Layered Geometric Learning

Hamideh Hajiabadi[1(✉)], Reza Godaz[2], Morteza Ghasemi[2], and Reza Monsefi[2]

[1] Computer Engineering Department, Birjand University of Technology, Birjand, Iran
hamideh.hajiabadi@mail.um.ac.ir

[2] Computer Engineering Department, Ferdowsi University of Mashhad (FUM), Mashhad, Iran
{reza.godaz,morteza.ghasemi}@mail.um.ac.ir, monsefi@um.ac.ir

Abstract. Through Metric learning techniques, a metric function is learned, which shows how similar/dissimilar two samples are. From the perspective of feature selection, metric learning can be represented as a transform function mapping each sample into a new point in the new feature space. Geometric Mean Metric Learning (GMML) is one of promising methods which achieve good performance in terms of accuracy and time complexity. In this paper, we propose the use of GMML algorithm in a neural network to perform Riemannian computing on the SPD matrices which improves accuracy and reduces time complexity. We also use the eigenvalue rectification layer as a non-linear activation function to enhance the non-linearity of our model. Experimental evaluations on several benchmark data sets demonstrate that the proposed method improves accuracy in comparison with the state-of-the-art approaches.

Keywords: Metric learning · Geometric metric learning · Artificial Neural Network

1 Introduction

Metric Learning is a kind of data transform method, which makes similar instances closer and dissimilar ones farther. The transformed data is later used in learning algorithms (e.g., classification, regression, etc.). Metric learning can be interpreted as a feature learning [17] which maps data to a new space hoping that in the new feature space, data would be better represented (e.g., Mahalanobis distance metric). Metric learning which learns the similarity/distance metric from the annotated data is of significant practical importance, which can be considered as a pre-process of variety tasks, e.g., classification, clustering, feature extraction, feature matching, etc., [7,13,18,25]. Moreover, metric learning approaches can overcome the challenges of extreme classification [8].

Currently, best metric learning approaches make use of state-of-the-art Artificial Neural Networks (ANN), which produces the best embedding by minimizing a loss function (which is usually related to the similarity/ distance of the points) [20,21]. However, most of these techniques learn a Mahalanobis distance in the

L. Rutkowski et al. (Eds.): ICAISC 2019, LNAI 11508, pp. 571–582, 2019.
https://doi.org/10.1007/978-3-030-20912-4_52

Euclidean space, which can be interpreted as a linear mapping of the data. These techniques have achieved an improvement in both modeling and the algorithm.

Most deep metric learning approaches learn a Euclidean metric in a mapped space [2]. The mapped space is either a linear transform of the original space or a non-linear one. Let $x \in \mathcal{R}^d$ be an instance in the origin space and $\phi(x)$ be the corresponding instance in the mapped space, where $\phi(.)$ is either a linear function represented by $\phi(x) = A \times x$ or a non-linear function. The conventional Euclidean distance $\|\phi(x) - \phi(y)\|$ can be still used in the mapped space to estimate how dissimilar two instances are [22,26]. More precisely, letting $\boldsymbol{x}, \boldsymbol{y}$ denote two samples in $\mathcal{R}^d$ space, we denote $(\boldsymbol{x}-\boldsymbol{y})M(\boldsymbol{x}-\boldsymbol{y})^T$ as the distance between those two samples where $M \in \mathcal{R}^{d \times d}$ is a Symmetric Positive Definite (SPD) matrix. The distance can be interpreted as a Euclidean distance, $\|\phi(x) - \phi(y)\|$, in the mapped space $\phi(.)$ where $\phi(x) = M^{\frac{1}{2}}x$. Since the distance is positive, the matrix learned through the metric learning ought to be an SPD matrix to confirm the positiveness of the distance. Some researchers make the learned matrix, symmetric-positive definite by setting the negative eigenvalues to 0, which might lead to ambiguity. This ambiguity can be prevented by making matrix space a Riemannian manifold.

In this paper, we revisit the structure of neural network and present a new ANN architecture based on geometric learning algorithms. In our proposed model, several metric learning layers are used. Each metric layer can be a new representation of a metric learning algorithm having a closed-form optimization on an SPD manifold. Through comprehensive experiments, we show the use of SPD manifold improves the performance of a ANN by experimentally evaluation on several benchmark datasets. Our main contribution is the use of geometric learning algorithms as new mertic layers in a neural network. These new layers include a Geometric Metric Mean Learning (GMML) as a transform function followed by a ReEig layer [15] which transforms data into a new space.

