Geometry Flow-Based Deep Riemannian Metric Learning

Yangyang Li, Chaoqun Fei, Chuanqing Wang, Hongming Shan, *Senior Member, IEEE*, and Ruqian Lu

***Abstract*—Deep metric learning (DML) has achieved great results on visual understanding tasks by seamlessly integrating conventional metric learning with deep neural networks. Existing deep metric learning methods focus on designing pair-based distance loss to decrease intra-class distance while increasing inter-class distance. However, these methods fail to preserve the geometric structure of data in the embedding space, which leads to the spatial structure shift across mini-batches and may slow down the convergence of embedding learning. To alleviate these issues, by assuming that the input data is embedded in a lower-dimensional sub-manifold, we propose a novel deep Riemannian metric learning (DRML) framework that exploits the non-Euclidean geometric structural information. Considering that the curvature information of data measures how much the Riemannian (non-Euclidean) metric deviates from the Euclidean metric, we leverage geometry flow, which is called a geometric evolution equation, to characterize the relation between the Riemannian metric and its curvature. Our DRML not only regularizes the local neighborhoods connection of the embeddings at the hidden layer but also adapts the embeddings to preserve the geometric structure of the data. On several benchmark datasets, the proposed DRML outperforms all existing methods and these results demonstrate its effectiveness.**

***Index Terms*—Curvature regularization, deep metric learning (DML), embedding learning, geometry flow, riemannian metric.**

I. Introduction

MEASURING the distance (dis)similarity between images is a fundamental problem in computer vision. In general, the geometric structure of image data set is complex, especially when it is embedded in a nonlinear sub-manifold [1], [2]. It is insufficient to reflect the real similarity of the data utterly using Euclidean metric, so metric learning was proposed [3]. The main purpose of metric learning is to learn an appropriate metric to measure the similarity among non-Euclidean data [4]. However, one significant weakness of the traditional metric learning is that currently available methods, including the online learning and offline learning ones, cannot efficiently process the large scale data due to their limited learning process [5].

With the rise of deep learning, researchers proposed deep metric learning (DML) [6], [7] which combines the traditional metric learning with deep neural networks. DML has achieved the state-of-the-art results on multiple benchmark datasets. Unlike traditional metric learning that learns a Mahalanobis metric matrix, DML proposes to learn an embedding mapping from the original features to a low-dimensional vector space and outputs a set of vector representations. In this framework, the distance between similar data (within the same class) is relatively small while the distance between different classes is relatively large. However, existing deep metric learning methods [8]−[10] all focus on designing different pair-based distance losses without considering the global geometric structure of the overall dataset. In addition, the geometric structure of features in hidden layers may be shifted due to the nonlinear activation function, which may slow down the convergence of DML methods. Despite the great success of DML, it remains challenging to fully exploit its geometric structure on non-Euclidean (manifold) data.

Geometric structure information of data is of great importance for learning tasks [11], [12]. On the one hand, the geometric structure reflects the feature distribution of initial data and contains the predictive information about learning tasks, such as the development process of diseases [13]. On the other hand, for the deep neural network, the geometric structure of data can be regarded as a set of additional unsupervised information [14], [15], which alleviates overfitting and improves the generalization performance. Specially, the local structural information of data characterizes how neighborhoods of a pair of images relate to each other [16], which is very useful information for the learning tasks with complex structured data. However, although the existing deep metric learning methods assume that the original data distribute in a non-Euclidean space, they leverage Euclidean metric to measure the similarity among data in practice. Euclidean metric has limitations for nonlinear distributed data. There can be different classes of images with tiny Euclidean distances due to the short-cuts

Manuscript received August 16, 2022; revised November 14, 2022; accepted December 20, 2022. This work was supported in part by the Young Elite Scientists Sponsorship Program by CAST (2022QNRC001), the National Natural Science Foundation of China (61621003, 62101136), Natural Science Foundation of Shanghai (21ZR1403600), Shanghai Municipal Science and Technology Major Project (2018SHZDZX01) and ZJLab, and Shanghai Municipal of Science and Technology Project (20JC1419500). Recommended by Associate Editor Hui Yu. *(Corresponding author: Yangyang Li.)*

Citation: Y. Y. Li, C. Q. Fei, C. Q. Wang, H. M. Shan, and R. Q. Lu, "Geometry flow-based deep riemannian metric learning," *IEEE/CAA J. Autom. Sinica*, vol. 10, no. 9, pp. 1882–1892, Sept. 2023.

Y. Y. Li, C. Q. Fei, C. Q. Wang, and R. Q. Lu are with the Key Lab of MADIS Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, China (e-mail: yyli@amss.ac.cn; cqfei@amss.ac.cn; tranking@amss.ac.cn; rqlu@math.ac.cn).

H. M. Shan is with the Institute of Science and Technology for Brain-Inspired Intelligence and Key Laboratory of Computational Neuroscience and Brain-Inspired Intelligence (Ministry of Education) and MOE Frontiers Center for Brain Science, Fudan University, Shanghai 200433, and also with the Shanghai Center for Brain Science and Brain-Inspired Technology, Shanghai 200031, China (e-mail: hmshan@fudan.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JAS.2023.123399

problem in non-Euclidean space. A formal illustration is shown in Fig. 1. To effectively take advantage of the geometric structure of data, we need a more discriminative metric, which can distinguish these scenarios in Fig. 1.

Fig. 1. Illustration of short-cuts problem in non-Euclidean space. The black curve indicates the geodesic between the two points, and the green solid line indicates the short circuit between the two points.

In this paper, we assume that input data processed by deep metric learning is embedded in a nonlinear sub-manifold. To quantify such geometric structure, we draw sensation from recent study of geometry flow learning [17]. Inspired by the power of geometry flow in continuous manifold (see Section III-C), e.g., the Ricci flow in Riemannian manifold, geometry flow learning proposes constructing an evolution function between the metric and curvature of data and is combined with metric learning to learn a Mahalanobis metric which approximates the real Riemannian metric; see Section IV-A. In theory, the discrete curvature measures how much the geometry of a pair of neighborhoods deviates from a "flat" space. Intuitively, the curvature quantifies the distribution smoothness of a neighborhood. Such information is useful to distinguish the short-cuts between neighbors and can be leveraged by a deep metric learning method.

We propose *geometry flow-based deep riemannian metric learning (DRML)*, the first deep metric learning method based on non-Euclidean geometric structure. In order to combine the geometry flow model with deep metric learning, we design a geometry flow layer in the hidden layers of the deep neural networks. Technically, we design a loss function in the geometry flow layer by exploiting the curvature information of data to regularize the feature distributions inspired by geometry flow. Theoretically, the geometric structure regularization in the hidden layer is equivalent to performing the Laplacian smoothing [18] to data, which can boost the convergence of the embedding learning and improve the performance of the final embedding results. To the best of our knowledge, this is the first work to leverage the intrinsic non-Euclidean geometric structure for deep metric learning.

The contributions of this paper are summarized as follows:

1) Propose a new deep Riemannian metric learning method based on geometry flow, in which the geometric structure is added as a regularization term to the hidden layer so that the feature distributions of hidden layers keep their initial manifold structure.

2) Use curvature information of feature distributions for the regularization of deep metric learning, which is first introduced in designing deep metric learning.

3) Extensive experimental results validate the superiority of the proposed DRML over the state-of-the-art methods.

The rest of this paper is organized as follows. In Section II, we introduce related works on deep metric learning and curvature-aware methods. In Section III, we present preliminaries on the basic definitions and theories. The proposed deep metric learning method based on geometry flow is described in Section IV. Section V gives the experiments and evaluation results. Finally, we conclude our work in Section VI.

II. Related Work

A. Deep Metric Learning

We briefly summarize previous works on deep metric learning. In 2015, Google team introduced Facenet model [6], which was the first to propose a CNN-based deep metric learning model to learn high-level feature representation of data. To rectify the compactness problem between classes, Wen *et al.* [8] proposed center loss, which did not consider the difference between classes. He *et al.* [9] designed triplet-center loss combined with triplet loss. All of these models above converge slowly due to the choice of pairwise samples. To address the above limitation, N-pair loss [19] extended triplet loss and constructed more negative samples which are far away from the anchor. Other loss functions include hierarchical triplet loss [20], multi-similarity loss [21], histogram loss [22], and angular loss [23]. For all the methods, they mainly focus on optimizing angular distance between data. However, none of them explicitly considers the feature distribution information. In 2020, Zhang *et al.* [24] proposed a new spherical embedding constraint to regularize the distribution of the norms, which can rectify the unstable gradient in batch optimization. Zhao *et al.* [25] presented a deep interpretable metric learning method for more transparent embedding learning based on structural matching strategy. Chen *et al.* [26] introduced a graph consistency regularization term to optimize the feature distribution. At the same time, Seidenschwarz *et al.* [27] used message passing networks to uncover the relations in a mini-batch. However, there have no methods to model the non-Euclidean geometric structure of the overall data.

