Adversarial Metric Learning

Shuo Chen[1], **Chen Gong**[1, 2], **Jian Yang**[1, 2, 3, *], **Xiang Li**[1, 3], **Yang Wei**[1, 3], **Jun Li**[2, 3]
[1]School of Computer Science & Engineering, Nanjing University of Science & Technology
[2]Key Laboratory of Intelligent Perception & Systems for High-Dimensional Information
[3]Jiangsu Key Laboratory of Image & Video Understanding for Social Security
{shuochen, chen.gong, csjyang, xiang.li.implus, csywei}@njust.edu.cn, junl.mldl@gmail.com

Abstract

In the past decades, intensive efforts have been put to design various loss functions and metric forms for metric learning problem. These improvements have shown promising results when the test data is similar to the training data. However, the trained models often fail to produce reliable distances on the ambiguous test pairs due to the different samplings between training set and test set. To address this problem, the Adversarial Metric Learning (AML) is proposed in this paper, which automatically generates adversarial pairs to remedy the sampling bias and facilitate robust metric learning. Specifically, AML consists of two adversarial stages, *i.e.* confusion and distinguishment. In confusion stage, the ambiguous but critical adversarial data pairs are adaptively generated to mislead the learned metric. In distinguishment stage, a metric is exhaustively learned to try its best to distinguish both adversarial pairs and original training pairs. Thanks to the challenges posed by the confusion stage in such competing process, the AML model is able to grasp plentiful difficult knowledge that has not been contained by the original training pairs, so the discriminability of AML can be significantly improved. The entire model is formulated into optimization framework, of which the global convergence is theoretically proved. The experimental results on toy data and practical datasets clearly demonstrate the superiority of AML to representative state-of-the-art metric learning models.

1 Introduction

The calculation of similarity or distance between a pair of data points plays a fundamental role in many machine learning and pattern recognition tasks such as retrieval [Zhan *et al.*, 2009], verification [Noroozi *et al.*, 2017], and classification [Yang *et al.*, 2016]. Therefore, "Metric Learning" [Bishop, 2006] was proposed to enable an algorithm to wisely acquire the appropriate distance metric so that the precise similarity between different examples can be faithfully reflected.

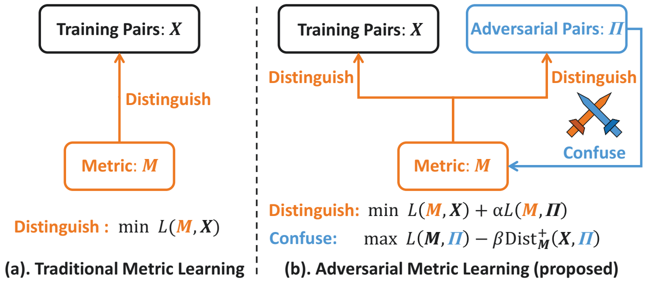
Training Pairs: X
Distinguish
Metric: M
Distinguish : min $L(M, X)$
(a). Traditional Metric Learning
Training Pairs: X
Adversarial Pairs: Π
Distinguish
Distinguish
Confuse
Metric: M
Distinguish: min $L(M, X) + \alpha L(M, \Pi)$
Confuse: max $L(M, \Pi) - \beta \mathrm{Dist}_M^+(X, \Pi)$
(b). Adversarial Metric Learning (proposed)

Figure 1: The comparison of traditional metric learning and our proposed model. (a) Traditional metric learning directly minimizes the loss $L(\boldsymbol{M}, \boldsymbol{X})$ to distinguish the training pairs. (b) Our proposed AML contains a distinguishment stage and a confusion stage, and the original training pairs and the adversarial pairs are jointly invoked to obtain an accurate $\boldsymbol{M}$. The confusion stage learns the adversarial pairs $\boldsymbol{\Pi}$ in the local region of $\boldsymbol{X}$, by maximizing $L(\boldsymbol{M}, \boldsymbol{\Pi})$ as well as the distance regularizer $-\mathrm{Dist}_{\boldsymbol{M}}^{+}(\boldsymbol{X}, \boldsymbol{\Pi})$.

In metric learning, the similarity between two example vectors $\boldsymbol{x}$ and $\boldsymbol{x}'$ is usually expressed by the distance function $\mathrm{Dist}(\boldsymbol{x}, \boldsymbol{x}')$. Perhaps the most commonly used distance function is Mahalanobis distance, which has the form $\mathrm{Dist}_{\boldsymbol{M}}(\boldsymbol{x}, \boldsymbol{x}') = (\boldsymbol{x} - \boldsymbol{x}')^{\top} \boldsymbol{M} (\boldsymbol{x} - \boldsymbol{x}')$[1]. Here the symmetric positive definite (SPD) matrix $\boldsymbol{M}$ should be learned by an algorithm to fit the similarity reflected by training data. By decomposing $\boldsymbol{M}$ as $\boldsymbol{M} = \boldsymbol{P}^{\top}\boldsymbol{P}$, we know that Mahalanobis distance intrinsically calculates the Euclidean distance in a projected linear space rendered by the projection matrix $\boldsymbol{P}$, namely $\mathrm{Dist}_{\boldsymbol{M}}(\boldsymbol{x}, \boldsymbol{x}') = ||\boldsymbol{P}(\boldsymbol{x} - \boldsymbol{x}')||_2^2$. Consequently, a large amount of models were proposed to either directly pursue the Mahalanobis matrix [Davis *et al.*, 2007; Zadeh *et al.*, 2016; Zhang and Zhang, 2017] or indirectly learn such a linear projection matrix $\boldsymbol{P}$ [Lu *et al.*, 2014; Harandi *et al.*, 2017]. Furthermore, considering that above linear transformation is not flexible enough to characterize the complex data relationship, some recent works utilized the deep neural networks, *e.g.* Convolutional Neural Network (CNN) [Oh Song *et al.*, 2016], to achieve the purpose of non-

[*] Corresponding author

[1] For simplicity, the notation of "square" on $\mathrm{Dist}_{\boldsymbol{M}}(\boldsymbol{x}, \boldsymbol{x}')$ has been omitted and it will not influence the final output.

linearity. Generally, the linear projection $\boldsymbol{P}\boldsymbol{x}$ is replaced by $\boldsymbol{P}\mathcal{W}(\boldsymbol{x})$, where $\mathcal{W}(\cdot)$ is a non-linear mapping.

However, above existing approaches simply learn the linear or non-linear metrics via designing different loss functions on the original training pairs. During test phase, due to the different samplings of training set and test set, some ambiguous data pairs that are difficult to be distinguished by the learned metric may appear, which will significantly impair the algorithm performance. To this end, we propose the Adversarial Metric Learning (AML) to learn a robust metric, which follows the idea of adversarial training [Li *et al.*, 2017], and is able to generate ambiguous but critical data pairs to enhance the algorithm robustness. Notice that the adversarial examples [Goodfellow *et al.*, 2015] are also generated by algorithms, yet they are used for classifiers. In Fig. 1, compared with the traditional metric learning methods that only distinguish the given training pairs, our AML learns the metric to distinguish both original training pairs and the generated adversarial pairs. Here, the adversarial data pairs are automatically synthesized by the algorithm to confuse the learned metric as much as possible. The adversarial pairs $\boldsymbol{\Pi}$ and the learned metric $\boldsymbol{M}$ form the adversarial relationship and each of them tries to "beat" the other one. Specifically, adversarial pairs tend to introduce the ambiguous examples which are difficult for the learned metric to correctly decide their (dis)similarities (*i.e.* confusion stage), while the metric makes its effort to discriminate the confusing adversarial pairs (*i.e.* distinguishment stage). In this sense, the adversarial pairs are helpful for our model to acquire the accurate metric. To ensure the efficiency, we convert the adversarial game to an optimization problem which has the optimal solution from the theoretical aspects. In the experiments, we show that the Mahalanobis metric learned by AML is superior to the state-of-the-art models on popular datasets with different tasks.