The rest of this paper is organized as follows. An overview of related work is briefly introduced in Sect. 2. We describe our proposed model in Sect. 3. Experimental evaluations are illustrated in Sect. 4. Finally, Sect. 5 contains conclusion with possible remarks for future works.

2 Related Works

In this section, we briefly review the promising deep distance metric learning algorithms and then concentrate on the geometric learning.

2.1 Deep Metric Learning

Since 2014 deep metric learning have been attracted by many researchers [9,10,14,19,22,24] and the idea of integrating metric learning into deep networks was first proposed in 1994 [6]. Faraki et al. combine geometrically dimension reduction and metric learning method and then integrate it into a deep framework [11]. They use Riemannian manifolds in their optimization algorithm [1] and achieve improvements.

In paper [14], a nonlinear manifold of similar face images by applying distance metric learning approaches into deep learning is learned. A joint loss function containing a logistic loss and a regularization term of network weights and biases are used. A new structure for feature embedding which learns full advantages of batches through training phase was proposed in 2016 [18]. First, some positive pairs are randomly selected and then the distances between those selected pairs and all negative pairs are calculated using $log - sum - exp$ formula.

An interesting metric learning approach which learns a map function transferring each sample to a new point in the mapped space was proposed in 2017 [22]. Some benefits of this work are as follows:- (1) A new loss is proposed which is not based on pairs or triples; therefore, data are not needed to be pre-processed to extract pairs or triples like other existing metric learning approaches. (2) During learning of embedding space, the network is encouraged to optimize a global metric for clustering; this method uses the global structure of embedding space to learn a quality metric for clustering.

An online deep metric learning framework was proposed in 2018 [16]. It consists of several metric layers in a neural network and each metric layer is actually an existing online metric learning algorithm which can be optimized in a closed form. Each metric layer is followed by a nonlinear function like *ReLu*. Let x^0, x^1 are the input and the output of the first metric layer respectively, the output is calculated as $x^1 = L^t x^0$ while $L^t L = M$ and M is the metric matrix which is calculated in a closed form optimization. The network is only updated through the forward pass. In the next subsection, We briefly overview some pioneer works that apply geometry to metric learning approaches.

2.2 Geometric Metric Learning

In 2016, a Mahalanobis-Based cost function named Geometric Mean Metric Learning (GMML) was proposed [26]. They revisited the convenient Euclidean-based optimization procedure and proposed a new geometric learning method on SPD manifolds. They then reached a geometrical closed-form solution for the Metric Learning problem, which significantly reduces the time complexity. In [23] a new local method based on GMML named L-GMML was introduced and applied to the task of ranking. Some local matrices and a corresponding anchor document were first learned and then the anchors were weighted.

A new Riemannian network architecture for deep networks using SPD matrices was proposed in [15]. They introduced some new geometric layers such as bi-linear mapping layers (BiMap), eigenvalue rectification layers (ReEig), and an eigenvalue logarithm layer (LogEig). In the following, these three layers are briefly explained.

BitMap Layer. The BiMap layer transforms an input SPD matrix to a new more compact matrix with higher ability to discriminate data. This layer uses a bi-linear mapping function f_b as follows

$$\boldsymbol{X}^k = f_b(\boldsymbol{X}^{k-1}; \boldsymbol{W}^k) = \boldsymbol{W}^k \boldsymbol{X}^{k-1} \boldsymbol{W}^{k^T} \tag{1}$$

where $\boldsymbol{X}^{k-1} \in \mathrm{Sym}^{+}_{d_{k-1}}$ is the input matrix of k-th layer, $\boldsymbol{W}^{k} \in R^{d_k \times d_{k-1}}_{*}, d_k < d_{k-1}$ is the connection weights and $\boldsymbol{X}^{k} \in R^{d_k \times d_k}$ is the output of the layer.