B. Curvature-Aware Methods

Different proposals for discrete curvature information have been introduced in recent years, including discrete curvature (geodesic curvature, Ricci curvature) on manifold and Ollivier-Ricci curvature [28] on graph. Kim *et al.* [29] first applied curvature information in semi-supervised learning. Similar applications have been shown in the characterization of image distributions [30], [31]. There are also applications of Ricci flow in manifold learning to dynamically extract the geometric features of data [32]. Recently, various methods have been proposed to implement Ollivier-Ricci curvature on graph neural network to reweigh different channels of the messages [33], [34]. Furthermore, Ollivier-Ricci curvature was also applied in network alignment [35] and community detection [36]. To the best of our knowledge, curvature information has not been used in training deep metric learning.

III. Preliminaries

In this section, we first present the basic definition of Rie-

mannian metric. Then, we give the basic theory of curvature information and geometry flow in geometry. Formally, suppose the input data is represented as a set of N data samples, $\{x_1, x_2, \ldots, x_N\}, x_i \in \mathbb{R}^D$, which are sampled from a low-dimensional manifold $\mathcal{M}$ embedded in $\mathbb{R}^D$, where D is the feature dimension.

A. Riemannian Metric

In the Riemannian geometry, Riemannian metric is used to compute the intrinsic geometric structure of $\mathcal{M}$. The corresponding Taylor expansion of Riemannian metric is given as [37]

$$g_{ij} = g_{ij}|_p + \partial_k g_{ij}|_p x^k + \frac{1}{2}\partial_l \partial_k g_{ij}|_p x^k x^l + \frac{1}{6}\partial_m \partial_l \partial_k g_{ij}|_p x^k x^l x^m + \cdots \quad (1)$$

where $g_{ij}|_p = \delta_{ij}$, $\partial_k g_{ij}|_p x^k = 0$, $\frac{1}{2}\partial_l \partial_k g_{ij}|_p x^k x^l = -\frac{1}{3}R_{ikjl}|_p x^k x^l$. p is an arbitrary point on $\mathcal{M}$, g_{ij} is the Riemannian metric, and R_{ikjl} is its Riemannian curvature under $\{i, j, k, l\}$th vector fields.

In (1), the Riemannian metric is not locally isometric to Euclidean metric due to the nonzero Riemannian curvature. Therefore, in order to learn the intrinsic geometric structure of the embedded manifold, the curvature information of this manifold needs to be explored firstly.

B. Curvature Information

In the Riemannian geometry, Riemannian curvature of a manifold is represented by a fourth-order tensor. However, for discrete data, the differential geometric structure does not exist. It is difficult to calculate the continuous curvature directly. In this paper, we evaluate the discrete curvature information of each neighborhood by summing up the geodesic curvatures between all neighbors. Geodesic represents the shortest path between two points on a manifold. And geodesic curvature quantifies the curve bending of geodesic. We follow the geodesic curvature estimation method [30], which is computed as:

$$R\left(x_i, x_j\right) = \frac{\|\theta\left(J_i, J_j\right)\|}{d_{ge}\left(x_i, x_j\right)} \quad (2)$$

$\theta(J_i, J_j)$ measures the principal angle between the tangent space $J_i \in \mathbb{R}^{D\times d_i}(T_{x_i}\mathcal{M})$ and $J_j \in \mathbb{R}^{D\times d_j}(T_{x_j}\mathcal{M})$. $d_{ge}(x_i, x_j)$ is the geodesic distance between x_i and x_j. The detailed computation process is shown in Appendix.

C. Geometry Flow

In the mathematical field of differential geometry, a geometry flow, also called a geometric evolution equation, is a type of partial differential equation for a geometric object such as a Riemannian metric or an embedding. Here, we take the classical Ricci flow [38] as an example to introduce geometry flow.

Given a manifold $\mathcal{M}$, g_{ij} is a Riemannian metric defined on it, and its Ricci curvature is R_{ij}. Hamilton's Ricci flow [39] is a parabolic partial differential evolution equation about time t,

$$\frac{\partial}{\partial t} g_{ij} = -2R_{ij}. \quad (3)$$

Ricci flow can be viewed as a heat equation for Riemannian metric [40]. One of the key properties of Ricci flow is that the Riemannian metric and curvature mutually evolve according to this partial differential equation (3). Thus Ricci flow tends to smooth out the irregularity of curvature [36]. By Ricci flow, neighborhoods with positive curvature tend to shrink and neighborhoods with negative curvature tend to expend [41].

In this work, we tend to smooth out the feature distributions of data by exploiting its curvature information. We follow the discrete geometry flow learning model [17] to construct an evolution function between metric and curvature of data, and learn how to use it to re-correct the relative distance among neighbors. Eventually, the curved regions tend to be flattened by iterating the discrete geometry flow.

IV. Proposed Method

The key idea of DRML is how to combine the geometry flow model with DML. In this section, we outline the deep Riemannian metric learning method and give the corresponding theoretical analysis.

A. Geometry Flow Learning

In 2018, [17] was the first to introduce geometric flow learning on discrete data. The proposed geometry flow model combined with metric learning is shown to provide a nearly perfect discrete version of Riemannian metric by optimizing an evolution function between the metric and curvature of data. They leverage the geometry flow in a data-driven manner, which is formally defined as

$$L_R = D_\phi(M_R, M_0) + \lambda \sum_{i,j}^{N} \left(d_{ge}\left(x_i, x_j\right) - d_{M_R}\left(x_i, x_j\right)\right)^2. \quad (4)$$

The first term D_ϕ quantifies the similarity between the learnt Mahalanobis metric M_R and Euclidean metric M_0. The second term represents the difference between geodesic distance $d_{ge}(x_i, x_j)$ and the corresponding Mahalanobis distance $d_{M_R}(x_i, x_j)$ under metric learning, which can be approximated by the principal angle $\|\theta(J_i, J_j)\|$ based on the curvature estimation ((2)). According to (2), (4) is transformed as

$$L_R = D_\phi(M_R, M_0) + \lambda \sum_{i,j}^{N} \left\{d_{M_R}^2\left(x_i, x_j\right) \cdot R^2\left(x_i, x_j\right)\right\}. \quad (5)$$

Geometry flow learning combines online learning to optimize this discrete evolution function (5). Geodesic curvature evolves under this function (5). The Mahalanobis distances between neighbors with nonzero geodesic curvature tend to relatively expand by iterating the geometry flow process. Here, geodesic curvature can be expressed as a regularization term of the Mahalanobis distance.

B. Geometry Flow Layer

In this section, we propose a CNN-based model that addresses the aforementioned limitations of DML algorithms by designing a new geometry flow layer. We introduce our algorithm process in each mini-batch. Formally, suppose the

input data in ith convolutional layer is $X^i = \{X_1^i, X_2^i, \ldots, X_n^i\}$. $X_j^i \in \mathbb{R}^{c\times h\times w}$ is a 3D tensor and n is the number of samples in one mini-batch. The parameters of ith hidden layer are set to W^i, b^i, and the activation function is expressed as σ.

After convolution and activation function, the output of CNN in the ith layer is expressed as follows:

$$\sigma_j^i = \sigma\left(W^i X_j^i + b^i\right), j = 1, \ldots, n. \tag{6}$$

Traditional CNN models take the result of (6) as the input of the $(i+1)$th convolutional layer.

In this work, we construct a new geometry flow layer between the ith activation function layer and the $(i+1)$th convolutional layer. In the geometry flow layer, we use the relation function (5) to regularize the feature distribution of the $(i+1)$th layer's input by leveraging the geometric structure inherited from previous layer. To achieve back propagation, we convert (5) into the following formula:

$$L_R = \sum_{j=1}^{n} \|X_j^{i+1} - \sigma_j^i\|^2 + \lambda \sum_{j,l=1}^{n} \left(\|X_j^{i+1} - X_l^{i+1}\|^2 \cdot \phi\left(R\left(X_j^i, X_l^i\right)\right)\right) \tag{7}$$

where $R(X_j^i, X_l^i)$ represents the geodesic curvature between X_j^i and X_l^i (see Appendix), and $\phi(R(X_j^i, X_l^i))$ is the curvature regularization term; see Section IV-C.