The most prominent advantage of our proposed AML is that the extra data pairs (*i.e.* adversarial pairs) are explored to boost the discriminability of the learned metric. In fact, several metric learning models have been proposed based on data augmentations [Ahmed *et al.*, 2015; Zagoruyko and Komodakis, 2015], or pair perturbations [Huang *et al.*, 2010]. However, the virtual data generated by these methods are largely based on the domain knowledge or heuristic rules which may significantly differ from the practical test data, so their performances are rather limited. In contrast, the additional adversarial pairs in AML are consciously designed to mislead the learning metric, so they are formed in an intentional and realistic way. To narrow the searching space of adversarial pairs, AML establishes the adversarial pairs within neighborhoods of original training pairs as shown in Fig. 2. Thanks to the learning on both real and generated pairs, the discriminability of our method can be substantially improved. The main contributions of this paper are summarized as:

• We propose a novel framework dubbed AML, which is able to generate adversarial data pairs in addition to the original given training data to enhance the model discriminability.

• AML is converted to an optimization framework, of which the convergence is analyzed.

• AML is empirically validated to outperform state-of-the-art metric learning models on typical applications.

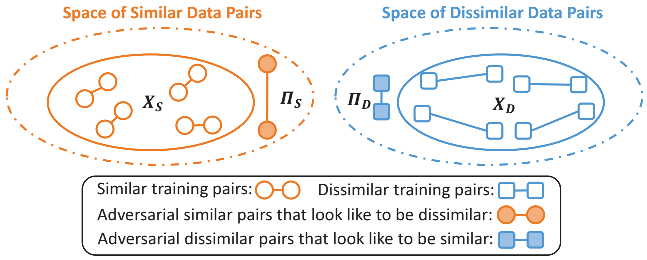
Space of Similar Data Pairs
Space of Dissimilar Data Pairs
X_S
Π_S
Π_D
X_D
Similar training pairs: ○–○
Dissimilar training pairs: □–□
Adversarial similar pairs that look like to be dissimilar: ●–●
Adversarial dissimilar pairs that look like to be similar: ■–■

Figure 2: Generations of adversarial similar pairs and adversarial dissimilar pairs. Similar and dissimilar pairs are marked with orange balls and blue blocks, respectively. Hollow balls and blocks denote the original training examples, while filled balls and blocks denote the adversarial examples which are automatically generated by our model. Note that the generated two points constituting the adversarial similar pairs (*i.e.* $\boldsymbol{\Pi}_S$) are far from each other, which describe the extreme cases for two examples to be similar pairs. Similarly, the generated two points constituting the adversarial dissimilar pairs (*i.e.* $\boldsymbol{\Pi}_D$) are closely distributed, which depict the worst cases for two examples to be dissimilar pairs.

2 Adversarial Metric Learning

We first introduce some necessary notations in Section 2.1, and then explain the optimization model of the proposed AML in Section 2.2. Finally, we provide the iterative solution as well as the convergence proof in Section 2.3 and Section 2.4, respectively.

2.1 Preliminaries

Let $\boldsymbol{X} = [\boldsymbol{X}_1, \boldsymbol{X}_2, \cdots, \boldsymbol{X}_N] \in \mathbb{R}^{2d\times N}$ be the matrix of N training example pairs, where $\boldsymbol{X}_i = [\boldsymbol{x}_i^\top, \boldsymbol{x}_i'^\top]^\top \in \mathbb{R}^{2d}$ $(i = 1, 2, \cdots, N)$ consists of a pair of d-dimensional training examples. Besides, we define a label vector $\boldsymbol{y} \in \{-1, 1\}^N$ of which the i-th element y_i represents the relationship of the pairwise examples recorded by the i-th column of $\boldsymbol{X}$, namely $y_i = 1$ if $\boldsymbol{x}_i$ and $\boldsymbol{x}_i'$ are similar, and $y_i = -1$ otherwise. Based on the supervised training data, the Mahalanobis metric $\boldsymbol{M} \in \mathbb{R}^{d\times d}$ is learned by minimizing the general loss function $L(\boldsymbol{M}, \boldsymbol{X}, \boldsymbol{y})$, namely

$$\min_{\boldsymbol{M}\in\mathcal{S}} d(\boldsymbol{M}) = L(\boldsymbol{M}, \boldsymbol{X}, \boldsymbol{y}), \tag{1}$$

in which $\mathcal{S}$ denotes the feasible set for $\boldsymbol{M}$, such as SPD constraint [Luo and Huang, 2018], bounded constraint [Yang *et al.*, 2016], low-rank constraint [Harandi *et al.*, 2017], *etc.* In our proposed AML, N' generated adversarial pairs are denoted by the matrix $\boldsymbol{\Pi} = [\boldsymbol{\Pi}_1, \boldsymbol{\Pi}_2, \cdots, \boldsymbol{\Pi}_{N'}] \in \mathbb{R}^{2d\times N'}$, where $\boldsymbol{\Pi}_i = [\boldsymbol{\pi}_i^\top, \boldsymbol{\pi}_i'^\top]^\top \in \mathbb{R}^{2d}$ $(i = 1, 2, \cdots, N')$ represents the i-th generated example pair. By setting N' to N in this work, the augmented distance $\mathrm{Dist}_{\boldsymbol{M}}^{+}(\boldsymbol{X}, \boldsymbol{\Pi})$ between $\boldsymbol{X}$ and $\boldsymbol{\Pi}$ is thus defined as the sum of the Mahalanobis distances between all the pairwise examples of $\boldsymbol{X}$ and $\boldsymbol{\Pi}$, *i.e.* $\mathrm{Dist}_{\boldsymbol{M}}^{+}(\boldsymbol{X}, \boldsymbol{\Pi}) = \sum_{i=1}^{N} \mathrm{Dist}_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{\pi}_i) + \mathrm{Dist}_{\boldsymbol{M}}(\boldsymbol{x}_i', \boldsymbol{\pi}_i')$.

