International Journal of Machine Learning and Cybernetics (2021) 12:3597–3605
https://doi.org/10.1007/s13042-021-01408-3

ORIGINAL ARTICLE

Metric learning with clustering-based constraints

Xinyao Guo[1] · Chuangyin Dang[1] · Jianqing Liang[1] · Wei Wei[1] · Jiye Liang[1]

Received: 18 March 2021 / Accepted: 11 August 2021 / Published online: 25 August 2021

Abstract
In most of the existing metric learning methods, the relation is fixed throughout the metric learning process. However, the fixed relation may be harmful to learn a good metric. The adversarial metric learning implements a dynamic update of the pairwise constraints. Inspired by the idea of dynamically updating constraints, we propose in this paper a metric learning model with clustering-based constraints (ML-CC), wherein the triple constraints of large margin are iteratively generated with the clusters of data points. The proposed method can overcome the shortage of the fixed triple constraints constructed under the Euclidian distance. The experimental results on synthetic and real datasets indicate that the performance of the ML-CC is superior to that of the existing state-of-the-art metric learning methods.

Keywords Metric learning · Triple constraints · Clustering · Large margin · Dynamic constraint

1 Introduction

In many machine learning tasks [1–3], describing whether two samples are similar is a core problem. Some standard metrics are for describing the similarity between two samples, which include Euclidean distance, Cosine distance [4], Hamming distance [5] and Wasserstein distance [6]. However, these distances only consider specific relationships between the data and are difficult to widely apply to different learning tasks. Therefore, it is necessary to adaptively mine the corresponding similarity metric based on the characteristics of the data.

✉ Chuangyin Dang
mecdang@cityu.edu.hk

Xinyao Guo
1303590343@qq.com

Jianqing Liang
liangjq@sxu.edu.cn

Wei Wei
weiwei@sxu.edu.cn

Jiye Liang
ljy@sxu.edu.cn

[1] Key Laboratory of Computational Intelligence and Chinese Information Processing of Ministry of Education, School of Computer and Information Technology, Shanxi University, Taiyuan 030006, Shanxi, China

Mahalanobis distance has been widely used in metric learning owing to its useful properties and excellent generalization ability. Among the metric learning methods with Mahalanobis distance, the methods based on pairwise constraints play an important role in metric learning [7–13]. The DML-eig [14] proposes that the constraints should only include the minimal cannot-link pairs and all must-link pairs, where the optimization model can be converted into a simple eigenvalue optimization problem. However, the model is extremely sensitive to noise and outliers. The RVML [15] presents an efficient metric learning model, which moves the points toward predefined virtual points and reduces point pairs from quadratic to linear. Nevertheless, the quality of the virtual points has a significant impact on the performance of the learned metric. In [16], it is believed that the distance between the must-link pairs should be less than a threshold, and the distance between the cannot-link pairs should be greater than the threshold. The pairwise constraints attempt to make the distance between must-link pairs (cannot-link pairs) smaller than (larger than) a threshold, but some datasets have large differences in similarity and some have small differences in dissimilarity.

To solve the problem with pairwise constraints, triple constraints were proposed in the literature, whereas the sample's neighborhood relations remain fixed throughout the learning process [17, 18]. The LMNN [17] uses triples violating the current metric criterion and reduces the amount of computation using the difference between two adjacent