ReEig Layer. The ReEig layer makes input SPD matrices far away from non-positive ones and it is formulated as follows

$$\boldsymbol{X}^{k} = f_r^{(k)}(\boldsymbol{X}^{k-1}) = \boldsymbol{U}^{k-1} \max(\epsilon \boldsymbol{I}, \boldsymbol{\Sigma}^{k-1}) \boldsymbol{U}^{k-1^{T}} \tag{2}$$

where $\boldsymbol{U}^{k-1}$ and $\boldsymbol{\Sigma}^{k-1}$ are the output of the eigenvalue decomposition and $\boldsymbol{X}^{k-1} = \boldsymbol{U}^{k-1} \boldsymbol{\Sigma}^{k-1} \boldsymbol{U}^{\boldsymbol{k-1}^{\boldsymbol{T}}}$. ϵ denotes as a rectification threshold, $\boldsymbol{I}$ is Identity matrix and $\max(\epsilon \boldsymbol{I}, \boldsymbol{\Sigma}^{k-1})$ is a diagonal matrix which is defined as follows:-

$$\boldsymbol{A}(i,i) = \begin{cases} \boldsymbol{\Sigma}^{k-1}(i,i) & , \boldsymbol{\Sigma}^{k-1}(i,i) > \epsilon \\ \epsilon & , \boldsymbol{\Sigma}^{k-1}(i,i) \leq \epsilon \end{cases} \tag{3}$$

LogEig Layer. The LogEig layer is defined as the following.

$$\boldsymbol{X}^{k} = f_l^{(k)}(\boldsymbol{X}^{k-1}) = \log(\boldsymbol{X}^{k-1}) = \boldsymbol{U}^{k-1} \log(\epsilon \boldsymbol{I}, \boldsymbol{\Sigma}^{k-1}) \boldsymbol{U}^{k-1^{T}} \tag{4}$$

where $\log(\boldsymbol{X}^{k-1})$ is the logarithm of diagonal elements. According to [3] this metric layer (Log-Euclidean Riemannian metric) provide a lie group structure to the Riemannian manifold of SPD matrices. As a result, the SPD manifold is reduced to a flat space in which conventional Euclidean computations can be simply conducted and there is no need to take the pain to do Riemannian operations such as geodesic calculations.

Riemannian Manifold Metric Learning (RMML) aims to reduce the geodesic distance of similar pairs while increasing the geometric distance of dissimilar ones on nonlinear manifolds. RMML is extended for both SPD and Grassmann manifolds, [27].

3 The Proposed Model

As discussed earlier, the aim of the metric learning approaches is to eventually obtain a metric that gives "small" distance for similar points and "large" distance for dissimilar ones. Different metric learning approaches are willing to fulfill this guideline implicitly or explicitly. Figure 1 shows the impact of learning a metric matrix M based on Mahalanobis distance.

In the Mahalanobis-based metric learning approaches, it is intended to find a matrix M through training stage where distance between ith and jth samples are defined as $d_M = (\boldsymbol{x}_i - \boldsymbol{x}_j)^T M (\boldsymbol{x}_i - \boldsymbol{x}_j)$. Matrix M must be an SPD matrix, so

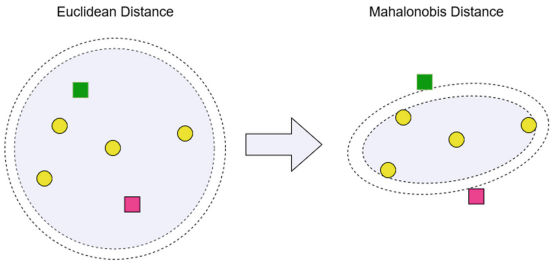

Fig. 1. Mahalanobis-based metric learning

there is a matrix W such that $W^TW = M$. We revisit the Mahalanobis distance as follows

$$\begin{aligned} d_M &= (\boldsymbol{x}_i - \boldsymbol{x}_j)^T W^T W (\boldsymbol{x}_i - \boldsymbol{x}_j) \\ &= (W\boldsymbol{x}_i - W\boldsymbol{x}_j)^T (W\boldsymbol{x}_i - W\boldsymbol{x}_j) \\ &= \|W\boldsymbol{x}_i - W\boldsymbol{x}_j\|_2^2. \end{aligned} \tag{5}$$

We interpret $W\boldsymbol{x}_i$ as the transformed point of the original point $\boldsymbol{x}_i$ and $\|W\boldsymbol{x}_i - W\boldsymbol{x}_j\|_2^2$ as the Euclidean distance in the transformed space. So, using metric learning algorithms, data are implicitly transformed to a new space, hoping that the data would be more discriminated in the transformed space.