2.2 Model Establishment

As mentioned in the Introduction, our AML algorithm alternates between learning the reliable distance metric (*i.e.* distinguishment stage) and generating the misleading adversarial data pairs (*i.e.* confusion stage), in which the latter is the core of AML to boost the learning performance. The main target of confusion stage is to produce the adversarial pairs $\boldsymbol{\Pi}$ to confuse the learned metric $\boldsymbol{M}$. That is to say, we should explore the pair $\boldsymbol{\Pi}_i$ of which the similarity predicted by $\boldsymbol{M}$ is opposite to its true label. Fig. 2 intuitively plots the generations of $\boldsymbol{\Pi}$. To achieve this effect, we search the data pairs $\boldsymbol{\Pi}_i$ in the neighborhood of $\boldsymbol{X}_i$ to violate the results predicted by the learned metric. Specifically, the loss function $\boldsymbol{L}(\boldsymbol{M}, \boldsymbol{\Pi}, \boldsymbol{y})$ is expected to be as large as possible, while the augmented distance $\mathrm{Dist}^{+}_{\boldsymbol{M}}(\boldsymbol{X}, \boldsymbol{\Pi})$ is preferred to be a small value in the following optimization objective

$$\max_{\boldsymbol{\Pi}} c(\boldsymbol{\Pi}) = L(\boldsymbol{M}, \boldsymbol{\Pi}, \boldsymbol{y}) - \beta \mathrm{Dist}^{+}_{\boldsymbol{M}}(\boldsymbol{X}, \boldsymbol{\Pi}), \qquad (2)$$

in which the regularizer coefficient $\beta \in \mathbb{R}^{+}$ is manually tuned to control the size of searching space. Since $\boldsymbol{\Pi}_i$ is found in the neighborhood of $\boldsymbol{X}_i$, the true label of $\boldsymbol{\Pi}_i$ is reasonably assumed as y_i, *i.e.* the label of $\boldsymbol{X}_i$. It means that Eq. (2) tries to find data pairs $\boldsymbol{\Pi}_1, \boldsymbol{\Pi}_2, \cdots, \boldsymbol{\Pi}_N$, of which their metric results are opposite to their corresponding true labels $y_1, y_2, \cdots, y_N$. Therefore, such an optimization exploits the adversarial pairs $\boldsymbol{\Pi}$ to confuse the metric $\boldsymbol{M}$.

Nevertheless, Eq. (2) cannot be directly taken as a valid optimization problem, as it is not bounded which means that Eq. (2) might not have the optimal solution. To avoid this problem and achieve the same effect with Eq. (2), we convert the maximization *w.r.t.* the true labels $\boldsymbol{y}$ to the minimization *w.r.t.* the opposite labels $-\boldsymbol{y}$. As the misleading adversarial pair $\boldsymbol{\Pi}_S$ in Fig. 2 is the worst case to be the similar pair with label 1 (*i.e.* maximizing Eq. (2)), we may in turn find the dissimilar pair with label -1 that incurs very small loss from the viewpoint of negative pairs (the same thing happens regarding $\boldsymbol{\Pi}_D$), *i.e.* minimizing the following form of *confusion*

$$\min_{\boldsymbol{\Pi}} C_{\boldsymbol{M}}(\boldsymbol{\Pi}) = L(\boldsymbol{M}, \boldsymbol{\Pi}, -\boldsymbol{y}) + \beta \mathrm{Dist}^{+}_{\boldsymbol{M}}(\boldsymbol{X}, \boldsymbol{\Pi}). \qquad (3)$$

The optimal solution to the above problem always exists, because any loss function $L(\cdot)$ and distance operator $\mathrm{Dist}^{+}_{\boldsymbol{M}}(\cdot)$ have the minimal values.

By solving Eq. (3), we obtain the generated adversarial pairs recorded in the matrix $\boldsymbol{\Pi}$, which can be employed to learn a proper metric. Since these confusing adversarial pairs are incorrectly predicted, the metric $\boldsymbol{M}$ should exhaustively distinguish them to improve the discriminability. By combining the adversarial pairs in $\boldsymbol{\Pi}$ and the originally available training pairs in $\boldsymbol{X}$, the augmented training loss utilized in the *distinguishment* stage has a form of

$$\min_{\boldsymbol{M} \in \mathcal{S}} D_{\boldsymbol{\Pi}}(\boldsymbol{M}) = L(\boldsymbol{M}, \boldsymbol{X}, \boldsymbol{y}) + \alpha L(\boldsymbol{M}, \boldsymbol{\Pi}, \boldsymbol{y}), \qquad (4)$$

where the regularizer coefficient $\alpha \in \mathbb{R}^{+}$ is manually tuned to control the weights of the adversarial data. Furthermore, to improve both the distinguishment (*i.e.* Eq. (4)) and the confusion (*i.e.* Eq. (3)) during their formed adversarial game, they have to be optimized alternatively, *i.e.*

$$\begin{cases} \boldsymbol{M}^{(t+1)} & = \arg\min D_{\boldsymbol{\Pi}^{(t)}}(\boldsymbol{M}), \ (\textit{Distinguishment}), \\ \boldsymbol{\Pi}^{(t+1)} & = \arg\min C_{\boldsymbol{M}^{(t+1)}}(\boldsymbol{\Pi}), \qquad (\textit{Confusion}). \end{cases} \qquad (5)$$

The straightforward implementation of Eq. (5) yet confronts two problems in the practical use. Firstly, Eq. (5) is iteratively performed, which greatly decreases the efficiency of the model. Secondly, the iterations with two different functions are not necessarily convergent [Ben-Tal *et al.*, 2009].

To achieve the similar effect of the direct alternation in Eq. (5) while avoiding the two disadvantages mentioned above, the iterative expression for $\boldsymbol{\Pi}$ is integrated to the optimization of $\boldsymbol{M}$. Therefore, our AML is ultimately expressed as a bi-level optimization problem [Bard, 2013], namely

$$\min_{\boldsymbol{M} \in \mathcal{S}, \boldsymbol{\Pi}^*} D_{\boldsymbol{\Pi}^*}(\boldsymbol{M}) \qquad \text{s.t. } \boldsymbol{\Pi}^* = \arg\min_{\boldsymbol{\Pi}} C_{\boldsymbol{M}}(\boldsymbol{\Pi}), \quad (6)$$

in which $\boldsymbol{\Pi}^*$ denotes the optimal adversarial pairs matrix, and $C_{\boldsymbol{M}}(\boldsymbol{\Pi})$ is required to be strictly quasi-convex[2]. Note that the strictly quasi-convex property ensures the uniqueness of $\boldsymbol{\Pi}^*$, and helps to make the problem well-defined.

2.3 Optimization

To implement Eq. (6), we instantiate the loss $L(\cdot)$ in $D(\cdot)$ and $C(\cdot)$ to obtain a specific learning model. To make $C_{\boldsymbol{M}}(\boldsymbol{\Pi})$ to be convex, here we employ the geometric-mean loss [Zadeh *et al.*, 2016] which has an unconstrained form of

$$L_{\mathrm{g}}(\boldsymbol{M}, \boldsymbol{X}, \boldsymbol{y}) = \sum\nolimits_{y_i=1} \mathrm{Dist}_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}'_i) + \sum\nolimits_{y_i=-1} \mathrm{Dist}'_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}'_i), \quad (7)$$

where the loss of dissimilar data pairs is expressed as $\mathrm{Dist}'_{\boldsymbol{M}}(\boldsymbol{x}_i, \boldsymbol{x}'_i) = \mathrm{Dist}_{\boldsymbol{M}^{-1}}(\boldsymbol{x}_i, \boldsymbol{x}'_i)$ to increase the distances between dissimilar examples. Moreover, we substitute the loss $L(\cdot)$ in Eq. (6) with $L_{\mathrm{g}}(\cdot)$, and impose the SPD constraint on $\boldsymbol{M}$ for simplicity, namely $\mathcal{S} = \{\boldsymbol{M} | \boldsymbol{M} \succ 0, \boldsymbol{M} \in \mathbb{R}^{d \times d}\}$. Then the detailed optimization algorithm for Eq. (6) is provided as follows.