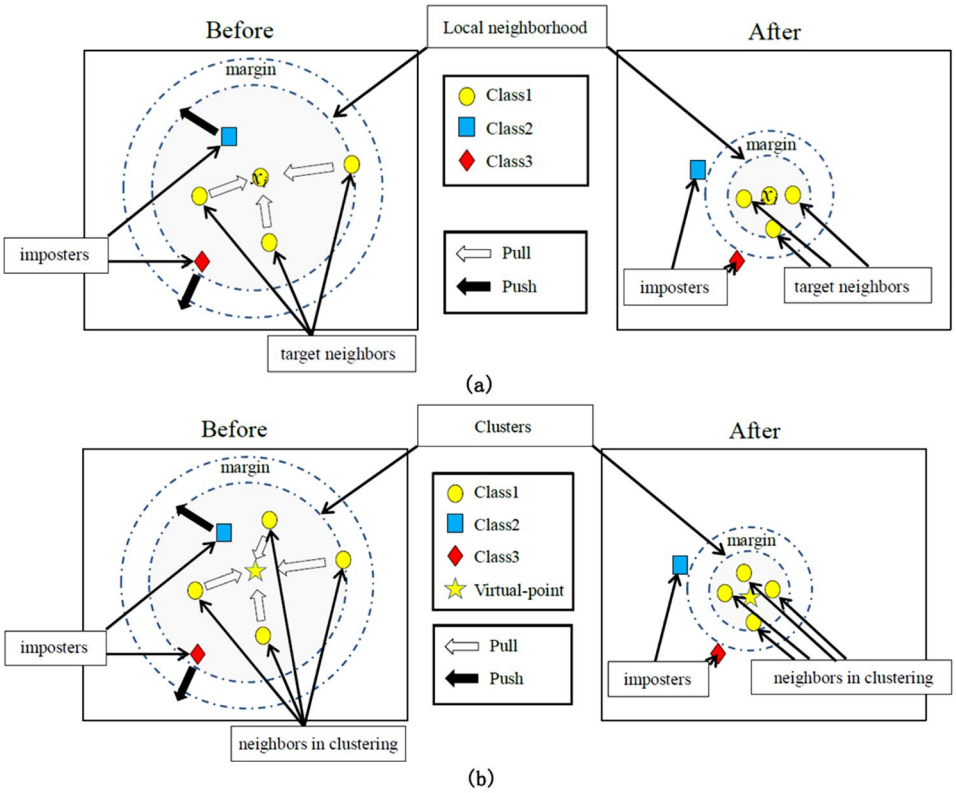

Fig. 1 Schematic illustration of one point's neighborhood before training (left) versus after training (right). In Fig. 1a, the neighborhood of one point consists of three nearest neighbors. In Fig. 1b, the cluster consists of the points that correspond with their cluster centers that overlap each other.

gradients. As an improvement of the LMNN, the PFLMNN [18] only considers the nearest dissimilar sample for the target neighborhood of the sample so that the number of samples used in the optimization process is considerably smaller. In [19, 20], several appropriate neighborhood relations are experimentally verified to significantly improve the quality of the learned metric. In addition, the ITML [21] proposes that if a metric can function well, the optimization model can find the metric closest to the metric while also satisfying the given randomly selected constraints. In [22], the triple constraints are selected by defining the importance of constraints. It can be seen from the above analysis that mining constraints are significant to distance metric learning.

In most of the methods, most constraints are given before metric learning and the constraints cannot be adaptively updated. The adversarial metric learning (AML) uses the idea of adversarial to generate "difficult" constraints (adversarial pairs) based on the original constraints, where the difficulty is overcome as much as possible in the metric learning phase. In the ASTCML [23], the adversarial sample is learned near the original sample to construct adversarial triple constraints. Although AML and ASTCML can dynamically learn constraints, the number of effective adversarial constraints is very small. Inspired by the anchor-based clustering methods [24, 25] and adaptive nearest-neighbor graph learning methods [26–28], we will adaptively construct constraints from the data. To mitigate the complexity of our model, we propose a metric learning model with clustering-based constraints (ML-CC), which iteratively updates triple constraints of large margin in the process of metric learning. The traditional metric learning combined with clustering methods tends to focus on how to learn metrics from the dataset to improve the performance of clustering [29–31]. Instead, our focus is on using clustering to dynamically mine constraints to improve the quality of the learned metric.

With a clustering method, the triple constraints are generated by regarding a cluster center as a target point and the cluster as a neighborhood of the target point (displayed in Fig. 1). The main contributions of this paper are summarized as follows.