In (7), the first term quantifies the metric distance between two layers which corresponds to the first term in (5); and the second term denotes the curvature regularization term to feature distribution, which is corresponding to the second term in (5).

We optimize (7) by computing the gradient, and the derivative of the solved variable is obtained,

$$\frac{\partial L_R}{\partial X_j^{i+1}} = 2\left(X_j^{i+1} - \sigma_j^i\right) + 2\lambda \sum_{l=1}^{n} \left(\left(X_j^{i+1} - X_l^{i+1}\right) \cdot \phi\left(R\left(X_j^i, X_l^i\right)\right)\right). \tag{8}$$

Let $\frac{\partial L_R}{\partial X_j^{i+1}} = 0$, then we obtain the following formula:

$$X_j^{i+1} = \frac{\sigma_j^i + \lambda \sum_{l\neq j,l=1}^{n} \left(X_l^{i+1} \cdot \phi\left(R\left(X_j^i, X_l^i\right)\right)\right)}{1 + \lambda \sum_{l\neq j,l=1}^{n} \phi\left(R\left(X_j^i, x_l^i\right)\right)}. \tag{9}$$

By replacing X_l^{i+1} with σ_l^i on the right hand of (9) approximately, (9) can be re-expressed as follows:

$$X_j^{i+1} = \frac{\sigma_j^i + \lambda \sum_{l\neq j,l=1}^{n} \sigma_l^i \cdot \phi\left(R\left(X_j^i, X_l^i\right)\right)}{1 + \lambda \sum_{l\neq j,l=1}^{n} \phi\left(R\left(X_j^i, X_l^i\right)\right)}. \tag{10}$$

It can be seen from (10) that adding geometry flow constraint between hidden layers is equivalent to re-representation of hidden layer features using curvature information. If we do not consider the geometric structure, namely $\lambda = 0$, X_j^{i+1} is equal to the traditional activation function output σ_j^i.

C. Curvature Regularization Term

Intuitively, geodesic curvature measures how much a geodesic deviates from being straight, and could be used to alleviate the short-cuts problem in the hidden layers. If the geodesic curvature between two neighbors is not zero, the corresponding Mahalanobis distance would increase by iterating the geometry flow process. By optimizing (7), if the geodesic curvature is larger, the distance between two points can be guaranteed to relatively extend more only if $\phi(R(X_j^i, X_l^i))$ is smaller. Therefore, in this method, we set $\phi(R(X_j^i, X_l^i))$ to be the similarity weight w_{jl}^{i+1} between X_j^{i+1}, X_l^{i+1}. Furthermore, we use the geodesic curvature as a penalty to reweigh the distance similarity between data, which is updated iteratively under geometry flow. The corresponding iterative equation is shown as

$$\phi\left(R\left(X_j^i, X_l^i\right)\right) = 1 - \epsilon \cdot R\left(X_j^i, X_l^i\right) \tag{11}$$

$$w_{jl}^{i+1} = \phi\left(R\left(X_j^i, X_l^i\right)\right). \tag{12}$$

The larger the geodesic curvature is, the smaller the similarity w_{jl}^{i+1} is. When the geodesic curvature $R\left(X_j^i, X_l^i\right) = 0$, the similarity metric between data is equal to Euclidean metric.

In addition, in order to reduce the complexity of our method, we use the nearest neighbor method [42] to construct an adjacency graph on each mini-batch. We set a parameter δ, and when the normalized distance between X_j^i and X_l^i is greater than δ, set $w_{jl}^{i+1} = 0$; otherwise, we compute the geodesic curvature and set w_{jl}^{i+1} to (12). The similarity weight w_{jl}^{i+1} inherits the local geometric structure of the previous layer's input. Thus the next layer's input X^{i+1} optimally preserves the local structure of the previous layer X^i. The overall illustration diagram of our method is shown in Fig. 2 and Algorithm 1.

Algorithm 1 DRML

1: **Input:** the input of ith convolutional layer X_j^i, output of the ith convolutional layer σ_j^i, j represents jth sample in the mini-batch, distance threshold δ, adjustment factor λ, ϵ.
2: calculate normalized L_2 distance d_{jl}^i for X_j^i, X_l^i, $j \neq l$
3: **if** $d_{jl}^i > \delta$ **then**
4: set $w_{jl}^{i+1} \leftarrow 0$
5: **else if** $d_{jl}^i \leq \delta$ **then**
6: calculate $R\left(X_j^i, X_l^i\right)$ using (2)
7: update $w_{jl}^{i+1} \leftarrow 1 - \epsilon \cdot R\left(X_j^i, X_l^i\right)$
8: **end if**
9: calculate X_j^{i+1} using (14)
10: **Output:** the input of next convolutional layer X_j^{i+1}

Our proposed DRML is a kind of adaptable method, which can be combined with all the existing deep metric learning methods. It provides a geometry regularization term for deep neural network, and improves the convergence and performance of the models. In Section IV-D, we will give some interpretation analysis for the proposed method in geometry.

D. Theoretical Analysis

In the following, we analyze the influence of geometry flow

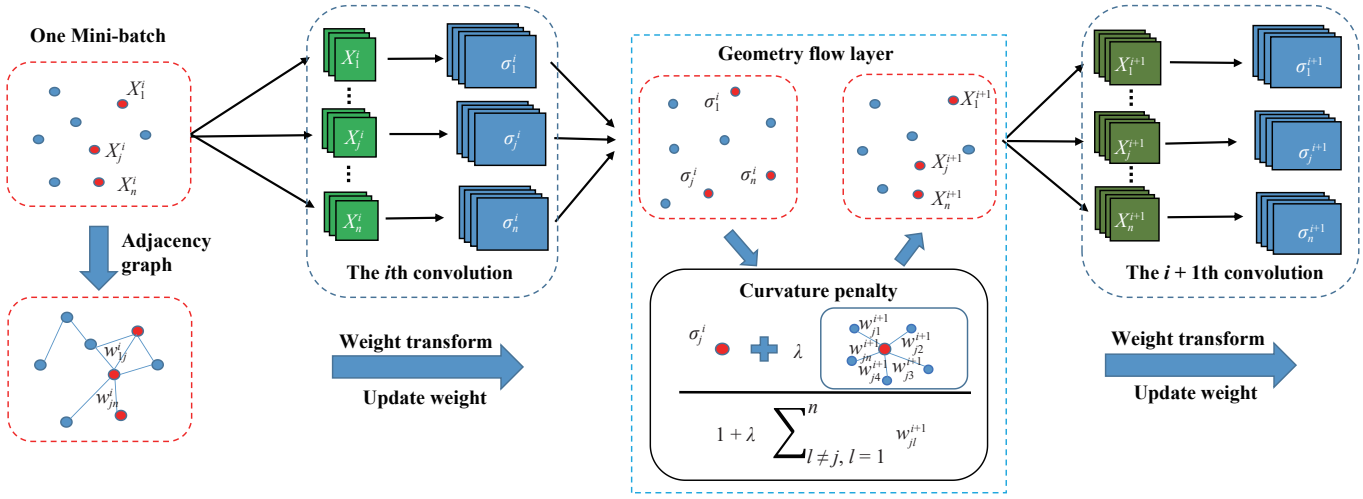


Fig. 2. The overall pipeline of DRML model between two convolution layers. Since the non-linear operation of activation function (such as ReLU) in each convolutional layer, the features distribution of each mini-batch may be shifted. We add an extra geometry flow layer behind the activation function operation in each convolutional layer to explicitly preserve and transmit the original non-Euclidean geometric structure of input data. The geometric structure of each mini-batch is extracted from the former convolutional layer and transmitted to the next layer by the middle geometry flow layer.

layer to the internal batch optimization of deep neural network. Based on the back propagation mechanism, adding geometry flow layers can affect the gradient change of each layer. For analysis of convenience, we regard the ith layer and $(i+1)$th layer as an example to illustrate the influence of the geometry flow layer on the updating of parameters and feature distributions. Suppose the regularization mapping in geometry flow layer is F^i ((10)). Under these assumptions, the feature mapping between two adjacent convolutional layers can be formulated as

$$X^{i+1} = F^i\left(\sigma\left(W^i X^i + b^i\right)\right). \tag{13}$$

The input of jth sample in the $(i+1)$th layer is represented as below:

$$X_j^{i+1} = \frac{\sigma_j^i + \lambda \sum_{l\neq j, l=1}^{n} \sigma_l^i w_{jl}^{i+1}}{1+\lambda \sum_{l\neq j, l=1}^{n} w_{jl}^{i+1}} \tag{14}$$

w_{jl}^{i+1} is the similarity weight between two input samples defined in (12). The matrix form of gradient $\frac{\partial F^i}{\partial \sigma}$ is

$$\frac{\partial F^i}{\partial \sigma} = \begin{bmatrix} \frac{1}{1+\lambda\sum_{l\neq 1} w_{1l}^{i+1}} & \cdots & \frac{\lambda w_{n1}^{i+1}}{1+\lambda\sum_{l\neq N} w_{nl}^{i+1}} \\ \vdots & \ddots & \vdots \\ \frac{\lambda w_{1n}^{i+1}}{1+\lambda\sum_{l\neq 1} w_{1l}^{i+1}} & \cdots & \frac{1}{1+\lambda\sum_{l\neq n} w_{nl}^{i+1}} \end{bmatrix}. \tag{15}$$

By analyzing the gradient matrix $\frac{\partial F^i}{\partial \sigma}$ of the geometry flow layer, we obtain the following conclusions.

