM using spherical manifold has the highest accuracy, and minimum computation time. Therefore, the spherical manifold appears suitable for the classification problem, and this conclusion is different from previous work [21, 22], which only considered the Grassmannian manifold.

4.2.3 Comparison to state-of-the-art methods

Take the conclusions from Section 4.2.2: the spherical manifold is suitable for classification problems. In Tables 3 and 5 (full tables are provided in the supplementary file), we compare the performance of our GSM models with 21 state-of-the-art (both traditional and deep learning) methods: Transfer Component Analysis (TCA) [38]; Global and Local Metrics for Domain Adaptation (IGLDA also called ITCA) [25]; Semi-supervised TCA (SSTCA) [38]; Transfer Joint Matching (TJM) [32]; Balanced distribution adaptation (BDA) [50]; Joint distribution alignment (JDA) [31]; Support Vector Machine (SVM) [6]; Geodesic Flow Kernel (GFK) [21]; Manifold Embedded Distribution Alignment (MEDA) [51]; AlexNet [28]; VGG-16 [43]; ResNet-50 [24]; Deep Adaptation Networks (DAN) [33]; Deep Domain Confusion (DDC) [46]; Deep Correlation Alignment (DCORAL) [44]; Adversarial Discriminative Domain Adaptation (ADDA) [47]; Collaborative Adversarial Network (CAN) [54], Join discriminative Domain Adaptation (JDDA) [10]; Joint Adaptation Networks (JAN) [35]; Residual Transfer Networks (RTN) [34]; Domain Adaptive Neural Networks (DANN) [20]; Domain Adaptive Hashing (DAH) [48]; Minimum Discrepancy Deep Adaptation (MDDA) [40]; and Conditional Domain Adversarial Networks (CDAN-RM, CDAN-M) [36]. Results for all traditional methods utilize the same features, which are

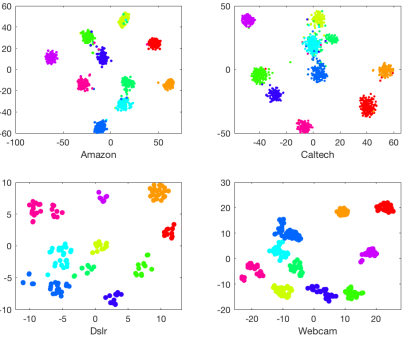

Figure 4: t-SNE view of four domains in Office + Caltech 10 dataset

extracted from the pre-trained Xception neural network.

We combine our GSM model with MEDA model (details are provided in the supplementary file). We first learn the underlying geometry using GSM instead of GFK model (which was used in original MEDA model), then we solve the classification problem using the MEDA model. In Tables 3, 4 and 5 it is clear that our GSM model has higher average accuracy than other methods.

Table 3: Accuracy (%) on Office + Caltech 10 datasets

Task	C → A	C → W	C → D	A → C	A → W	A → D	W → C	W → A	W → D	D → C	D → A	D → W	**Average**
TCA [38]	77.0	80.7	84.7	82.2	68.1	72.6	79.3	86.4	88.5	82.2	86.4	84.7	81.1
ITCA [25]	81.0	65.8	79.6	82.9	70.8	79.0	78.2	85.5	92.4	77.9	82.5	90.5	80.5
SSTCA [33]	79.6	70.5	80.9	76.5	72.5	83.4	69.9	79.5	90.4	78.7	85.2	87.8	79.6
TJM [30]	86.7	84.7	86.0	82.8	78.3	86.0	82.0	86.0	**100**	83.8	89.6	99.3	87.1
BDA [31]	89.5	78.6	81.5	79.6	73.2	84.7	78.1	83.3	**100**	79.7	88.5	98.6	84.6
JDA [31]	88.4	84.4	85.4	81.6	80.7	81.5	82.2	89.8	**100**	86.0	91.5	99.3	87.6
SVM [5]	91.0	78.0	85.4	83.3	72.5	83.4	62.9	72.1	99.4	65.0	78.2	96.6	80.7
GFK [21]	88.8	77.3	86.0	77.4	66.8	79.0	72.0	76.5	**100**	75.5	84.7	99.0	81.9
MEDA [31]	93.0	91.2	89.8	89.0	90.8	88.5	89.0	92.2	99.4	88.6	93.2	98.6	91.9
AlexNet [28]	91.9	83.7	87.1	83.0	79.5	87.4	73.0	83.8	**100**	79.0	87.1	97.7	86.1
DAN [32]	92.0	90.6	89.3	84.1	91.8	91.7	81.2	92.1	**100**	80.3	90.0	98.5	90.1
DDC [46]	91.9	85.4	88.8	85.0	86.1	89.0	78.0	83.8	**100**	79.0	87.1	97.7	86.1
DCORAL [42]	89.8	97.3	91.0	91.9	**100**	90.5	83.7	81.5	90.1	88.6	80.1	92.3	89.7
RTN [29]	93.7	96.9	94.2	88.1	95.2	95.5	86.6	92.5	**100**	84.6	93.8	99.2	93.4
MDDA [41]	93.6	95.2	93.4	89.1	95.7	96.6	86.5	94.8	**100**	84.7	94.7	**99.4**	93.6
GSM	**95.8**	**98.6**	**94.9**	**94.8**	97.6	**99.4**	**95.0**	**95.0**	**100**	**94.3**	**95.7**	98.3	**96.6**