Solving $\boldsymbol{\Pi}$: We can directly obtain the closed-form solution (*i.e.* the optimal adversarial pairs $\boldsymbol{\Pi}^*$) for optimizing $\boldsymbol{\Pi}$. Specifically, by using the convexity of $C_{\boldsymbol{M}}(\boldsymbol{\Pi})$, we let $\nabla C_{\boldsymbol{M}}(\boldsymbol{\Pi}) = 0$, and arrive at

$$\begin{cases} \frac{1}{2} \nabla_{\boldsymbol{\pi}_i} C = \boldsymbol{M}^{-y_i} (\boldsymbol{\pi}_i - \boldsymbol{\pi}'_i) + \beta \boldsymbol{M} (\boldsymbol{\pi}_i - \boldsymbol{x}_i) = 0, \\ \frac{1}{2} \nabla_{\boldsymbol{\pi}'_i} C = \boldsymbol{M}^{-y_i} (\boldsymbol{\pi}'_i - \boldsymbol{\pi}_i) + \beta \boldsymbol{M} (\boldsymbol{\pi}'_i - \boldsymbol{x}'_i) = 0, \end{cases} \qquad (8)$$

which holds for any $i = 1, 2, \cdots, N$. It is clear that the equation system Eq. (8) has the unique solution

$$\begin{cases} \boldsymbol{\pi}^*_i = (2\boldsymbol{M}^{-y_i} + \beta \boldsymbol{M})^{-1} (\boldsymbol{M}^{-y_i} \overline{\boldsymbol{x}}_i + \beta \boldsymbol{M} \boldsymbol{x}_i), \\ \boldsymbol{\pi}'^*_i = (2\boldsymbol{M}^{-y_i} + \beta \boldsymbol{M})^{-1} (\boldsymbol{M}^{-y_i} \overline{\boldsymbol{x}}_i + \beta \boldsymbol{M} \boldsymbol{x}'_i), \end{cases} \qquad (9)$$

where $\overline{\boldsymbol{x}}_i = \boldsymbol{x}_i + \boldsymbol{x}'_i$. Such a closed-form solution $\boldsymbol{\Pi}^*$ means that the minimizer of $C_{\boldsymbol{M}}(\boldsymbol{\Pi})$ can be directly expressed as

$$\boldsymbol{\Pi}^* = \mathcal{F}(\boldsymbol{M}), \qquad (10)$$

where $\mathcal{F}(\cdot)$ is a mapping from $\mathbb{R}^{d \times d}$ to $\mathbb{R}^{2d \times N}$ decided by Eq. (9). Hence Eq. (6) is equivalently converted to

$$\min_{\boldsymbol{M} \succ 0} D(\boldsymbol{M}) = L_{\mathrm{g}}(\boldsymbol{M}, \boldsymbol{X}, \boldsymbol{y}) + \alpha L_{\mathrm{g}}(\boldsymbol{M}, \mathcal{F}(\boldsymbol{M}), \boldsymbol{y}), \quad (11)$$

which is a normal optimization problem regarding the single variable $\boldsymbol{M}$.

Solving $\boldsymbol{M}$: The key point is to calculate the gradient of the second term $L_{\mathrm{g}}(\boldsymbol{M}, \mathcal{F}(\boldsymbol{M}), \boldsymbol{y})$. We substitute $\mathcal{F}(\boldsymbol{M})$ with Eq. (9) and obtain that

$$L_{\mathrm{g}}(\boldsymbol{M}, \mathcal{F}(\boldsymbol{M}), \boldsymbol{y}) = \beta^2 \sum\nolimits_{i=1}^{N} \widehat{\boldsymbol{x}}_i^{\top} \boldsymbol{U} \left(\frac{\boldsymbol{\Lambda}^{3+2y_i}}{(2\mathbf{I} + \beta^2 \boldsymbol{\Lambda}^{1+y_i})^2} \right) \boldsymbol{U}^{\top} \widehat{\boldsymbol{x}}_i, \qquad (12)$$

[2] If a function $h(\cdot)$ satisfies $h(\lambda \boldsymbol{x}_1 + (1-\lambda)\boldsymbol{x}_2) < \max(h(\boldsymbol{x}_1), h(\boldsymbol{x}_2))$ for all $\boldsymbol{x}_1 \neq \boldsymbol{x}_2$ and $\lambda \in (0, 1)$, then $h(\cdot)$ is strictly quasi-convex.

Algorithm 1 Solving AML in Eq. (6) via gradient descent.

```
Input: Training data pairs encoded in $\boldsymbol{X}$; labels $\boldsymbol{y}$; parameters $\alpha$, $\beta$, $\rho$.
Initialize: $t = 1$; $\boldsymbol{M}^{(t)} = \mathbf{I}$.
Repeat:
    1. Compute the gradient $\nabla D(\boldsymbol{M}^{(t)})$ by Eq. (15);
    2. Update $\boldsymbol{M}^{(t+1)} = \mathcal{P}_{\mathcal{S}}(\boldsymbol{M}^{(t)} - \rho \nabla D(\boldsymbol{M}^{(t)}))$;
    3. Update $t = t + 1$;
Until Convergence.
Output: The converged $\boldsymbol{M}$.
```

where $\widehat{\boldsymbol{x}}_i = \boldsymbol{x}_i - \boldsymbol{x}'_i$, and $\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\top$ is the eigen-decomposition of $\boldsymbol{M}$. Each term to be summed in the above Eq. (12) can be compactly written as $H_i(\boldsymbol{M}) = \widehat{\boldsymbol{x}}_i^\top \boldsymbol{U}\boldsymbol{h}_i(\boldsymbol{\Lambda})\boldsymbol{U}^\top \widehat{\boldsymbol{x}}_i$, where $\boldsymbol{h}_i(\boldsymbol{\Lambda}) = \operatorname{diag}(h_i(\lambda_1), h_i(\lambda_2), \cdots, h_i(\lambda_d))$ and h_i is a differentiable function. The gradient of such a term can be obtained from the properties of eigenvalues and eigenvectors [Bellman, 1997], namely $\partial\lambda_i = \boldsymbol{U}_i^\top \partial \boldsymbol{M}\boldsymbol{U}_i$, and $\partial \boldsymbol{U}_i = (\lambda_i \mathbf{I} - \boldsymbol{M})^\dagger \partial \boldsymbol{M}\boldsymbol{U}_i$. By further leveraging the chain rule of function derivatives [Petersen *et al.*, 2008], the gradient of Eq. (12) can be expressed as

$$\nabla_{\boldsymbol{M}} L_{\mathrm{g}}(\boldsymbol{M}, \mathcal{F}(\boldsymbol{M}), \boldsymbol{y}) = \beta^2 \sum\nolimits_{i=1}^{N} \nabla H_i, \tag{13}$$

in which

$$\nabla H_i = \sum\nolimits_{j=1}^{d} h'_i(\lambda_j)\left(\widehat{\boldsymbol{x}}_i^\top \boldsymbol{U}_j \boldsymbol{U}_j^\top \widehat{\boldsymbol{x}}_i\right)\boldsymbol{U}_j\boldsymbol{U}_j^\top + \sum\nolimits_{j=1}^{d} 2h_i(\lambda_j)(\lambda_j\mathbf{I} - \boldsymbol{M})^\dagger \widehat{\boldsymbol{x}}_i \widehat{\boldsymbol{x}}_i^\top \boldsymbol{U}_j \boldsymbol{U}_j^\top. \tag{14}$$