Proposition 1: All the eigenvalues γ of the gradient matrix $\frac{\partial F^i}{\partial \sigma}$ satisfy $|\gamma| \leq 1$.

Proof: Suppose γ is an arbitrary eigenvalue of $[\frac{\partial F^i}{\partial \sigma}]^T$, and Γ is the corresponding eigenvector, then we can get $[\frac{\partial F^i}{\partial \sigma}]^T\Gamma = \gamma\Gamma$. That is,

$$\sum_{l=1}^{n} a_{jl}\Gamma_l = \gamma\Gamma_j, \quad j = 1,2,\ldots,n$$

where $[\frac{\partial F^i}{\partial \sigma}]^T = \left(a_{jl}\right)_{n\times n}$, $\Gamma = [\Gamma_1,\Gamma_2,\ldots,\Gamma_n]^T$.

$$\Gamma_j\left(\gamma - a_{jj}\right) = \sum_{l=1,l\neq j}^{n} a_{jl}\Gamma_l$$

$$|\gamma - a_{jj}| = |\sum_{l=1,l\neq j}^{n} a_{jl}\frac{\Gamma_l}{\Gamma_j}| \leq \sum_{l=1,l\neq j} |a_{jl}|\cdot|\frac{\Gamma_l}{\Gamma_j}|.$$

If $|\Gamma_j| = \max_l|\Gamma_l|$, $|\frac{\Gamma_l}{\Gamma_j}| \leq 1$, then we obtain the following result:

$$|\gamma - a_{jj}| \leq \sum_{l=1,l\neq j} |a_{jl}|.$$

For each eigenvalue of $[\frac{\partial F^i}{\partial \sigma}]^T$, there must exist Γ_j, which satisfies that $|\Gamma_j| = \max_l|\Gamma_l|$, so that the above equation is transformed as follows:

$$a_{jj} - \sum_{l=1,l\neq j}^{n} |a_{jl}| \leq \gamma \leq \sum_{l=1}^{n} |a_{jl}|.$$

For each element a_{jl}, it satisfies $0 < a_{jl} < 1$, so we get the following result:

$$-1 \leq a_{jj} - \sum_{l=1,l\neq j}^{n} |a_{jl}| \leq \gamma \leq 1.$$

■

Notice that geometry flow layer only regularizes the feature distributions of hidden convolutional layers, but does not explain whether these geometric structures are inherited and preserved among the training process. Therefore, we seek to use Proposition 1 characterizing how the geometric structures of data relate to each other layer by layer. We get the following conclusions shown in Propositions 2 and 3.

In order to analyze the variation of feature distributions in the hidden layer, we leverage Gram matrix to describe the

spatial distribution of data. Gram matrix is an inner product matrix, which can be seen as a distance metric and the corresponding eigenvalues measure the distribution variance among data.

Proposition 2: Geometry flow layer regularization is a type of Laplacian smoothing operator. The neighborhood regions of data tend to be more densely after regularization by geometry flow layer. That is,

$$\lambda_j\left(\hat{G}^i\right) \le \mu_j\left(G^i\right), j = 1, 2, \ldots, n$$

where G^i represents the Gram matrix among the output features of the ith convolutional layer, $\hat{G}^i$ represents the corresponding Gram matrix of the output with geometry flow layer regularization, and λ_j, μ_j are the corresponding eigenvalues.

Proof: According to the definition of Gram matrix, it represents the inner product matrix between all the samples of the ith mini-batch. Therefore, $G^{i+1}, \hat{G}^{i+1}$ can be represented as follows:

$$G^{i+1} = \begin{bmatrix} \sigma_1^i \\ \vdots \\ \sigma_n^i \end{bmatrix} \cdot \begin{bmatrix} \sigma_1^i & \cdots & \sigma_n^i \end{bmatrix}$$

$$\hat{G}^i = \begin{bmatrix} X_1^{i+1} \\ \vdots \\ X_n^{i+1} \end{bmatrix} \cdot \begin{bmatrix} X_1^{i+1} & \cdots & X_n^{i+1} \end{bmatrix}.$$

According to (14) in this paper, we have

$$\begin{bmatrix} X_1^{i+1} & \cdots & X_n^{i+1} \end{bmatrix} = \begin{bmatrix} \sigma_1^i & \cdots & \sigma_n^i \end{bmatrix} \cdot N$$

where $N = \frac{\partial F^i}{\partial \sigma}$. Therefore, the Gram matrix $\hat{G}^i$ is represented as follows:

$$\hat{G}^i = N^T \cdot G_i \cdot N.$$

According to Theorem 1, the eigenvalues of N satisfy $|\gamma| \le 1$, thus, $\lambda_j(\hat{G}^i) \le \mu_j(G^i)$, for all $j = 1, 2, \ldots, n$. ■

In the following proposition we prove the convergence of geodesic curvature.

Proposition 3: The geometry flow associated with the geodesic curvature evolves the relative distance among neighbors, namely, the higher geodesic curvature between any two neighbors decreases faster than the lower curvature by iterating the deep neural network.

Proof: We use similar triangle property to prove the variation of geodesic curvature between adjacent data points. Consider three adjacent data points expressed as X_j^i, X_k^i, X_l^i formed as a geodesic triangle. Suppose the Euclidean distance between any two points is equal. That is,

$$d(X_j^i, X_k^i) = d(X_j^i, X_l^i) = d(X_k^i, X_l^i).$$

Also suppose the relationship of geodesic curvature between these data points is shown as follows:

$$R(X_j^i, X_k^i) > R(X_j^i, X_l^i) > R(X_k^i, X_l^i).$$

Then the corresponding similarity weights in the ith geometry flow layer between them satisfies the following relation:

$$w_{jk}^{i+1} < w_{jl}^{i+1} < w_{kl}^{i+1}.$$

In addition, the geodesic distance between them satisfy the following relation:

$$d_{ge}(X_j^i, X_k^i) > d_{ge}(X_j^i, X_l^i) > d_{ge}(X_k^i, X_l^i).$$

According to the graph Laplacian, by regularizing geometry flow layer, we obtain the following relationship between $X_j^{i+1}, X_k^{i+1}, X_l^{i+1}$ in theory:

$$d(X_j^{i+1}, X_k^{i+1}) > d(X_j^{i+1}, X_l^{i+1}) > d(X_k^{i+1}, X_l^{i+1}).$$

According to the similar triangle property, the geodesic curvature between X_j^{i+1}, X_k^{i+1} drops most.

When training the deep neural network model, since we do not know the hidden propagation mechanism, it is difficult to prove the geodesic curvature decreasing step by step. ■

By the above theoretical analysis, geometry flow layer can guarantee the initial geometric structure of neighborhoods to inherit and propagate in hidden layers. Curvature information effectively helps to make the distribution of hidden layers more smoothing. Furthermore, our method is generally efficient in both convergence and generalization due to the Laplacian smoothing operation.

V. Experimental Setup and Results

In this section, we first do deep metric learning task on three real word image datasets. Secondly, we do ablation study on several parameters.

A. Datasets

CUB200-2011 [43]: There are 11788 bird images in this dataset, including 200 bird subsets, 5864 images in the training set and 5924 in the test set. Each image provides image class marking information, including birds bounding box, key part information, and bird attribute information.