We propose incorporating metric learning algorithms into neural networks by introducing a geometric layer. Let $x^{(i)}$ and $x^{(i+1)} = Wx^{(i)}$ be the input and the output of the ith metric layer respectively where $M = W^tW$ is the metric matrix. Matrix M is an SPD matrix which is learned through a closed-form optimization on the SPD manifold. We use a closed-form geometric metric optimization algorithm (GMML) that has been proposed in [26]. In the following, we explain the process that has been conducted in each metric layer.

In each metric layer, we wish to find a matrix M that decrease the sum of distances over all similar points while M^{-1} increase the sum of distances over dissimilar pairs simultaneously. We propose the use of the following objective function like that was proposed by [26].

$$\min_{M \succ 0} \sum_{(\boldsymbol{x}_i, \boldsymbol{x}_j) \in \mathcal{S}} d_M(\boldsymbol{x}_i, \boldsymbol{x}_j) + \sum_{(\boldsymbol{x}_i, \boldsymbol{x}_j) \in \mathcal{D}} d_{M^{-1}}(\boldsymbol{x}_i, \boldsymbol{x}_j) \tag{6}$$

where $\mathcal{S}$, $\mathcal{D}$ are sets of similar indices and dissimilar indices respectively. $d_M(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is the distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ and defined as $(\boldsymbol{x}_i - \boldsymbol{x}_j)M(\boldsymbol{x}_i - \boldsymbol{x}_j)^T$.

In the Euclidean space, the gradient of $d_M(\boldsymbol{x}_i, \boldsymbol{x}_j)$ with respect to M is

$$(\boldsymbol{x}_i, \boldsymbol{x}_j)(\boldsymbol{x}_i, \boldsymbol{x}_j)^T$$

while the gradient of $d_{M^{-1}}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is

$$-M^{-1}(\boldsymbol{x}_i, \boldsymbol{x}_j)(\boldsymbol{x}_i, \boldsymbol{x}_j)^T M^{-1}$$

The inner product of those two gradients is negative, so, they are in the opposite direction and an increase in the gradient of M cause a decrease in that of M^{-1}. Since the first and second terms of the cost function 6 are in the opposite direction, the minimization of the cost function causes small distance for similar pairs and large distance for dissimilar ones.

The Eq. 6 can be reformulated as follows:

$$\begin{aligned}\min_{M\succ 0} \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{S}} & tr\left(M(\boldsymbol{x}_i-\boldsymbol{x}_j)(\boldsymbol{x}_i-\boldsymbol{x}_j)^t\right) \\ & + \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{D}} tr\left(M^{-1}(\boldsymbol{x}_i-\boldsymbol{x}_j)(\boldsymbol{x}_i-\boldsymbol{x}_j)^t\right)\end{aligned} \tag{7}$$

where $tr(.)$ is the summation of diagonal elements. By considering

$$\begin{aligned}S &:= \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{S}} (\boldsymbol{x}_i-\boldsymbol{x}_j)(\boldsymbol{x}_i-\boldsymbol{x}_j)^T, \\ D &:= \sum_{(\boldsymbol{x}_i,\boldsymbol{x}_j)\in\mathcal{D}} (\boldsymbol{x}_i-\boldsymbol{x}_j)(\boldsymbol{x}_i-\boldsymbol{x}_j)^T\end{aligned} \tag{8}$$

in which $\mathcal{S}$ and $\mathcal{D}$ are:

$$\begin{aligned}\mathcal{S} &:= \{(\boldsymbol{x}_i,\boldsymbol{x}_j)|\boldsymbol{x}_i \text{ and } \boldsymbol{x}_j \text{ are in the same class}\}, \\ \mathcal{D} &:= \{(\boldsymbol{x}_i,\boldsymbol{x}_j)|\boldsymbol{x}_i \text{ and } \boldsymbol{x}_j \text{ are in different classes}\}.\end{aligned} \tag{9}$$