Table 4: Accuracy (%) on Office 31 datasets

Task	A → W	A → D	W → A	W → D	D → A	D → W	**Average**
TCA [38]	82.6	84.1	69.1	99.6	66.1	97.0	83.1
ITCA [25]	81.0	78.7	68.9	99.6	66.6	97.4	82.0
MEDA [31]	83.3	83.3	66.2	96.0	66.7	91.7	81.2
DAN [32]	80.5	78.6	62.8	99.6	63.6	97.1	80.4
RTN [29]	84.5	77.5	64.8	99.4	66.2	96.8	81.6
DANN [20]	82.0	79.7	67.4	99.1	68.2	96.8	81.6
ADDA [47]	86.2	77.8	68.9	98.4	69.5	96.2	82.9
CAN [24]	81.5	65.9	**98.2**	85.5	**99.7**	63.4	82.4
JDDA [10]	82.6	79.8	66.7	99.7	57.4	95.2	80.2
JAN [35]	85.4	84.7	70.0	99.8	68.6	97.4	84.3
GSM	**92.5**	**91.6**	76.1	**100**	77.7	**98.1**	**89.3**

Table 5: Accuracy (%) on Office-Home datasets

Task	A → C	A → P	A → R	C → A	C → P	C → R	P → A	P → C	P → R	R → A	R → C	R → P	**Ave.**
AlexNet [28]	26.4	32.6	41.3	22.1	41.7	42.1	20.5	20.3	51.1	31.0	27.9	54.9	34.3
VGG16 [45]	30.4	45.9	57.5	35.4	48.7	50.8	35.8	30.5	60.2	49.6	34.5	64.0	45.3
DCORAL [42]	32.2	40.5	54.5	31.5	45.8	47.3	30.0	32.3	55.3	44.7	42.8	59.4	42.8
RTN [29]	31.3	40.2	54.6	32.5	46.6	48.3	28.2	32.9	56.4	45.5	44.8	61.3	43.5
DAH [43]	31.6	40.8	51.7	34.7	51.9	52.8	29.9	39.6	60.7	45.0	45.1	62.5	45.5
MDDA [41]	35.2	44.4	57.2	36.8	52.5	53.7	34.8	37.2	62.2	50.0	46.3	66.1	48.0
ResNet-50 [27]	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 [32]	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 [20]	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 [35]	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-RM [34]	49.2	64.8	72.9	53.8	62.4	62.9	49.8	48.8	71.5	65.8	56.4	79.2	61.5
CDAN-M [34]	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
GSM	**55.0**	**80.6**	**82.1**	**66.6**	**81.5**	**80.1**	**68.0**	**54.0**	**81.9**	**70.6**	**57.8**	**83.6**	**71.8**

5 Discussion

One obvious advantage of the GSM model is that it can generate more samples which follow the constraints of their manifold geometry. Unlike the model of Gong et al. [21], we use a different calculation of the Grassmannian manifold, and we can see that the estimated geodesic of our model can recover the ground truth geodesic. In contrast, the sampling results of Gopalan et al. [22] are far from the true ones. Furthermore, from image classification results, we also find that in almost all tested tasks, our model has higher accuracy than other methods. However, a disadvantage of our approach is that input data needs to be normalized, which can cause some information loss, although it might not lose the most critical information.

The spherical GSM model has lower computation time than the other two manifolds across all classification tasks as seen in Table 2. Kendall's manifold has a similar computation time as the spherical manifold, while Grassmannian manifold needs a longer computation time. As described in Sections 3.3-3.5, for Kendall's manifold, only Log map requires the singular value decomposition; while both Log map and Exp map require SVD for Grassmannian manifolds. The singular value decomposition (SVD) has time complexity of $\mathcal{O}(n^3)$. Thus, when n is large, it will have significant computational cost (as demonstrated by the computation time of $Tr1 \longrightarrow Te1$ of Grassmannian GSM in Table 2). Although the classification results of Grassmannian and Kendall's manifold are similar to the spherical manifold, it is undesirable to choose Grassmannian and Kendall's GSM to solve classification problem if n is large.