Cars196 [44]: This dataset contains a total of 16185 images of different models of vehicles. When it is used for classification tasks, there are 8054 images in the training set and 8131 images in the test set. When used to metric learning tasks, the first 98 classes are typically used as training sets and the last 98 classes as test sets.

SOP [45]: Stanford online products (SOP) dataset has 22634 classes with 120053 product images. Each product has approximately 5.3 images.

B. Deep Metric Learning Tasks

We employ these three fine-grained image clustering and retrieval benchmarks following the protocol in [24] to split the training and testing sets. For CUB200-2011 dataset, we split the first 100 classes for training (5864 images) and the rest 100 classes for testing (5924 images). For Cars196 dataset, we split the first 98 classes for training (8054 images) and the other 98 classes for testing (8131 images). For the SOP dataset, the first 11318 classes (59551 images) are used for training and the other 11316 (60502 images) classes are used for testing.

NMI, F1, and Recall@K are used as the evaluation metrics. The backbone network is BN-Inception [46] pretrained on

TABLE I
EXPERIMENTAL RESULTS OF DEEP METRIC LEARNING ON CUB200-2011 AND CARS196 DATASETS. NMI, F1, AND RECALL@K ARE REPORTED

Method	CUB200-2011						Cars196					
	NMI	F1	R@1	R@2	R@4	R@8	NMI	F1	R@1	R@2	R@4	R@8
CGML (Triplet)	60.20	27.00	53.30	64.90	75.70	84.50	63.50	32.9	75.8	84.70	90.90	95.20
CGML (N-pair)	60.40	28.50	52.10	64.20	75.40	84.50	62.60	31.00	75.80	84.40	90.50	94.40
IBCDML	74.00	−	70.30	80.30	87.60	92.70	74.80	−	**88.10**	93.30	96.20	98.20
Triplet	59.34	23.12	52.98	65.15	75.30	84.25	56.03	24.94	61.09	70.79	79.47	86.27
+SEC	64.24	30.83	60.82	71.61	81.40	88.86	59.17	25.51	67.89	78.56	85.59	90.99
+DRML	**84.35**	**47.47**	**81.87**	**90.55**	**95.83**	**98.38**	**80.34**	**40.13**	**87.82**	**93.73**	**97.36**	**99.08**
Semitriplet	70.36	41.02	65.89	76.64	85.07	90.76	68.58	37.47	81.08	88.21	92.89	95.59
+SEC	71.62	42.05	67.35	78.73	86.63	91.90	72.67	**44.67**	85.19	91.53	95.28	97.29
+DRML	**84.33**	**48.12**	**82.33**	**90.75**	**95.86**	**98.48**	**78.95**	37.60	**87.66**	**93.64**	**97.16**	**98.87**
NORM N-pair	70.48	40.67	62.06	74.79	84.05	89.68	68.43	38.14	79.09	87.69	93.12	95.58
+SEC	72.24	43.21	66.00	77.23	86.01	91.83	70.61	42.12	82.29	89.60	94.26	97.07
+DRML	**84.54**	**49.50**	**80.27**	**89.52**	**95.27**	**98.18**	**81.28**	**44.17**	**85.97**	**93.16**	**97.22**	**98.95**
Multi-Simi	70.63	41.17	66.58	77.74	85.75	91.67	70.87	42.2	84.43	90.64	94.54	96.91
+SEC	72.85	44.82	68.79	79.42	87.20	92.49	73.95	**46.49**	85.73	91.96	95.51	97.54
+DRML	**84.23**	**47.86**	**81.06**	**89.67**	**94.97**	**98.06**	**80.77**	42.96	**86.42**	**93.47**	**97.36**	**98.98**

ImageNet [47]. We freeze the BatchNorm layers during training and set the parameters following the protocol in [24]. The training and testing rule is that we train each 20 iterations, then test once. The compared loss functions we choose in this paper are triplet loss ($m = 1.0$), semihard triplet loss ($m = 0.2$) [6], normalized N-pair loss ($s = 25$) [19], and multi-similarity loss ($\epsilon = 0.1, \lambda = 0.5, \alpha = 2, \beta = 40$) [21]. For our DRML model, geometry flow layer applies to the first two convolution layers combined with these four loss functions. The hyper-parameter λ is set to $\lambda = 0.5$.

C. Evaluation Results

We evaluate four baseline loss functions (triplet loss, semi-triplet loss, normalized N-pair loss, and multi-similarity loss), and two regularization constraints (SEC [24] and DRML) on clustering tasks. Meanwhile, we compared our method with two recently proposed methods, CGML [26] and IBCDML [27], which also considered the feature distribution of each mini-batch. The comparison results are shown in Tables I and II. As shown in these tables, the semihard triplet loss outperforms triplet loss on all these datasets, which shows that designing the new hard example mining strategies is positive to the clustering tasks. Normalized N-pair loss performs better than triplet loss as well as semihard triplet loss in most cases, since a sample can compare with more than one negative samples in the optimization. Moreover, multi-similarity loss outperforms all these three baseline losses as it constructs more than one similarities among samples. Further, for the spherical embedding constraint (SEC), it boosts the performance of these four baseline loss functions in all cases, which means that constraining the embeddings on one hypersphere is effective to the learning tasks. Our proposed DRML performs best compared with all the other loss functions on the three datasets. For example, on CUB200-2011 dataset, DRML shows a huge improvement on NMI, F1, and R@K compared with the SEC by 20.11%, 16.64%, 21.05%, 18.94%, 14.43%, and 9.52%, respectively. Among all these comparison methods, the results of IBCDML are closest to our method. IBCDML used Euclidean metric to construct a message passing network to uncover the intra relation of each mini-batch. However, it did not consider the non-Euclidean structure of datasets.

TABLE II
EXPERIMENTAL RESULTS OF DEEP METRIC LEARNING ON SOP DATASET. NMI, F1, AND RECALL@K ARE REPORTED

Method	SOP					
	NMI	F1	R@1	R@10	R@100	R@1000
CGML (Triplet)	87.10	23.20	64.10	79.50	90.20	96.80
CGML (N-pair)	88.10	27.00	68.40	84.30	93.20	97.80
IBCDML	92.60	−	81.40	91.30	95.90	−
Triplet	88.72	30.26	63.19	80.78	92.02	97.63
+SEC	89.68	34.29	68.86	83.76	92.93	98.00
+DRML	**93.12**	**49.36**	**77.06**	**95.31**	**98.83**	**98.92**
Semitriplet	91.21	42.03	74.97	88.45	95.93	98.48
+SEC	91.72	44.90	**77.59**	90.12	96.04	98.80
+DRML	**92.25**	**44.90**	76.34	**94.53**	**98.83**	**98.91**
NORM N-pair	91.04	41.57	74.69	87.98	95.21	98.43
+SEC	91.49	43.75	**76.89**	89.64	95.77	98.68
+DRML	**92.43**	**45.91**	75.18	**94.66**	**98.83**	**98.91**
Multi-Simi	91.35	43.67	76.56	89.48	95.38	98.45
+SEC	91.89	46.04	**78.67**	90.77	96.15	98.76
+DRML	**92.68**	**46.90**	77.53	**95.23**	**98.84**	**98.93**

Specially, compared with these four baseline loss functions, the performance of our proposed method is comparatively sta-

ble on these four retrieval benchmarks. On the one hand, this shows the superiority of DRML as it constrains the hidden layer representations to preserve the intrinsic manifold structure of the original dataset. On the other hand, it also demonstrates that the geometric structure of dataset is far more important for deep metric learning tasks than designing different pair-based loss functions.

D. Ablation Study and Analysis

In this following, we evaluate our DRML in various settings and provide detailed discussion through comparison experiments.

Effects of Hyper-Parameter λ: We perform an ablation study on the hyper-parameter λ, which controls the influence of geometric structure to hidden embeddings. In Fig. 3, we compare the performance (NMI, F1, R@1) with $\lambda = \{0.2, 0.3, \ldots, 0.9\}$. We find that our DRML model is robust across different λ. To verify the effectiveness of geometry flow layer, we employ the triplet loss and normalized N-pair loss under $\lambda = \{0, 0.2, 0.3, 0.4\}$. Note that $\lambda = 0$ means no geometric structure is used, which is identical to the baseline. In Fig. 4, we compare the performance R@1 of triplet loss and normalized N-pair loss under different λ values. We observe that for the two loss functions with $\lambda > 0$, the performance outperforms the case with $\lambda = 0$ by a large margin.