- We incorporate the triple constraints of large margin into the clustering model to construct a metric learning model and implement mutual guidance of metric learning and constraint construction.
- We construct a target point neighborhood, which is the basis of constructing triple constraints in a metric learning model, via clustering data points instead of searching the neighbors of data points in terms of Euclidian distance.
- The ML-CC is empirically validated to outperform the state-of-the-art metric learning methods.

2 Related work

Currently, most of the metric learning methods acquire fixed constraints based on prior knowledge. The framework of the model can be simply and uniformly described as:

$$l(B, C(X)) + \alpha R(B), \tag{1}$$

where l is a loss function, B is the Mahalanobis metric parameter matrix, C is the function of constraints on dataset X, R is a regularization term of B, and α is a tradeoff parameter. $C(X)$ is obtained with prior knowledge and has no connection with B.

2.1 Convex clustering with metric learning

Let the data matrix $X = \{x_1, x_2, \dots, x_N\}$ with $x_i \in \mathbb{R}^d$ be a collection of N data points and matrix $U = \{u_1, u_2, \dots, u_N\}$ with $u_i \in \mathbb{R}^d$ a collection of cluster centers. In [24], convex clustering with a metric learning problem is formulated as the following optimization problem:

$$\begin{aligned} &\min_{U,B} \frac{1}{2}\sum_{j=1}^{N} d_B^2(x_j, u_j) + \gamma \sum_{1\le j_1<j_2\le N} w_{\{j_1,j_2\}} \|u_{j_1} - u_{j_2}\|_1 \\ &\text{s.t. } \log\det(B) \ge 0, \end{aligned} \tag{2}$$

where $d_B^2(x_j, u_j) = (x_j - u_j)^T B(x_j - u_j)$, the constraint $\log\det(B) \ge 0$ ensures that the metric matrix B has a full rank, γ is a positive tuning constant, and $w_{\{j_1,j_2\}}$ is a nonnegative weight, which can be used to control the scope of clustering centers fusion. The goal of convex clustering is to cluster the dataset X by merging the cluster centers in U. If the data are clustered into k clusters, there will be k unique rows of U. The problem (2) can be solved with the ADMM method. Due to the high cost of solving the cluster centers, it is difficult to work on large scale datasets.

2.2 Adversarial metric learning

Let $X = \{X_1, X_2, \dots, X_N\} \in \mathbb{R}^{2d\times N}$ be the matrix of N training constraints, where $X_i = [x_i^T, x_i'^T]^T \in \mathbb{R}^{2d}$ consists of a pair of d-dimensional training points. Let $V = \{V_1, V_2, \dots, V_N\} \in \mathbb{R}^{2d\times N}$, where $V_i = [v_i^T, v_i'^T]^T \in \mathbb{R}^{2d}$ represents the i-th generated "difficult" constraints, and let x_i have the similar label with v_i. This method divides the process of metric learning into two steps:

The first step is to adaptively generate the "difficult" constraints which mislead the learned metric. The goal is to learn a "difficult" constraints matrix V through the following optimization problem:

$$\begin{aligned} \min_V C(V) = &\sum_{(v_i,v_i')\in D} d_B^2(v_i, v_i') + \sum_{(v_i,v_i')\in S} d_{B^{-1}}^2(v_i, v_i') \\ &+ \beta(d_B^2(x_i, v_i) + d_B^2(x_i', v_i')). \end{aligned} \tag{3}$$

where $d_B^2(v_i, v_i') = (v_i - v_i')^T B(v_i - v_i')$, $d_{B^{-1}}^2(v_i, v_i') = (v_i - v_i')^T B^{-1}(v_i - v_i')$, $\beta \in \mathbb{R}^+$ is manually tuned to control the degree of proximity to the original constraint, D is a disimilar matrix and S is a similar matrix. In the problem, if $S_{ij} = 1$, it means that v_i and v_j have the similar label, otherwise $D_{ij} = 1$.