The Eq. 7 is briefed as

$$\min_{M\succ 0} \ \mathrm{tr}(MS)+\mathrm{tr}(M^{-1}D). \tag{10}$$

To achieve the optimal solution for Eq. (10), we set its derivative to zero. Derivative of Eq. (10) with respect to matrix M is as follows

$$S-M^{-1}DM^{-1}=0 \Rightarrow MSM=D. \tag{11}$$

To obtain matrix M from the above equation, both $D \succeq 0$ and $S \succeq 0$ should hold, and it results in a positive distance as described in the following:

$$\begin{aligned}&S^{1/2}MSMS^{1/2}=S^{1/2}DS^{1/2} \Rightarrow \\ &(S^{1/2}MSMS^{1/2})^{1/2}=(S^{1/2}DS^{1/2})^{1/2} \Rightarrow \\ &(S^{1/2}MS^{1/2}S^{1/2}MS^{1/2})^{1/2}=(S^{1/2}DS^{1/2})^{1/2} \Rightarrow \\ &(S^{1/2}MS^{1/2})=(S^{1/2}DS^{1/2})^{1/2} \Rightarrow \\ &S^{-1/2}(S^{1/2}MS^{1/2})S^{-1/2}=S^{-1/2}(S^{1/2}DS^{1/2})^{1/2}S^{-1/2} \Rightarrow \\ &M=S^{-1/2}(S^{1/2}DS^{1/2})^{1/2}S^{-1/2}\end{aligned} \tag{12}$$

The last equation is indeed the midpoint of the geodesic joining S^{-1} to D [4]. Therefore, the obtained result automatically satisfies the constraint of $M \succeq 0$. Figure 2 shows the proposed architecture which is a conventional ANN extended with several metric layers.

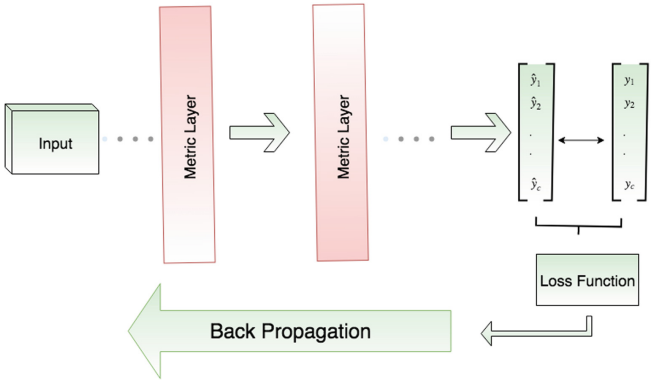

Fig. 2. Our proposed architecture

3.1 The Proposed Geometric Network

We propose a Geometric network based on SPD manifolds. We put a GMML layer as a metric layer followed by a ReEig layer. These successive layers are repeated through the network. It is noted that GMML method is fast and lead to accurate results and can be optimized in a closed-form way.

Typical neural networks apply a Stochastic Gradient Descent (SGD) algorithm for backpropagation that uses the gradient of the loss function and a learning rate to create a descent step [12], which ultimately reduces the value of loss function. In the forward path, a predicted label is produced and then is compared with the desired one to obtain the error. Afterward, the gradient of the loss function flow back through the network and update all the weights in the opposite direction of the gradient to reduce the loss value.

Alternatively, we propose to update the weights of the geometric metric layers by a closed-form optimization in two steps as follows. We first obtain the optimal matrix, M, by optimizing Eq. (7) which leads to Eq. (10). Then, as demonstrated in Eq. (5), the optimal weights of the metric layers are obtained by $W = M^{\frac{1}{2}}$. By doing so, there is no more need to update metric layers' weights through back-propagation.

As it is shown in Eq. (5), each metric layer implicitly transforms data into the new space, $X_{new} = WX_{old}$, hoping that the data in the new space would be more discriminative. W is obtained by the decomposition of the metric matrix M where $M = W^T W$. Two matrices M and W have the same dimension. The optimal value of M is obtained according to Eq. (12). We calculate the output of a metric layer by $x^{k+1} = Wx^k$ where x^k is the input of the layer. The process is shown in Algorithm 1. If each metric layer is convex we will advance of using a closed-form optimization to reach a simple and global optimal value. The prove of convexity for GMML is straightforward, since it is the summation of two convex function [5]. Note that other closed form metric learning algorithm can be used instead of GMML.