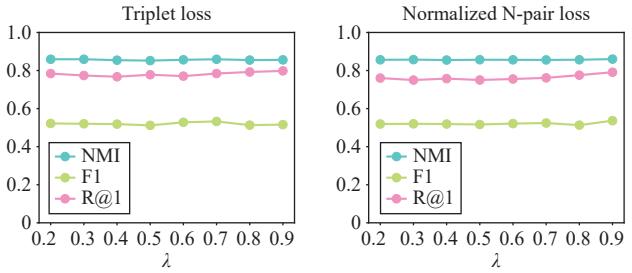


Fig. 3. Effects of the λ on CUB200-2011 dataset. (a) The NMI, F1, and R@1 performances of triplet loss with different λ values; (b) The NMI, F1, and R@1 performances of normalized N-pair loss with different λ values.

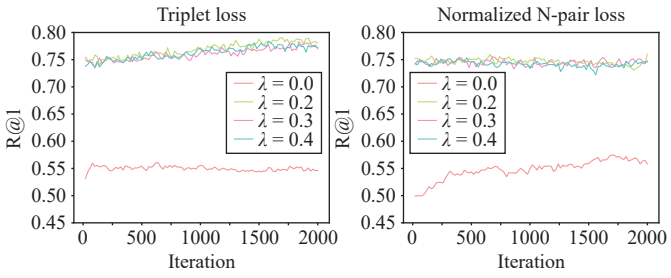


Fig. 4. Testing R@1 on CUB200-2011 about the effects of λ. (a) The R@1 performance of triplet loss with $\lambda = 0, 0.2, 0.3, 0.4$; (b) The R@1 performance of normalized N-pair loss with $\lambda = 0, 0.2, 0.3, 0.4$.

Furthermore, to verify the effectiveness of geometry flow layer for NMI and F1, we employ the triplet loss and normalized N-pair loss under $\lambda = \{0, 0.2, 0.3, 0.4\}$ on CUB200-2011 dataset. In Fig. 5, we observe that for these two loss functions with $\lambda > 0$, the performances entirely outperform the case with $\lambda = 0$. In addition, the convergence rate of NMI with $\lambda > 0$ is faster than the baseline.

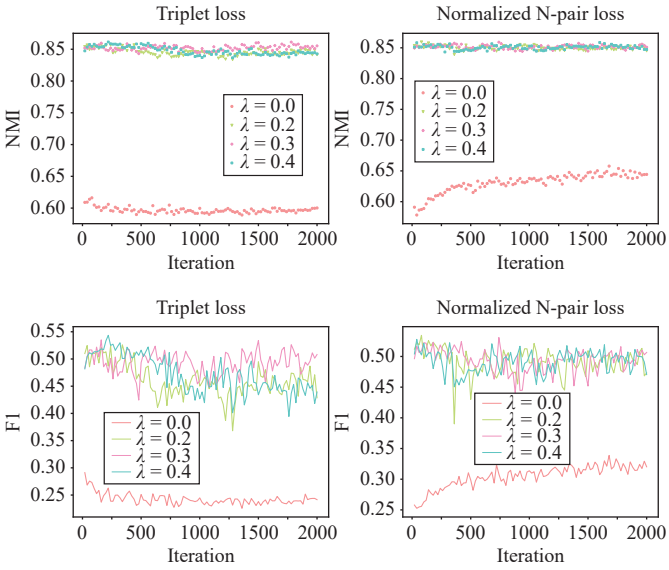


Fig. 5. Testing NMI and F1 about the effects of the λ on CUB200-2011 dataset. (a) The NMI performance of triplet loss with $\lambda = 0, 0.2, 0.3, 0.4$; (b) The NMI performance of normalized N-pair loss with $\lambda = 0, 0.2, 0.3, 0.4$; (c) The F1 performance of triplet loss with $\lambda = 0, 0.2, 0.3, 0.4$; (d) The F1 performance of normalized N-pair loss with $\lambda = 0, 0.2, 0.3, 0.4$.

Effects of Geodesic Curvature: We compare our method on triplet loss and semitriplet loss with the geodesic curvature and Euclidean metric (geodesic curvature is set to zero in geometry flow layer), and set $\lambda = 0.5$. The comparison results are shown in Table III. From Table III, the R@K performances without geodesic curvature are lower than those of our proposed method in all cases. In addition, the Laplacian smoothing regularization based on Euclidean metric (zero curvature) achieves higher performances compared with the traditional methods without geometry regularization. These results show that the geometric structure regularization is necessary to deep metric learning and the geodesic curvature regularization can boost the geometric structure precision of data.

TABLE III
EXPERIMENTAL RESULTS OF DEEP METRIC LEARNING ON CARS196 DATASET (WITH AND WITHOUT GEODESIC CURVATURE)

Method	Cars196			
	R@1	R@10	R@100	R@1000
Triplet	61.09	70.79	79.47	86.27
+DRML (without curvature)	80.23	85.65	93.36	97.49
+DRML (with curvature)	**87.82**	**93.73**	**97.36**	**99.08**
Semitriplet	81.08	88.21	92.89	95.59
+DRML (without curvature)	84.43	91.75	95.84	97.16
+DRML (with curvature)	**87.66**	**93.64**	**97.16**	**98.87**

Effects of Mini-Batch Size: In this part, we analyze how the mini-batch size affects our DRML. We test our method on triplet loss and normalized N-pair loss with mini-batch size increasing from 20 to 90 under $\lambda = 0$ and $\lambda = 0.2$. In Fig. 6(a), the NMI performance grows with the mini-batch size increas-

ing and converges until the mini-batch size reaches 60. In Fig. 6(b), the F1 saturates before mini-batch size reaches 60 and then it declines slightly. These results indicate that with a fixed mini-batch size 60, we can obtain a significant performance boost compared with the baseline.

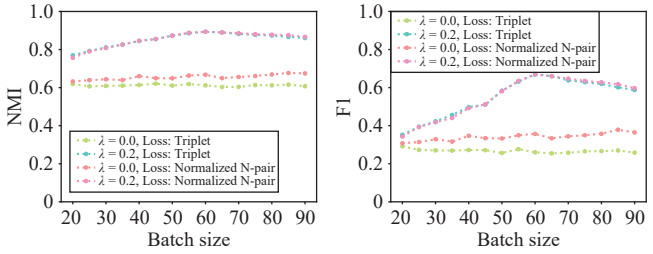


Fig. 6. Testing NMI, F1 on CUB200-2011 dataset about the effects of the mini-batch size.

Convergence Rate: We analyze the convergence of deep metric learning based on two losses with and without geometry flow layer, and the comparison results are shown in Fig. 7. First, from Fig. 7(a), we can see that when combined with geometry flow layer, our DRML converges much faster than that without geometry flow layer under two different learning rates. Moreover, compared with the original loss functions, our DRML model can obtain a much better performance with fewer iterations. Second, from Fig. 7(b), when testing R@1 convergence rate on semihard triplet loss, we can obtain the similar results to Fig. 7(a). Our DRML consistently outperforms the baseline loss on all the settings.

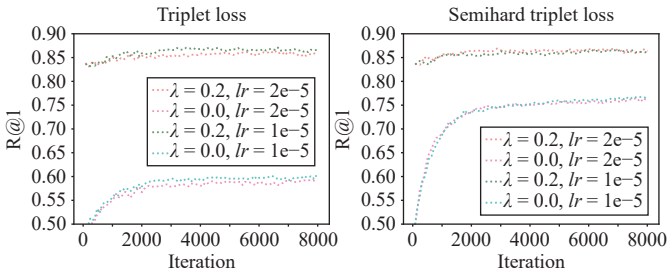


Fig. 7. Testing R@1 convergence rate on Cars196 dataset with $\lambda = 0.2$ and $\lambda = 0$ with two baseline losses, triplet loss and semihard triplet loss.

In addition, because of the operation of the nonlinear activation function, it leads to the features distribution shift of the data in the hidden layers, which can affect the convergence speed of the model. In contrast, we add the geometric flow layers into the first two covolutional layers, which can correct the geometric structure of the hidden features well at the first iteration. Therefore, the performance of DRML is well even at the initial iteration and convergence is also reached soon.

Training and Testing Speed Comparison: We compare our method and the method without geometric flow layer about the time cost on training and testing phrase. In this part, we employ the triplet loss as the baseline, and set $\lambda = 0.2$. We evaluate the corresponding time cost on Cars196 and CUB200-2011 datasets. The comparison results are shown in Fig. 8. For training speed, we calculate the average time cost on each epoch with 2000 iterations in total and the corresponding result is shown in Fig. 8(a). For testing speed, we calculate the average time cost on testing set with 100 times in total, shown in Fig. 8(b). From Figs. 8(a) and 8(b), we can see that the training and testing speed of our method is lower than triplet loss due to the addition of geometric flow layers. However, the baseline methods without geometric flow layer need nearly 8000 iterations to reach the best performance in all. While, our proposed method only needs 2000 iterations to achieve the best result. The total iterations time comparison is shown in Fig. 8(c). The overall time cost of the baseline method is far beyond our method.