The second step is to try its best to distinguish both the "difficult" constraints and the original constraints. The goal is to learn a metric matrix B through the following optimization problem:

$$\begin{aligned} \min_B D(B) = &\sum_{(x_i,x_i')\in S} d_B^2(x_i, x_i') + \sum_{(x_i,x_i')\in D} d_{B^{-1}}^2(x_i, x_i') \\ &+ \alpha(\sum_{(v_i,v_i')\in S} d_B^2(v_i, v_i') + \sum_{(v_i,v_i')\in D} d_{B^{-1}}^2(v_i, v_i')). \end{aligned} \tag{4}$$

where the parameter $\alpha \in \mathbb{R}^+$ control the weights of the "difficult" constraints. Futhermore, two problems have to be optimized alternatively, i.e.

$$\begin{cases} B^{t+1} = arg\min D_{V^t}(B) \\ V^{t+1} = arg\min C_{B^{t+1}}(V) \end{cases}$$

3 Metric learning with clustering-based constraints

3.1 Preliminaries

Given the data matrix $X = \{x_1, x_2, \dots, x_N\}$, each point $x_i \in \mathbb{R}^d$ has a label $y_i \in \{1, 2, \dots, c\}$, where c is the number of classes. Let C_k denote the set of points in the k-th class and $|C_k|$ be the number of points in C_k. Let matrix $U = \{u_1, u_2, \dots, u_N\}$ with $u_i \in \mathbb{R}^d$ be a collection of cluster centers, where each cluster center u_i has a one-to-one corresponding point x_i. The cluster center u_i is assigned the same label as x_i, where point x_i is closer to the cluster center u_i than any other point (one real or cluster center).

3.2 Model establishment

We incorporate triple constraints into a clustering model to construct a novel model of metric learning. In the proposed model, the center of each cluster is regarded as a target point, and the cluster corresponding to a target point is regarded as the neighborhood of the target points. The metric learning problem is formulated as optimizing a regularized objective function:

$$\begin{aligned}\min_{U,B}\ & \mu\sum_{i=1} d_B^2(x_i,u_i)+\beta\sum_{k=1}^{c}\sum_{\substack{u_{j_1}\neq u_{j_2}\\ x_{j_1},x_{j_2}\in C_k}} w_{\{j_1,j_2\}}\\ & \|u_{j_1}-u_{j_2}\|_2^2+(1-\mu)\sum_{i,l,y_i\neq y_l}\xi_{il}\\ s.t.\ & \sum_{y_i\neq y_l} d_B^2(x_l,u_i)-d_B^2(x_i,u_i)\geq 1-\xi_{il},\xi_{il}\geq 0,\\ & B\succeq 0\end{aligned} \tag{5}$$

where β and μ are two positive tuning constants and $w_{\{j_1,j_2\}}$ is a non-negative weight. $w_{\{j_1,j_2\}}=1$ if data point x_{j_2} locates in K-nearest neighbors with the same label as data point x_{j_1}; $w_{\{j_1,j_2\}}=0$ otherwise. Note that $B\succeq 0$ means that B is a positive semi-definite matrix, and ξ_{il} is the slack variable.

The objective function consists of three terms: the first term indicates that this convex optimization problem aims to cluster the data points such that the distance between the data points and their corresponding cluster centers are minimized; the second term is a regularizer that leverages group sparsity to control the number of cluster centers of the clustering solution for each class; the third term is a relaxation variable to allow some constraints to be unsatisfied. Furthermore, a large margin constraint is employed to ensure that each cluster center is as close as possible to its corresponding data point and remains distant from the data points with different labels. We use an alternating procedure that alternates between cluster learning (minimizing over U) and metric learning (minimizing over B) to solve the optimization problem (5).