Algorithm 1. The GMML Layer

Input of first layer: $\mathcal{S}^0$: set of similar pairs, $\mathcal{D}^0$: set of dissimilar pairs
Input of kth layer: X^{k-1}: output of $k-1$th layer
Output of kth layer: $X^k = W^k X^{k-1}$ where W^k calculates as below

1: Calculate S^{k-1} and D^{k-1} according to

$$\begin{aligned} S^{k-1} &:= \sum_{(\boldsymbol{x}_i^{k-1}, \boldsymbol{x}_j^{k-1}) \in \mathcal{S}^{k-1}} (\boldsymbol{x}_i^{k-1} - \boldsymbol{x}_j^{k-1})(\boldsymbol{x}_i^{k-1} - \boldsymbol{x}_j^{k-1})^T, \\ D^{k-1} &:= \sum_{(\boldsymbol{x}_i^{k-1}, \boldsymbol{x}_j^{k-1}) \in \mathcal{D}^{k-1}} (\boldsymbol{x}_i^{k-1} - \boldsymbol{x}_j^{k-1})(\boldsymbol{x}_i^{k-1} - \boldsymbol{x}_j^{k-1})^T \end{aligned} \tag{13}$$

2: Compute M^k as:

$$M^k = (S^{k-1})^{-1/2}((S^{k-1})^{1/2} D^{k-1} (S^{k-1})^{1/2})^{1/2} (S^{k-1})^{-1/2} \tag{14}$$

3: Decompose matrix $M^k = (W^k)^T (W^k)$ to obtain W^k
4: Compute the output of GMML layer as $X^k = W^k X^{k-1}$

The ReLU layer in typical neural networks includes $max(0, x)$ non-linearity. This layer is used to improve the non-linearity of the network. In the [15], the ReLU layer is replaced by a new geometric layer called ReEig. This layer rectifies the small positive eigenvalues. We use ReEig layer instead of simple ReLu to enhance the non-linearity of the network. This layer has been defined in Eqs. (2) and (3). It prevents the input SPD matrices from having non-positive Eigenvalues. Our proposed GMML layer and ReEig layer has been implemented in the Algorithm 1 and Algorithm 2 respectively.

Algorithm 2. The kth ReEig Layer

Input: X^{k-1}, ϵ: a rectification threshold
Output: X^k

1: Decompose $\boldsymbol{X}^{k-1}$ as $\boldsymbol{X}^{k-1} = \boldsymbol{U}^{k-1} \boldsymbol{\Sigma}^{k-1} (U^{k-1})^T$
2: Calculate $\boldsymbol{X}^k$ as $\boldsymbol{X}^k = f_r^{(k)}(\boldsymbol{X}^{k-1}) = \boldsymbol{U}^{k-1} \max(\epsilon \boldsymbol{I}, \boldsymbol{\Sigma}^{k-1}) (U^{k-1})^T$ where $\boldsymbol{A}(i,i) = \begin{cases} \boldsymbol{\Sigma}^{k-1}(i,i) & , \boldsymbol{\Sigma}^{k-1}(i,i) > \epsilon \\ \epsilon & , \boldsymbol{\Sigma}^{k-1}(i,i) \leq \epsilon \end{cases}$

In Fig. 2 we propose a back-propagation stage based on the SGD algorithm that can accelerate the convergence speed. As the forward propagation in the geometric layers can find new feature spaces through the close-form optimization, the back-propagation for these layers can be omitted. In the experiments, we do not use backpropagation for metric layers.

4 Experiments

In this section, we evaluate how our proposed method works in a simple classification problem. The proposed approach is evaluated on several small benchmark datasets described in Table 1 with some statistics about each.

Table 1. Description about chosen datasets

Name	# of input	# of Classes	Dimension
Breast-cancer	569	2	30
Wine	178	3	13
Iris	150	3	4
Vehicle	846	4	18
Vowel	990	11	14
German	1000	2	24

We have incorporated our proposed metric learning architecture into a Multi Layers Perceptron (MLP) to learn a metric and then classify data in the new learned space. We have first set up a network containing two metric learning layers (GMML + ReEng layer) followed by a simple MLP. The employed MLP is a network with two hidden layers including $20, 10$ nodes respectively. The gradient of the Cross-entropy loss function is used for error back-propagation. In all experiments, we have used 10 fold cross validation for model selection. It means that the original dataset is partitioned into 10 disjoint subsets where 9 subsets have been used for training and the remaining one for testing.

The data have been initially normalized. Table 2 shows the experimental results on six benchmark datasets. We have compared our model with three promising metric learning algorithms including GMML, LMNN, ITML. The results show that our model performs better than others in all datasets. Results for the other methods are based on the experiments reported by [26].