Finally, the gradient of $D(\boldsymbol{M})$ in Eq. (11) equals to

$$\nabla D(\boldsymbol{M}) = \boldsymbol{A} - \boldsymbol{M}^{-1}\boldsymbol{B}\boldsymbol{M}^{-1} + \alpha\beta^2 \sum\nolimits_{i=1}^{N} \nabla H_i, \tag{15}$$

in which the matrices $\boldsymbol{A} = \sum_{y_i=1}(\boldsymbol{x}_i - \boldsymbol{x}'_i)(\boldsymbol{x}_i - \boldsymbol{x}'_i)^\top$ and $\boldsymbol{B} = \sum_{y_i=-1}(\boldsymbol{x}_i - \boldsymbol{x}'_i)(\boldsymbol{x}_i - \boldsymbol{x}'_i)^\top$. It should be noticed that the pseudo-inverse $(\lambda_i\mathbf{I} - \boldsymbol{M})^\dagger$ can be calculated efficiently for the SPD matrix $\boldsymbol{M}$, which only depends on the eigen-decomposition.

Now we can simply employ the gradient-based method for SPD optimization to solve our proposed model. By following the popular SPD algorithm in the existing metric learning models [Ye *et al.*, 2017; Luo and Huang, 2018], the projection operator is utilized to remain the SPD effect. Specifically, for a symmetric matrix $\boldsymbol{M} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\top$, the projection $\mathcal{P}_{\mathcal{S}}(\cdot)$ from $\mathbb{R}^{d\times d}$ to $\mathbb{R}^{d\times d}$ truncates negative eigenvalues of $\boldsymbol{\Lambda}$, *i.e.*

$$\mathcal{P}_{\mathcal{S}}(\boldsymbol{M}) = \boldsymbol{U}\max(\boldsymbol{\Lambda}, 0)\boldsymbol{U}^\top. \tag{16}$$

It can be proved that the metric $\boldsymbol{M}$ remains symmetry after the gradient descent, so the projection operator is leveraged in the gradient descent to find the optimal solution. The pseudo-code for solving Eq. (6) is provided in Algorithm 1, where the step size ρ is recommended to be fixed to 0.001 in our experiments.

2.4 Convergence Analysis

Since our proposed bi-level optimization problem greatly differs from the traditional metric learning models, here we provide the theoretical analysis for the algorithm convergence.

Firstly, to ensure the definition of AML in Eq. (6) is valid, we prove that the optimal solution $\boldsymbol{\Pi}^*$ always exists uniquely by showing the strict (quasi-)convexity of $C_{\boldsymbol{M}}(\boldsymbol{\Pi})$ when $L(\boldsymbol{M}, \boldsymbol{\Pi}, -\boldsymbol{y})$ is instantiated by $L_{\mathrm{g}}(\boldsymbol{M}, \boldsymbol{\Pi}, -\boldsymbol{y})$, namely

Theorem 1. *Assume that $L(\boldsymbol{M}, \boldsymbol{\Pi}, -\boldsymbol{y}) = L_g(\boldsymbol{M}, \boldsymbol{\Pi}, -\boldsymbol{y})$. Then $C_{\boldsymbol{M}}(\boldsymbol{\Pi})$ is strictly convex.*

Proof. Assume that $\boldsymbol{\Omega} = [\boldsymbol{\Omega}_1, \boldsymbol{\Omega}_2, \cdots, \boldsymbol{\Omega}_N] \in \mathbb{R}^{2d\times N}$, $\boldsymbol{\Omega}_i = [\boldsymbol{\omega}_i^\top, \boldsymbol{\omega}_i'^\top]^\top \in \mathbb{R}^{2d}$ $(i = 1, 2, \cdots, N)$ and $\mu \in (0, 1)$. By invoking the SPD property of both $\boldsymbol{M}$ and $\boldsymbol{M}^{-1}$, we have

$$\begin{aligned} & L\left(\boldsymbol{M}, \mu\boldsymbol{\Pi} + (1-\mu)\boldsymbol{\Omega}, -\boldsymbol{y}\right) \\ &= \sum\nolimits_{i=1}^{N} \left(\mu\widehat{\boldsymbol{\pi}}_i + (1-\mu)\widehat{\boldsymbol{\omega}}_i\right)^\top \boldsymbol{M}^{-y_i} \left(\mu\widehat{\boldsymbol{\pi}}_i + (1-\mu)\widehat{\boldsymbol{\omega}}_i\right) \\ &< \sum\nolimits_{i=1}^{N} \left(\mu\widehat{\boldsymbol{\pi}}_i{}^\top \boldsymbol{M}^{-y_i}\widehat{\boldsymbol{\pi}}_i + (1-\mu)\widehat{\boldsymbol{\omega}}_i{}^\top \boldsymbol{M}^{-y_i}\widehat{\boldsymbol{\omega}}_i\right) \\ &= \mu L(\boldsymbol{M}, \boldsymbol{\Pi}, -\boldsymbol{y}) + (1-\mu)L(\boldsymbol{M}, \boldsymbol{\Omega}, -\boldsymbol{y}), \end{aligned} \tag{17}$$

where $\widehat{\boldsymbol{\pi}}_i = \boldsymbol{\pi}_i - \boldsymbol{\pi}'_i$, $\widehat{\boldsymbol{\omega}}_i = \boldsymbol{\omega}_i - \boldsymbol{\omega}'_i$, and $\boldsymbol{M}^{-y_i}$ denotes $\boldsymbol{M}$ $(y_i = -1)$ or $\boldsymbol{M}^{-1}$ $(y_i = 1)$. Hence $L(\boldsymbol{M}, \boldsymbol{\Pi}, \boldsymbol{1} - \boldsymbol{y})$ satisfies the definition of strictly convex function. Similarly, it is easy to check the convexity of $\mathrm{Dist}^+_{\boldsymbol{M}}(\boldsymbol{X}, \boldsymbol{\Pi})$ *w.r.t.* $\boldsymbol{\Pi}$, which completes the proof. □

Furthermore, as we employ the projection $\mathcal{P}_{\mathcal{S}}(\cdot)$ in gradient descent, it is necessary to demonstrate that any result $\boldsymbol{M} - \rho\nabla D(\boldsymbol{M})$ has the orthogonal eigen-decomposition. Otherwise, $\mathcal{P}_{\mathcal{S}}(\cdot)$ cannot be executed and the SPD property of $\boldsymbol{M}$ is not guaranteed. Therefore, in the following Theorem 2, we prove that the gradient matrix $\nabla D(\boldsymbol{M})$ is symmetric, and thus any converged iteration points are always included in the feasible set $\mathcal{S}$.