VI. Conclusion

In this paper, we introduce a deep Riemannian metric learning method and design a geometry flow layer for deep neural networks that overcomes several drawbacks of existing deep metric learning methods. Our approach combines the geometry structure of dataset with the adaptive capability of learnable methods. In theory, we prove that the proposed method can make a positive influence on the convergence of deep neural networks. Finally, we test the effectiveness of our method on unsupervised clustering tasks on several popular computer vision datasets. Results show that the proposed DRML significantly outperforms the existing deep metric learning methods.

We construct an adjacency graph to represent the discrete manifold, which can not only reflect the topological similarity among image pairs, but also conceal the semantic similarity between non-adjacency images. However, for high-noise data, such as ImageNet, just using normalized distance to measure the local similarity between images is inaccurate, so that the corresponding adjacency graph cannot quantify the intrinsic distribution of data. The geometry flow layer based on the adjacency graph cannot generate gains for this kind of data. Therefore, in the next work, we propose to construct more robust pre-representations of features and learn a more accurate distance metric to construct adjacency graphs on these datasets. Furthermore, we seek to capture the underlying semantic similarity of data by leveraging geometry flow-based analysis to improve the performance of self-supervised learning or deep unsupervised learning.

Appendix
The Evaluation of Geodesic Curvature

Suppose the input data is represented as a set of n image data $\{x_1, x_2, \ldots, x_n\}$, where $x_i \in \mathbb{R}^{D_1 \times D_2}$. Each image feature is represented by a pixel matrix with size $D_1 \times D_2$. Set a parameter δ, if $\|x_i - x_j\|_2 \le \delta$, then x_i and x_j are neighbors. In this paper, we just evaluate the geodesic curvature between neighbors. First, suppose x_i and x_j are neighbors, we respectively obtain the neighborhood sets of x_i and x_j, represented as N_i and N_j. Second, we compute the covariance matrices of N_i and N_j respectively expressed as C_i, C_j. Then, use principal component analysis to compute the d_i and d_j principal component vectors of N_i and N_j, shown as $J_i = \{J_{i_1}, J_{i_2}, \ldots, J_{i_{d_i}}\}$, $J_j = \{J_{j_1}, J_{j_2}, \ldots, J_{j_{d_j}}\}$. The corresponding principal angle $\theta(J_i, J_j)$ between C_i and C_j evaluates as follows:

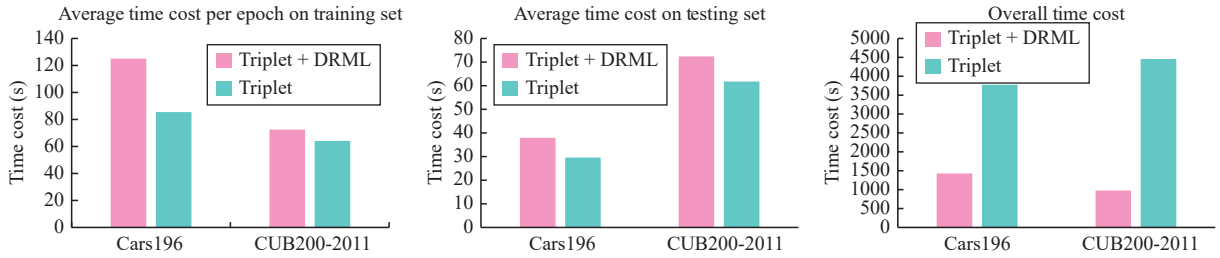


Fig. 8. The training and testing speed comparison about the effects of geometric flow layer on CUB200-2011 and Cars196 datasets. (a) The training speed comparison of our method and triplet loss without geometric flow layer; (b) The testing speed comparison of our method and triplet loss without geometric flow layer; (c) The overall time cost comparison of our method and triplet loss without geometric flow layer.

$$\cos\left(\theta\left(J_i, J_j\right)\right) = \max\{\frac{< J_{i_k}, J_{j_l} >}{||J_{i_k}|| \cdot ||J_{j_l}||} : J_{i_k} \in J_i, J_{j_l} \in J_j\} \tag{A1}$$

$$\theta\left(J_i, J_j\right) = \arccos\left(\theta\left(J_i, J_j\right)\right). \tag{A2}$$

After the deep neural network, the features in the hidden layer are represented as a set of 3D tensors. Suppose the corresponding features of the input data are represented as $\{X_1, X_2, \ldots, X_n\}$ in the i-th hidden layer, where $X_j \in \mathbb{R}^{c\times h\times w}$, c is the number of channel, and $h \times w$ is the matrix size of each channel image. The principal angle between two neighborhood tensors X_j and X_k is evaluated as follows:

$$R\left(X_j, X_k\right) = \frac{||\theta_{jk}||}{d\left(X_j, X_k\right)} \tag{A3}$$

where θ_{jk} is the principal angle between X_j and X_k, and $d(X_j, X_k)$ is the Euclidean distance between X_j and X_k.

In the following, we give the evaluation of θ_{jk}. For each channel, suppose the feature set is represented as $\{X_1^m, X_2^m, \ldots, X_n^m\}, X_j^m \in \mathbb{R}^{h\times w}, m = 1, 2, \ldots, c$. In each channel, according to (A2), the principal angle between X_j^m and X_k^m is represented as $\theta(J_j^m, J_k^m)$, where J_j^m, J_k^m are the tangent spaces of X_j^m and X_k^m. Then, the principal angle θ_{jk} is shown as follows:

$$\theta_{jk} = \sum_{m=1}^{c} \theta\left(J_j^m, J_k^m\right). \tag{A4}$$

REFERENCES

[1] J. B. Tenenbaum, V. De Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000.

[2] B. Lin, X. He, and J. Ye, "A geometric viewpoint of manifold learning," *Applied Informatics*, 2015.

[3] E. Xing, M. Jordan, S. J. Russell, and A. Ng, "Distance metric learning with application to clustering with side information," *Advances in Neural Information Processing Systems*, vol. 15, pp. 521–528, 2002.

[4] K. Q. Weinberger and L. K. Saul, "Distance metric learning for large margin nearest neighbor classification," *Journal of Machine Learning Research*, vol. 10, p. 2, 2009.

[5] J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon,"Information-theoretic metric learning," in *Proc. 24th Int. Conf. Machine Learning*, pp. 209–216, 2007.

[6] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 815–823, 2015.

[7] Y. Jin, Y. Dong, Y. Zhang, and X. Hu, "SSMD: Dimensionality reduction and classification of hyperspectral images based on spatial-spectral manifold distance metric learning," *IEEE Trans. Geoscience and Remote Sensing*, vol. 60, p. 5538916, 2022.

[8] Y. Wen, K. Zhang, Z. Li, and Y. Qiao, "A discriminative feature learning approach for deep face recognition," in *Proc. European Conf. Computer Vision*, pp. 499–515, Springer, 2016.

[9] X. He, Y. Zhou, Z. Zhou, S. Bai, and X. Bai, "Triplet-center loss for multi-view 3D object retrieval," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 1945–1954, 2018.

[10] H. Wang, Y. Wang, Z. Zhou, X. Ji, D. Gong, J. Zhou, Z. Li, and W. Liu, "Cosface: Large margin cosine loss for deep face recognition," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 5265–5274, 2018.

[11] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and Vandergheynst, "Geometric deep learning: Going beyond euclidean data," *IEEE Signal Processing Magazine*, vol. 34, no. 4, pp. 18–42, 2017.

[12] J. H. White and R. W. Beard, "An iterative pose estimation algorithm based on epipolar geometry with application to multi-target tracking," *IEEE/CAA J. Autom. Sinica*, vol. 7, no. 4, pp. 942–953, 2020.

[13] D. Ye, B. Desjardins, J. Hamm, H. Litt, and K. M. Pohl, "Regional manifold learning for disease classification," *IEEE Trans. Medical Imaging*, vol. 33, no. 6, pp. 1236–1247, 2014.

[14] X. Chen, J. Weng, W. Lu, J. Xu, and J. Weng, "Deep manifold learning combined with convolutional neural networks for action recognition," *IEEE Trans. Neural Networks and Learning Systems*, vol. 29, no. 9, pp. 3938–3952, 2017.