Table 2. Comparison with the state-of-the-art metric learning methods

Name	GMML	ITML	LMNN	OURS
Wine	0.96	0.92	0.94	**0.96**
Iris	0.97	0.974	0.95	**0.98**
Breast cancer	0.96	0.92	0.91	**0.99**
Vehicle	0.78	0.70	0.77	**0.81**
Vowel	0.57	0.56	0.53	**0.6**
German	0.72	0.705	0.71	**0.78**

We also have explored how the number of constraints would affect the accuracy of the classification. We first picked 30% of data randomly and make similar and dissimilar sets over the selected data. We gradually increase the number of constraints and investigate the effect of increased constraints on the classification. We first pick 30% of data to generate similar and dissimilar pairs over them and explore how our proposed model work on this configuration. Figure 3 shows how an increase in the number of constraints affects the performance of the classifier.

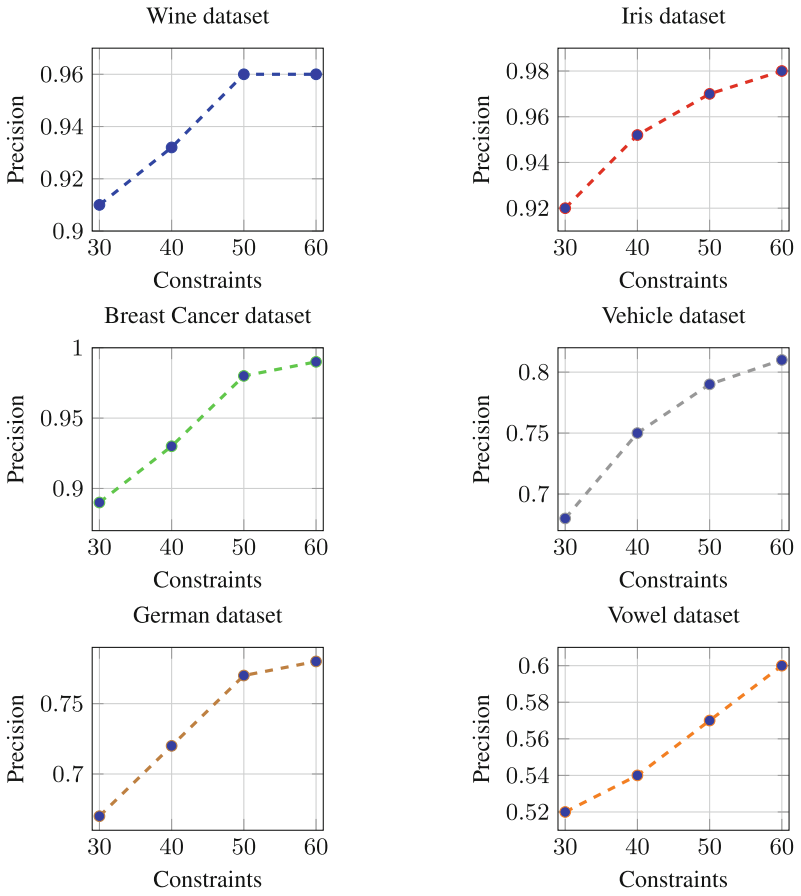

Fig. 3. Horizontal axis represents the percentage of data which are used for constraints and the vertical axis represents the precision

5 Conclusion

In this paper, we have proposed a new neural network architecture based on metric learning approaches which are updated on the SPD manifold. We focused on

classification tasks and it is considered as an initial attempt to explore the use of geometric learning on neural networks. Our proposed method has been implemented and evaluated on several benchmark datasets which showed a significant improvement in comparison with the state-of-the-art metric learning algorithms. We also explored how the number of constraints affects the performance of the classifier. As a future work, we aim to integrate this architecture into more complicated deep networks. Also, we plan to incorporate more metric learning algorithms into the proposed model.