One interesting phenomenon is that although the sampling results of Gopalan et al. [22] do not follow the rule of the geometry of the manifold, their approach still gets reasonable results. One reason is that it uses a kernel trick so that it can represent the data in some sense, and generate better results than other methods. We also observe that the performance of GSM model is not always better than other methods (e.g., the D $\longrightarrow$ W task), and this variation is caused by the differences across specific domain tasks.

6 Conclusion

In this paper, we propose a geodesic sampling model for different manifolds (Sphere, Kendall's shape and Grassmannian). To validate our model, we first compare geodesic curve recovery of our model and original model of Gopalan et al. [22]. Further, to improve classification accuracy, we choose the spherical manifold and combine distribution alignment leading to the GSM model. The extensive experiments demonstrate that our GSM model outperforms other models and does so efficiently.

There are some obvious next steps. Exp and Log maps can be developed for additional manifolds, and tested on a broader set of supervised learning tasks. This method should also be of value to unbalanced classification tasks in which lots of unlabelled data are available.

References

[1] P-A Absil, Robert Mahony, and Rodolphe Sepulchre. Riemannian geometry of grassmann manifolds with a view on algorithmic computation. *Acta Applicandae Mathematica*, 80(2):199–220, 2004.

[2] Mikhail Belkin and Partha Niyogi. Semi-supervised learning on Riemannian manifolds. *Machine Learning*, 56 (1-3):209–239, 2004.

[3] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine Learning*, 79(1-2):151–175, 2010.

[4] Shai Ben-David, Tyler Lu, Teresa Luu, and Dávid Pál. Impossibility theorems for domain adaptation. In *International Conference on Artificial Intelligence and Statistics*, pages 129–136, 2010.

[5] Alessandro Bergamo and Lorenzo Torresani. Exploiting weakly-labeled web images to improve object classification: a domain adaptation approach. In *Advances in Neural Information Processing Systems*, pages 181–189, 2010.

[6] John Blitzer, Ryan McDonald, and Fernando Pereira. Domain adaptation with structural correspondence learning. In *Proc. Conf. on Empirical Methods in Natural Language Processing*, pages 120–128, 2006.

[7] John Blitzer, Mark Dredze, and Fernando Pereira. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proc. 45th Annual Meeting of the Assoc. of Computational Linguistics*, pages 440–447, 2007.

[8] John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman. Learning bounds for domain adaptation. In *Advances in Neural Information Processing Systems*, pages 129–136, 2008.

[9] Olivier Chapelle, Bernhard Schlkopf, and Alexander Zien. *Semi-Supervised Learning*. MIT Press, 1st edition, 2010. ISBN 0262514125, 9780262514125.

[10] Chao Chen, Zhihong Chen, Boyuan Jiang, and Xinyu Jin. Joint domain alignment and discriminative feature learning for unsupervised deep domain adaptation. *arXiv preprint arXiv:1808.09347*, 2018.

[11] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1251–1258, 2017.

[12] Ronan Collobert, Fabian Sinz, Jason Weston, and Léon Bottou. Large scale transductive svms. *Journal of Machine Learning Research*, 7:1687–1712, December 2006. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=1248547.1248609.

[13] Wenyuan Dai, Gui-Rong Xue, Qiang Yang, and Yong Yu. Transferring naive bayes classifiers for text classification. In *AAAI*, volume 7, pages 540–545, 2007.

[14] Hal Daume III and Daniel Marcu. Domain adaptation for statistical classifiers. *Journal of Artificial Intelligence Research*, 26:101–126, 2006.

[15] Manfredo Perdigão Do Carmo. *Riemannian geometry*. Birkhauser, 1992.

[16] Alan Edelman, Tomás A Arias, and Steven T Smith. The geometry of algorithms with orthogonality constraints. *SIAM journal on Matrix Analysis and Applications*, 20(2):303–353, 1998.

[17] P Thomas Fletcher, Conglin Lu, Stephen M Pizer, and Sarang Joshi. Principal geodesic analysis for the study of nonlinear statistics of shape. *IEEE Transactions on Medical Imaging*, 23(8):995–1005, 2004.

[18] Jean Gallier. Notes on differential geometry and Lie groups. Technical report, University of Pennsylvania, 2012.

[19] Kyle A Gallivan, Anuj Srivastava, Xiuwen Liu, and Paul Van Dooren. Efficient algorithms for inferences on grassmann manifolds. In *Statistical Signal Processing, 2003 IEEE Workshop on*, pages 315–318. IEEE, 2003.

[20] Muhammad Ghifary, W Bastiaan Kleijn, and Mengjie Zhang. Domain adaptive neural networks for object recognition. In *Pacific Rim international conference on artificial intelligence*, pages 898–904. Springer, 2014.