Theorem 2. *For any differentiable functions $h_i(\lambda)$ $(i = 1, 2, \cdots, N)$, the matrix $\nabla D(\boldsymbol{M})$ in Eq. (15) is symmetric.*

Proof. Assume that the SPD matrix $\boldsymbol{M} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\top \in \mathbb{R}^{d\times d}$, where $\boldsymbol{\Lambda} = \operatorname{diag}(\lambda_1, \lambda_2, \cdots, \lambda_d) \in \mathbb{R}^{d\times d}$ and $\boldsymbol{U} = [\boldsymbol{U}_1, \boldsymbol{U}_2, \cdots, \boldsymbol{U}_d] \in \mathbb{R}^{d\times d}$ are the matrices consisting of distinct eigenvalues and unit eigenvectors of $\boldsymbol{M}$, respectively. For each term in the summation of Eq. 13, we simply consider one $H(\boldsymbol{M}) = \widehat{\boldsymbol{x}}^\top \boldsymbol{U}\boldsymbol{h}(\boldsymbol{\Lambda})\boldsymbol{U}^\top \widehat{\boldsymbol{x}}$ without loss of generality. By using the Maclaurin's formula [Bellman, 1997] on the all eigenvalues of $\boldsymbol{M}$, namely $\boldsymbol{h}(\boldsymbol{\Lambda}) = \sum_{i=0}^{+\infty} \frac{h^{(i)}(0)}{i!}\boldsymbol{\Lambda}^i$, then $H(\boldsymbol{M})$ equals to

$$\widehat{\boldsymbol{x}}^\top \boldsymbol{U} \sum\nolimits_{i=0}^{+\infty} \frac{h^{(i)}(0)}{i!} \boldsymbol{\Lambda}^i \boldsymbol{U}^\top \widehat{\boldsymbol{x}} = \sum\nolimits_{i=0}^{+\infty} \frac{h^{(i)}(0)}{i!} \widehat{\boldsymbol{x}}^\top \boldsymbol{M}^i \widehat{\boldsymbol{x}}. \tag{18}$$

Since the gradient of $\widehat{\boldsymbol{x}}^\top \boldsymbol{M}^i \widehat{\boldsymbol{x}}$ is symmetric for any $i \in \mathbb{N}$ [Petersen *et al.*, 2008], the summation of the gradient matrices is also symmetric and the proof is completed. □

Now it has been proved that $\boldsymbol{M} - \rho\nabla D(\boldsymbol{M})$ remains symmetric during iterations, and thus the projection $\mathcal{P}_{\mathcal{S}}(\cdot)$ ensures the SPD property of $\boldsymbol{M}$, *i.e.,* the constraint $\boldsymbol{M} \in \mathcal{S}$ is always satisfied. It means that the gradient descent is always performed in the feasible region of the optimization problem. Then according to the theoretically proved convergence of the gradient descent method [Boyd and Vandenberghe, 2004], Algorithm 1 converges to the stationary point of Eq. (11).

3 Experiments

In this section, empirical investigations are conducted to validate the effectiveness of AML. In detail, we first visualize the mechanism of the proposed AML on a synthetic dataset. Then we compare the performance of the proposed method AML (Algorithm 1) with three classical metric learning methods (ITML [Davis *et al.*, 2007], LMNN [Weinberger and Saul, 2009] and FlatGeo [Meyer *et al.*, 2011]) and five state-of-the-art metric learning methods (RVML [Perrot and Habrard, 2015], GMML [Zadeh *et al.*, 2016], ERML [Yang *et al.*, 2016], DRML [Harandi *et al.*, 2017], and DRIFT [Ye *et al.*, 2017]) on seven benchmark classification datasets. Next, all methods are compared on three practical datasets related to face verification and image matching. Finally, the parametric sensitivity of AML is studied.

3.1 Experiments on Synthetic Dataset

We first demonstrate the effectiveness of AML on a synthetic dataset which contains 200 training examples and 200 test examples across two classes. The data points are sampled from a 10-dimensional normal distribution, and are visualized by the first two principal components [Abdi and Williams, 2010]. As shown in Figs. 3(a) and (b), the training set is clean, but the test examples belonging to two classes overlap in the intersection region and lead to many ambiguous test data pairs.

Since GMML [Zadeh *et al.*, 2016] shares the same loss function with our AML, and the only difference between GMML and AML is that AML utilizes the adversarial points while GMML does not, so here we only compare the results of GMML and AML to highlight the usefulness of our adversarial framework. The training and test results of both methods are projected to Euclidean space by using the learned metrics. It can be found that the traditional metric learning model GMML simply learns the optimal metric for training data, and thus its corresponding projection matrix directly maps the data points onto the horizontal-axis in the training set (Fig. 3(c)). However, such a learned metric is confused by the data points in the test set (Fig. 3(d)) as the two classes are very close to each other in the test set. As a result, the two classes are not well-separated by the learned metric. In contrast, the proposed AML not only produces very impressive result on the training set (Fig. 3(e)), but also generates very discriminative results on test set (Fig. 3(f)). The test data belonging to the same class is successfully grouped together while the examples of different classes are separated apart. This good performance of AML owes to the adversarial data pairs as shown by "×" in Fig. 3(a). Such difficult yet critical training pairs effectively cover the ambiguous situations that may appear in the test set, and therefore enhancing the generalizability and discriminability of our AML.

3.2 Experiments on Classification

To evaluate the performances of various compared methods on classification task, we follow existing works [Xie and Xing, 2013; Zhan *et al.*, 2016] and adopt the k-NN classifier ($k = 5$) based on the learned metrics to investigate the classification error rate. The datasets are from the well-known UCI repository [Asuncion and Newman, 2007], which include *Breast-Cancer*, *Vehicle*, *German-Credit*, *Image-Segment*, *Isolet*, *Letters* and *MNIST*. The number of contained classes, examples and features are displayed in Table 1. We compare all methods over 20 random trials. In each trial, 80% of examples are randomly selected as the training examples, and the rest are used for testing. By following the recommendation in [Zadeh *et al.*, 2016], the training pairs are generated by randomly picking up $1000c(c-1)$ pairs among the training examples. The parameters in our method such as α and β are tuned by searching the grid $\{10^{-3}, 10^{-2}, \cdots, 10^{3}\}$. The parameters for baseline algorithms are also carefully tuned to achieve the optimal results. The average classification error rates of compared methods are showed in Table 1, and we find that AML obtains the best results when compared with other methods in most cases. We also perform the t-test (significance level 0.05) to investigate the superiority of our method to the best baseline method on above datasets.

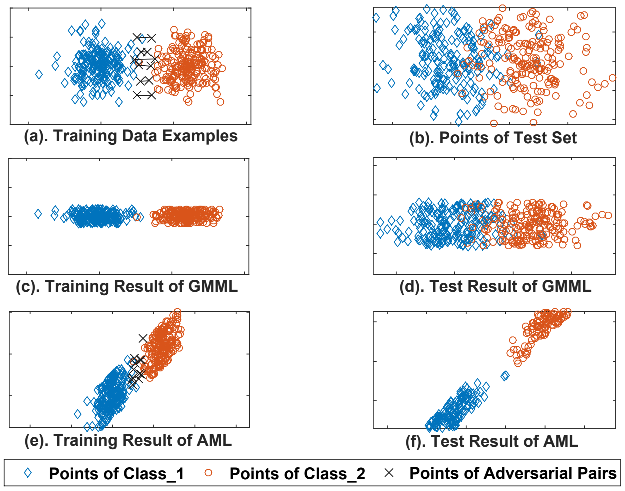

Figure 3: Visual comparison of GMML and the proposed AML on synthetic dataset. Although the satisfactory training result is obtained by the traditional metric learning model GMML, it cannot well handle the test cases with ambiguous pairs. In contrast, our proposed AML shows good discriminability on both training and test sets. The reason lies in that the generated adversarial training data pairs help to boost the discriminability of AML.