References

1. Absil, P.-A., Mahony, R., Sepulchre, R.: Optimization Algorithms on Matrix Manifolds. Princeton University Press, Princeton (2009)
2. Altiotancay, H.: Improving the k-nearest neighbour rule: using geometrical neighbourhoods and manifold-based metrics. Expert Syst. **28**(4), 391–406 (2011)
3. Arsigny, V., Fillard, P., Pennec, X., Ayache, N.: Geometric means in a novel vector space structure on symmetric positive-definite matrices. SIAM J. Matrix Anal. Appl. **29**(1), 328–347 (2007)
4. Bhatia, R.: Positive Definite Matrices, vol. 24. Princeton University Press, Princeton (2009)
5. Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press, Cambridge (2004)
6. Bromley, J., Guyon, I., LeCun, Y., Säckinger, E., Shah, R.: Signature verification using a "siamese" time delay neural network. In: Advances in Neural Information Processing Systems, pp. 737–744 (1994)
7. Chopra, S., Hadsell, R., LeCun, Y.: Learning a similarity metric discriminatively, with application to face verification. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition CVPR 2005, vol. 1, pp. 539–546. IEEE (2005)
8. Choromanska, A., Agarwal, A., Langford, J.: Extreme multi class classification. In: NIPS Workshop: eXtreme Classification, vol. 1, pp. 1–2 (2013)
9. Coskun, H., Tan, D.J., Conjeti, S., Navab, N., Tombari, F.: Human motion analysis with deep metric learning. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) Computer Vision – ECCV 2018. LNCS, vol. 11218, pp. 693–710. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01264-9_41
10. Duan, Y., Zheng, W., Lin, X., Lu, J., Zhou, J.: Deep adversarial metric learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2780–2789 (2018)
11. Faraki, M., Harandi, M.T., Porikli, F.: Large-scale metric learning: a voyage from shallow to deep. IEEE Trans. Neural Netw. Learn. Syst. **29**(9), 4339–4346 (2018)
12. Hajiabadi, H., Monsefi, R., Yazdi, H.S.: Relf: robust regression extended with ensemble loss function. Appl. Intell. **49**, 1–14 (2018)
13. Hershey, J.R., Chen, Z., Roux, J.L., Watanabe, S.: Deep clustering: discriminative embeddings for segmentation and separation. In: 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 31–35. IEEE (2016)
14. Hu, J., Lu, J., Tan, Y.-P.: Discriminative deep metric learning for face verification in the wild. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1875–1882 (2014)

15. Huang, Z., Van Gool, L.J.: A riemannian network for SPD matrix learning. In: AAAI, vol. 1, p. 3 (2017)
16. Li, W., Huo, J., Shi, Y., Gao, Y., Wang, L., Luo, J.: Online deep metric learning. arXiv preprint arXiv:1805.05510 (2018)
17. Lin, W.-C., Lu, Y.-H., Tsai, C.-F.: Feature selection in single and ensemble learning-based bankruptcy prediction models. Expert Syst. **36**(1), e12335 (2019)
18. Oh Song, H., Xiang, Y., Jegelka, S., Savarese, S.: Deep metric learning via lifted structured feature embedding. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4004–4012 (2016)
19. Schroff, F., Kalenichenko, D., Philbin, J.: Facenet: a unified embedding for face recognition and clustering. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 815–823 (2015)
20. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
21. Sohn, K.: Improved deep metric learning with multi-class n-pair loss objective. In: Advances in Neural Information Processing Systems, pp. 1857–1865 (2016)
22. Oh Song, H., Jegelka, S., Rathod, V., Murphy, K.: Deep metric learning via facility location. In: Computer Vision and Pattern Recognition (CVPR), vol. 8 (2017)
23. Su, Y., King, I., Lyu, M.: Learning to rank using localized geometric mean metrics. arXiv preprint arXiv:1705.07563 (2017)
24. Sun, Y., Chen, Y., Wang, X., Tang, X.: Deep learning face representation by joint identification-verification. In: Advances in Neural Information Processing Systems, pp. 1988–1996 (2014)
25. Yen, I.E.-H., Huang, X., Ravikumar, P., Zhong, K., Dhillon, I.: Pd-sparse: a primal and dual sparse approach to extreme multiclass and multilabel classification. In: International Conference on Machine Learning, pp. 3069–3077 (2016)
26. Zadeh, P., Hosseini, R., Sra, S.: Geometric mean metric learning. In: International Conference on Machine Learning, pp. 2464–2471 (2016)
27. Zhu, P., Cheng, H., Hu, Q., Wang, Q., Zhang, C.: Towards generalized and efficient metric learning on riemannian manifold. In: IJCAI, pp. 3235–3241 (2018)