[15] D. Wu and X. Luo, "Robust latent factor analysis for precise representation of high-dimensional and sparse data," *IEEE/CAA J. Autom. Sinica*, vol. 8, no. 4, pp. 796–805, 2020.

[16] D. Li, W. Hung, J. Huang, S. Wang, N. Ahuja, and M. Yang, "Unsupervised visual representation learning by graph-based consistent constraints," in *Proc. European Conf. Computer Vision*, pp. 678–694, 2016.

[17] Y. Li and R. Lu, "Riemannian metric learning based on curvature flow," in *Proc. 24th Int. Conf. Pattern Recognition*, pp. 806–811, 2018.

[18] G. Taubin, "A signal processing approach to fair surface design," in *Proc. 22nd Annu. Conf. Computer Graphics and Interactive Techniques*, pp. 351–358, 1995.

[19] K. Sohn, "Improved deep metric learning with multi-class n-pair loss objective," in *Advances in Neural Information Processing Systems*, pp. 1857–1865, 2016.

[20] W. Ge, "Deep metric learning with hierarchical triplet loss," in *Proc. European Conf. Computer Vision*, pp. 269–285, 2018.

[21] X. Wang, X. Han, W. Huang, D. Dong, and M. R. Scott, "Multi-similarity loss with general pair weighting for deep metric learning," in *Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition*, pp. 5022–5030, 2019.

[22] E. Ustinova and V. Lempitsky, "Learning deep embeddings with histogram loss," arXiv preprint arXiv: 1611.00822, 2016.

[23] J. Wang, F. Zhou, S. Wen, X. Liu, and Y. Lin, "Deep metric learning with angular loss," in *Proc. IEEE Int. Conf. Computer Vision*, pp. 2593–2601, 2017.

[24] D. Zhang, Y. Li, and Z. Zhang, "Deep metric learning with spherical embedding," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[25] W. Zhao, Y. Rao, Z. Wang, J. Lu, and J. Zhou, "Towards interpretable deep metric learning with structural matching," in *Proc. IEEE/CVF Int.*

Conf. Computer Vision, pp. 9887–9896, 2021.

[26] B. Chen, P. Li, Z. Yan, B. Wang, and L. Zhang, "Deep metric learning with graph consistency," in *Proc. 35th AAAI Conf. Artificial Intelligence*, pp. 982–990, 2021.

[27] J. Seidenschwarz, I. Elezi, and L. Leal-Taixe, "Learning intra-batch connections for deep metric learning," in *Proc. 38th Int. Conf. Machine Learning*, 2021.

[28] Y. Ollivier, "Ricci curvature of markov chains on metric spaces," *Journal of Functional Analysis*, vol. 256, no. 3, pp. 810–864, 2009.

[29] K. I. Kim, J. Tompkin, and C. Theobalt, "Curvature-aware regularization on riemannian submanifolds," in *Proc. IEEE Int. Conf. Computer Vision*, pp. 881–888, 2013.

[30] D. Gong, X. Zhao, and G. Medioni, "Robust multiple manifolds structure learning," arXiv preprint arXiv: 1206.4624, 2012.

[31] Y. Li, "Curvature-aware manifold learning," *Pattern Recognition*, vol. 83, pp. 273–286, 2018.

[32] W. Xu, E. R. Hancock, and R. C. Wilson, "Ricci flow embedding for rectifying non-euclidean dissimilarity data," *Pattern Recognition*, vol. 47, no. 11, pp. 3709–3725, 2014.

[33] H. Li, J. Cao, J. Zhu, Y. Liu, Q. Zhu, and G. Wu, "Curvature graph neural network," *Information Sciences*, vol. 592, pp. 50–66, 2022.

[34] Z. Ye, K. S. Liu, T. Ma, J. Gao, and C. Chen, "Curvature graph network," in *Proc. Int. Conf. Learning Representations*, 2019.

[35] C. Ni, Y. Lin, J. Gao, and X. Gu, "Network alignment by discrete ollivier-Ricci flow," in *Proc. Int. Symp. Graph Drawing and Network Visualization*, pp. 447–462, Springer, 2018.

[36] C. Ni, Y. Lin, F. Luo, and J. Gao, "Community detection on networks with Ricci flow," *Scientific Reports*, vol. 9, no. 1, pp. 1–12, 2019.

[37] N. G. J. David, T. Guarrera, and H. F. Wolfe, "The tayer expansion of a riemannianmetric,"[online]Available:https://studylib.net/doc/13872779/the-taylor-expansion-of-a-riemannian-metric, 2002.

[38] B. Chow and D. Knopf, "The Ricci flow: An introduction," *Mathematical Surveys and Monographs*, vol. 110, 2008.

[39] R. S. Hamilton, "Three-manifolds with positive Ricci curvature," *Journal of Differential Geometry*, vol. 17, no. 2, pp. 255–306, 1982.

[40] S. Brendle and R. Schoen, "Curvature, Sphere theorems, and the Ricci flow," arXiv preprint arXiv: 1001.2278v2, 2010.

[41] Y. Li and R. Lu, "Applying Ricci flow to high dimensional manifold learning," *Science China Information Sciences*, vol. 62, no. 9, pp. 192101:1–192101:14, 2019.

[42] S. Cost and S. Salzberg, "A weighted nearest neighbor algorithm for learning with symbolic features," *Machine Learning*, vol. 10, pp. 57–78, 1993.

[43] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The caltech-ucsd birds-200-2011 dataset," 2011.

[44] J. Krause, M. Stark, J. Deng, and F. Li, "3D object representations for fine-grained categorization," in *Proc. IEEE Int. Conf. Computer Vision Workshops*, pp. 554–561, 2013.

[45] H. Oh Song, Y. Xiang, S. Jegelka, S. Savarese, "Deep metric learning via lifted structured feature embedding," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 4004–4012, 2016.

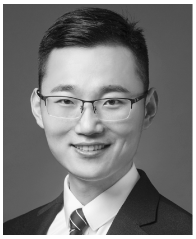
[46] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Machine Learning*, pp. 448–456, 2015.

[47] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and F. Li, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 248–255, 2009.

Yangyang Li received the B.S. degree in math and computer science from the Hebei university in 2012, and the Ph.D. degree from the University of Chinese Academy of Sciences in 2019. From 2019 to 2021, she was a Postdoctoral Researcher at Academy of Mathematics and Systems Science, CAS. She is now an Assistant Researcher at the Academy of Mathematics and Systems Sciences, CAS. Her research interests include pattern recognition, image processing, and geometry machine learning.

Chaoqun Fei received the bachelor degree from Zhengzhou University in 2013, the M.S. degree from Wuhan University in 2015, and the Ph.D. degree from the University of Chinese Academy of Sciences. He is currently a Postdoctoral Researcher at Academy of Mathematics and Systems Science, Chinese Academy of Sciences. His research interests include knowledge graph, machine learning, and biomedical image analysis.

Chuanqing Wang received the bachelor degree from Henan University in 2015. He is currently a Ph.D. candidate of University of Chinese Academy of Sciences. His research focuses on programming language designment and development for knowledge and knowledge graph network.



Hongming Shan (Senior Member, IEEE) received the Ph.D. degree in machine learning from Fudan University in 2017. He is currently an Associate Professor with the Institute of Science and Technology for Brain-Inspired Intelligence, Fudan University, and also a "Qiusuo" Research Leader with the Shanghai Center for Brain Science and Brain-Inspired Technology. From 2017 to 2020, he was a Postdoctoral Research Associate and a Research Scientist at the Rensselaer Polytechnic Institute, USA. His research interests include machine learning, medical imaging, and computer vision. He was recognized with Youth Outstanding Paper Award at World Artificial Intelligence Conference 2021.

Ruqian Lu received the diploma degree in mathematics from Jena University, Germany in 1959. He is an Academician of Chinese Academy of Sciences, Professor with the Institute of Mathematics, Academy of Mathematics and Systems Science. His current research interests include artificial intelligence, knowledge engineering and knowledge based software engineering.

Prof. Lu is holding a concurrent professorship with the Institute of Computing Technology, Chinese Academy of Sciences. He has published more than 200 papers and authored a dozen books. He has received China's National Second Class and CAS's First Class Prize for Progress in Science and Technology (twice), the Hua Luogeng Mathematics Prize from China's Mathematics Society, and the Lifelong Achievement Prize from the China Computer Federation (CCF).