3.3 Experiments on Verification

We also use two face datasets and one image matching dataset to evaluate the capabilities of all compared methods on image verification task. The *PubFig* face dataset [Nair and Hinton, 2010] consists of of 2×10^4 pairs of images belonging to 140 people, in which the first 80% pairs are selected for training and the rest are used for testing. Similar experiments are performed on the *LFW* face dataset [Huang *et al.*, 2007] which includes 13233 unconstrained face images of 5749 individuals. The image matching dataset *MVS* [Brown *et al.*, 2011] consists of 64×64 gray-scale image sampled from 3D reconstructions of the Statue of Liberty (LY), Notre Dame (ND) and Half Dome in Yosemite (YO). By following the settings in [Simo-Serra *et al.*, 2015], LY and ND are put together to form a training set with over 10^5 image patch pairs, and 10^4 patch pairs in YO are used for testing. The adopted features

Methods	**Breast-Cancer** 9, 2, 699	**Vehicle** 18, 4, 848	**German-Credit** 24, 2, 1000	**Image-Segment** 19, 7, 2310	**Isolet** 617, 26, 7797	**Letters** 16, 20, 20000	**MNIST** 784, 10, 4000	**Reference**
ITML	$.073 \pm .010$	$.301 \pm .051$	$.292 \pm .032$	$.051 \pm .022$	$.092 \pm .011$	$.062 \pm .002$	$.143 \pm .023$	ICML 2007
LMNN	$.052 \pm .012$	$.234 \pm .021$	$.274 \pm .013$	$.027 \pm .008$	$.032 \pm .012$	$.042 \pm .020$	$.174 \pm .020$	JMLR 2009
FlatGeo	$.075 \pm .021$	$.283 \pm .042$	$.322 \pm .031$	$.042 \pm .012$	$.065 \pm .001$	$.082 \pm .016$	$.146 \pm .043$	JMLR 2011
RVML	$.045 \pm .005$	$.223 \pm .032$	$.280 \pm .020$	$.035 \pm .006$	$.035 \pm .015$	$.056 \pm .010$	$.132 \pm .008$	NIPS 2015
GMML	$.040 \pm .015$	$.235 \pm .035$	$.273 \pm .021$	$.031 \pm .005$	$.075 \pm .012$	$.051 \pm .001$	$.120 \pm .005$	ICML 2016
ERML	$.045 \pm .002$	$.245 \pm .040$	$.279 \pm .012$	$.036 \pm .003$	$.065 \pm .001$	$.054 \pm .010$	$.133 \pm .016$	IJCAI 2016
DRML	$.049 \pm .011$	$.246 \pm .053$	$.279 \pm .034$	$.026 \pm .007$	$.039 \pm .014$	$.041 \pm .027$	$.118 \pm .012$	ICML 2017
DRIFT	$.043 \pm .005$	$.240 \pm .023$	$.278 \pm .026$	$.032 \pm .007$	$.034 \pm .033$	$.049 \pm .012$	$.127 \pm .034$	IJCAI 2017
AML	$.044 \pm .001$	$.203 \pm .021$	$.251 \pm .012$	$.024 \pm .004$	$.029 \pm .009$	$.032 \pm .020$	$.105 \pm .001$	Ours
t-test	–	✓	✓	✓	✓	✓	✓	

Table 1: Classification and clustering error rates of different methods. Three numbers below each datasets correspond to the feature dimensionality (d), number of classes (c) and number of examples (n). The best two results in each dataset are highlighted in red and blue, respectively. "✓" indicates that AML is significantly better than the best baseline method.

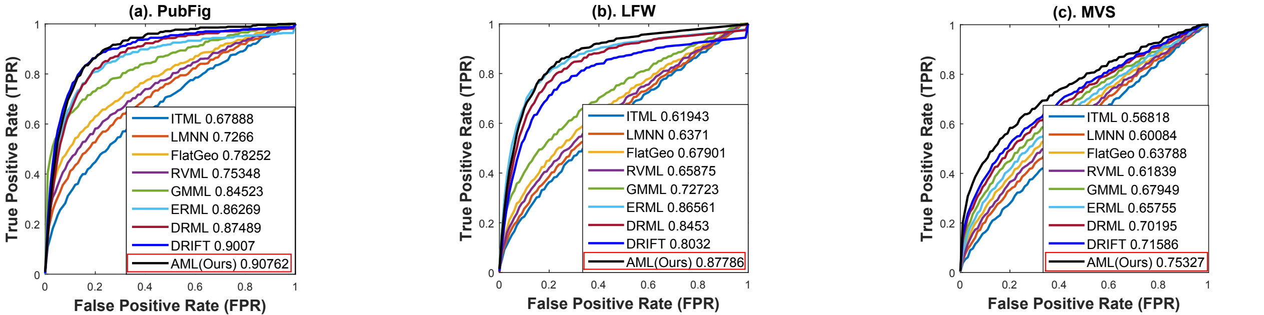

Figure 4: ROC curves of different methods on (a) PubFig, (b) LFW and (c) MVS datasets. AUC values are presented in the legends.

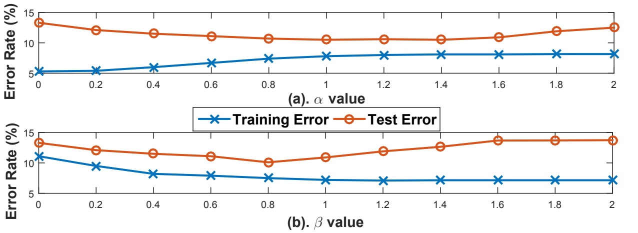

Figure 5: Parametric Sensitivity on MNIST dataset. (a) Error rates under different α values ($\beta = 1$); (b) Error rates under different β values ($\alpha = 1$).

for above experiments are extracted by DSIFT [Cheung and Hamarneh, 2009] and Siamese-CNN [Zagoruyko and Komodakis, 2015] for face datasets (*i.e.* PubFig and LFW) and image patch dataset (*i.e.* MVC), respectively. We plot the Receiving Operator Characteristic (ROC) curve by changing the thresholds of different distance metrics. Then the values of Area Under Curve (AUC) are calculated to evaluate the performances quantitatively. From the ROC curves and AUC values in Fig. 4, it is clear to see that AML consistently outperforms other methods.

3.4 Parametric Sensitivity

In our proposed AML, there are two parameters which might influence the model performance. Parameter α in Eq. (4) determines the weights between original training data and generated adversarial data, and parameter β in Eq. (3) controls the size of neighborhood producting adversarial data.

Intuitively, the growing of α increases the importance of adversarial data, and the decrease of α makes the model put more emphasize on the original training data. As shown in Fig. 5(a), here we change the value of α and record the training error and test error on the MNIST dataset that has been used in Section 3.2. An interesting finding is that, the training error grows when α increases in the range $(0, 1)$, but the test error consistently decreases at this time. This is because tuning up α helps to alleviate the over-fitting problem, and thus the different samplings between the test data and the training data can be better dealt with. We also find that the training error and test error make a compromise when α is around 1, and thus 1 is an ideal choice for the parameter α. Similarly, the parameter β varies within $(0, 2)$ and the corresponding training error and test error are recorded in Fig. 5(b). It is clear to find that $\beta \approx 0.8$ renders the highest test accuracy and the performance is generally stable around 0.8, which mean that this parameter can be easily tuned for practical use.