[21] Boqing Gong, Yuan Shi, Fei Sha, and Kristen Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2066–2073. IEEE, 2012.

[22] Raghuraman Gopalan, Ruonan Li, and Rama Chellappa. Domain adaptation for object recognition: An unsupervised approach. In *IEEE International Conference on Computer Vision (ICCV)*, pages 999–1006. IEEE, 2011.

[23] John C Gower. Generalized procrustes analysis. *Psychometrika*, 40(1):33–51, 1975.

[24] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[25] Min Jiang, Wenzhen Huang, Zhongqiang Huang, and Gary G Yen. Integration of global and local metrics for domain adaptation learning via dimensionality reduction. *IEEE Transactions on Cybernetics*, 47(1):38–51, 2017.

[26] Ian Jolliffe. Principal component analysis. In *International encyclopedia of statistical science*, pages 1094–1096. Springer, 2011.

[27] David G Kendall. Shape manifolds, procrustean metrics, and complex projective spaces. *Bulletin of the London Mathematical Society*, 16(2):81–121, 1984.

[28] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012.

[29] Abhishek Kumar, Avishek Saha, and Hal Daume. Co-regularization based semi-supervised domain adaptation. In *Advances in Neural Information Processing Systems*, pages 478–486, 2010.

[30] Hugo Larochelle and Yoshua Bengio. Classification using discriminative restricted boltzmann machines. In *Proceedings of the 25th International Conference on Machine Learning*, pages 536–543. ACM, 2008.

[31] Mingsheng Long, Jianmin Wang, Guiguang Ding, Jiaguang Sun, and Philip S Yu. Transfer feature learning with joint distribution adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2200–2207, 2013.

[32] Mingsheng Long, Jianmin Wang, Guiguang Ding, Jiaguang Sun, and Philip S Yu. Transfer joint matching for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1410–1417, 2014.

[33] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael I Jordan. Learning transferable features with deep adaptation networks. *arXiv preprint arXiv:1502.02791*, 2015.

[34] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Unsupervised domain adaptation with residual transfer networks. In *Advances in Neural Information Processing Systems*, pages 136–144, 2016.

[35] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael 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.

[36] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Conditional adversarial domain adaptation. In *Advances in Neural Information Processing Systems*, pages 1647–1657, 2018.

[37] Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Domain adaptation: Learning bounds and algorithms. *arXiv preprint arXiv:0902.3430*, 2009.

[38] Sinno Jialin Pan, Ivor W Tsang, James T Kwok, and Qiang Yang. Domain adaptation via transfer component analysis. *IEEE Trans. on Neural Networks*, 22(2):199–210, 2011.

[39] Xavier Pennec. Statistical computing on manifolds: from riemannian geometry to computational anatomy. In *Emerging Trends in Visual Computing*, pages 347–386. Springer, 2009.

[40] Mohammad Mahfujur Rahman, Clinton Fookes, Mahsa Baktashmotlagh, and Sridha Sridharan. On minimum discrepancy estimation for deep domain adaptation. *arXiv preprint arXiv:1901.00282*, 2019.

[41] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *European conference on computer vision*, pages 213–226. Springer, 2010.

[42] Patrice Y Simard, Yann A LeCun, John S Denker, and Bernard Victorri. Transformation invariance in pattern recognition-tangent distance and tangent propagation. In *Neural networks: tricks of the trade*, pages 239–274. Springer, 1998.

[43] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[44] Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *European Conference on Computer Vision*, pages 443–450. Springer, 2016.

[45] Gokhan Tur. Co-adaptation: Adaptive co-training for semi-supervised learning. In *IEEE International Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3721–3724. IEEE, 2009.

[46] Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474*, 2014.

[47] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7167–7176, 2017.

[48] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5018–5027, 2017.

[49] Chang Wang and Sridhar Mahadevan. Manifold alignment without correspondence. In *IJCAI*, volume 2, page 3, 2009.

[50] Jindong Wang, Yiqiang Chen, Shuji Hao, Wenjie Feng, and Zhiqi Shen. Balanced distribution adaptation for transfer learning. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, pages 1129–1134, 2017.

[51] Jindong Wang, Wenjie Feng, Yiqiang Chen, Han Yu, Meiyu Huang, and Philip 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. doi: 10.1145/3240508.3240512.

[52] Richard C Wilson and Edwin 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.

[53] Miaomiao Zhang and P Thomas Fletcher. Probabilistic principal geodesic analysis. In *Advances in Neural Information Processing Systems*, pages 1178–1186, 2013.

[54] Weichen Zhang, Wanli Ouyang, Wen Li, and Dong 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.