4 Conclusion

In this paper, we propose a metric learning framework, named Adversarial Metric Learning (AML), which contains two important competing stages including confusion and distinguishment. The confusion stage adaptively generates adversarial data pairs to enhance the capability of learned metric to deal with the ambiguous test data pairs. To the best of our knowledge, this is the first work to introduce the adversarial framework to metric learning, and the visualization results demonstrate that the generated adversarial data critically enriches the knowledge for model training and thus making the learning algorithm acquire the more reliable and precise metric than the state-of-the-art methods. Furthermore, we show that such adversarial process can be compactly unified into a bi-level optimization problem, which is theoretically proved to have a globally convergent solver. Since the proposed AML framework is general in nature, it is very promising to apply AML to more deep neural networks based metric learning models for the future work.

Acknowledgments

The authors would like to thank the anonymous reviewers for their critical and constructive comments and suggestions. This work was supported by the National Science Fund of China under Grant Nos. U1713208 and 61472187, the 973 Program No.2014CB349303, NSF of China (No: 61602246), NSF of Jiangsu Province (No: BK20171430), the Fundamental Research Funds for the Central Universities (No: 30918011319), the "Summit of the Six Top Talents" Program (No: DZXX-027) and Program for Changjiang Scholars.

References

[Abdi and Williams, 2010] Hervé Abdi and Lynne J Williams. Principal component analysis. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(4):433–459, 2010.

[Ahmed *et al.*, 2015] Ejaz Ahmed, Michael Jones, and Tim K Marks. An improved deep learning architecture for person re-identification. In *CVPR*, pages 733–740, 2015.

[Asuncion and Newman, 2007] Arthur Asuncion and David Newman. Uci machine learning repository, 2007.

[Bard, 2013] Jonathan F Bard. *Practical Bilevel Optimization: Algorithms and Applications*, volume 30. Springer Science & Business Media, 2013.

[Bellman, 1997] Richard Bellman. *Introduction to Matrix Analysis*. SIAM, 1997.

[Ben-Tal *et al.*, 2009] Aharon Ben-Tal, Laurent El Ghaoui, and Arkadi Nemirovski. *Robust Optimization*. Princeton University Press, 2009.

[Bishop, 2006] Christopher M Bishop. *Pattern Recognition and Machine Learning*. springer, 2006.

[Boyd and Vandenberghe, 2004] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge university, 2004.

[Brown *et al.*, 2011] Matthew Brown, Gang Hua, and Simon Winder. Discriminative learning of local image descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):43–57, 2011.

[Cheung and Hamarneh, 2009] Warren Cheung and Ghassan Hamarneh. N-sift: n-dimensional scale invariant feature transform. *IEEE Transactions on Image Processing*, 18(9), 2009.

[Davis *et al.*, 2007] Jason V Davis, Brian Kulis, Prateek Jain, Suvrit Sra, and Inderjit S Dhillon. Information-theoretic metric learning. In *ICML*, pages 209–216, 2007.

[Goodfellow *et al.*, 2015] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *ICLR*, pages 212–227, 2015.

[Harandi *et al.*, 2017] Mehrtash Harandi, Mathieu Salzmann, and Richard Hartley. Joint dimensionality reduction and metric learning: A geometric take. In *ICML*, pages 1943–1950, 2017.

[Huang *et al.*, 2007] Gary B Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical report, Technical Report 07-49, University of Massachusetts, Amherst, 2007.

[Huang *et al.*, 2010] Kaizhu Huang, Rong Jin, Zenglin Xu, and Cheng-Lin Liu. Robust metric learning by smooth optimization. *UAI*, 2010.

[Li *et al.*, 2017] Zheng Li, Yu Zhang, and Ying Wei. End-to-end adversarial memory network for cross-domain sentiment classification. In *IJCAI*, pages 2133–2140, 2017.

[Lu *et al.*, 2014] Jiwen Lu, Xiuzhuang Zhou, Yap-Pen Tan, Yuanyuan Shang, and Jie Zhou. Neighborhood repulsed metric learning for kinship verification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(2):331–345, 2014.

[Luo and Huang, 2018] Lei Luo and Heng Huang. Matrix variate gaussian mixture distribution steered robust metric learning. In *AAAI*, pages 933–940, 2018.

[Meyer *et al.*, 2011] Gilles Meyer, Silvère Bonnabel, and Rodolphe Sepulchre. Regression on fixed-rank positive semidefinite matrices: a riemannian approach. *Journal of Machine Learning Research*, 12(Feb):593–625, 2011.

[Nair and Hinton, 2010] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, pages 807–814, 2010.

[Noroozi *et al.*, 2017] Vahid Noroozi, Lei Zheng, Sara Bahaadini, Sihong Xie, and Philip S Yu. Seven: Deep semi-supervised verification networks. In *IJCAI*, pages 133–140, 2017.

[Oh Song *et al.*, 2016] Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding. In *CVPR*, pages 4004–4012, 2016.

[Perrot and Habrard, 2015] Michael Perrot and Amaury Habrard. Regressive virtual metric learning. In *NIPS*, 2015.

[Petersen *et al.*, 2008] Kaare Brandt Petersen, Michael Syskind Pedersen, et al. The matrix cookbook. *Technical University of Denmark*, 7:15, 2008.

[Simo-Serra *et al.*, 2015] Edgar Simo-Serra, Eduard Trulls, Luis Ferraz, Iasonas Kokkinos, Pascal Fua, and Francesc Moreno-Noguer. Discriminative learning of deep convolutional feature point descriptors. In *ICCV*, pages 118–126, 2015.

[Weinberger and Saul, 2009] Kilian Q Weinberger and Lawrence K Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10(Feb):207–244, 2009.

[Xie and Xing, 2013] Pengtao Xie and Eric P Xing. Multi-modal distance metric learning. In *IJCAI*, pages 1806–1812, 2013.

[Yang *et al.*, 2016] Xun Yang, Meng Wang, Luming Zhang, and Dacheng Tao. Empirical risk minimization for metric learning using privileged information. In *IJCAI*, pages 2266–2272, 2016.

[Ye *et al.*, 2017] Han-Jia Ye, De-Chuan Zhan, Xue-Min Si, and Yuan Jiang. Learning mahalanobis distance metric: Considering instance disturbance helps. In *IJCAI*, pages 866–872, 2017.

[Zadeh *et al.*, 2016] Pourya Zadeh, Reshad Hosseini, and Suvrit Sra. Geometric mean metric learning. In *ICML*, 2016.

[Zagoruyko and Komodakis, 2015] Sergey Zagoruyko and Nikos Komodakis. Learning to compare image patches via convolutional neural networks. In *ICCV*, pages 4353–4361, 2015.

[Zhan *et al.*, 2009] De-Chuan Zhan, Ming Li, Yu-Feng Li, and Zhi-Hua Zhou. Learning instance specific distances using metric propagation. In *ICML*, pages 1407–1413, 2009.

[Zhan *et al.*, 2016] De-Chuan Zhan, Peng Hu, Zui Chu, and Zhi-Hua Zhou. Learning expected hitting time distance. In *AAAI*, pages 2309–2314, 2016.

[Zhang and Zhang, 2017] Jie Zhang and Lijun Zhang. Efficient stochastic optimization for low-rank distance metric learning. In *AAAI*, pages 933–940, 2017.