3.3 Solving U with fixed B

After fixing B, the optimization problem of formula (5) can be equivalently written as

$$\begin{aligned}\min_{U}\ & \mu\sum_{i=1}^{N} d_B^2(x_i,u_i)+\beta\sum_{k=1}^{c}\sum_{\substack{u_{j_1}\neq u_{j_2}\\ x_{j_1},x_{j_2}\in C_k}} w_{\{j_1,j_2\}}\|u_{j_1}\\ & -u_{j_2}\|_2^2+(1-\mu)\sum_{i,l,y_i\neq y_l}\xi_{il}\\ s.t.\ & \sum_{y_i\neq y_l} d_B^2(x_l,u_i)-d_B^2(x_i,u_i)\geq 1-\xi_{il},\xi_{il}\geq 0.\end{aligned} \tag{6}$$

One can rewrite the loss function in Eq. (6) as

$$\begin{aligned}L(U)=\ & \mu\sum_{i=1}^{N} d_B^2(x_i,u_i)+\beta\sum_{k=1}^{c}\sum_{\substack{u_{j_1}\neq u_{j_2}\\ x_{j_1},x_{j_2}\in C_k}}\\ & w_{\{j_1,j_2\}}\|u_{j_1}-u_{j_2}\|_2^2\\ & +(1-\mu)\sum_{i,l,y_i\neq y_l}[1+d_B^2(x_i,u_i)-d_B^2(x_l,u_i)]_+\end{aligned}$$

Let P_i be a set of integers such that $l\in P_i$ if and only if (i,l) triggers a hinge loss. A subgradient condition sufficient for an optimal U is for all $i\in\{1,\ldots,n\}$:

$$\begin{aligned}\frac{\partial L(U)}{\partial u_i}=\ & 2\mu B(u_i-x_i)+2\beta\sum_{(x_i,x_j)\in C_k,u_i\neq u_j} w_{ij}(u_i-u_j)\\ & +2(1-\mu)\sum_{l\in P_i} B(x_l-x_i)=0.\end{aligned} \tag{7}$$

Let $G=\{(u_i,u_j)|u_i\neq u_j,(x_i,x_j)\in C_k\}$. One can rewrite the optimality condition as

$$\begin{aligned}u_i=(\mu B+\beta\sum_{(u_i,u_j)\in G} w_{ij})^{-1}(\mu Bx_i+\beta\sum_{(u_i,u_j)\in G}\\ w_{ij}u_j-(1-\mu)\sum_{l\in P_i} B(x_l-x_i)).\end{aligned} \tag{8}$$

Because B is a positive semi-definite matrix for $(B+\beta\sum_{(u_i,u_j)\in G} w_{ij})^{-1}=L(\Lambda+\beta\sum_{(u_i,u_j)\in G} w_{ij})^{-1}L^T$, where $B=L\Lambda L^T$ is the eigenvalue decomposition of matrix B, we can quickly solve it at a small cost.

When the difference between two cluster centers clusters (C_1, C_2) is small, we are required to merge these into one cluster C. The new cluster center in C is:

$$u_C=\frac{|C_1|u_{C_1}+|C_2|u_{C_2}}{|C_1|+|C_2|}. \tag{9}$$

To judge whether two clusters are required to be merged, we use a small threshold on $\|u_{C_1}-u_{C_2}\|$, which we typically take to be a fraction of the smallest nonzero difference in all cluster centers $min_{i,j}\|u_{C_i}-u_{C_j}\|$, and then set the threshold: $1.02*min_{i,j}\|u_{C_i}-u_{C_j}\|$. We compute the distance between cluster centers, search for all pairs of cluster centers smaller than the current constraint using breadth-first search(BFS), and compute the fused cluster center by formula (9).

3.4 Solving B with fixed U

Fix U, the optimization problem (5) can be equivalently written as:

$$\begin{aligned}\min_{B}\ & \mu\sum_{i=1}^{N} d_B^2(x_i,u_i)+(1-\mu)\sum_{i,l,y_i\neq y_l}\xi_{il},\\ s.t.\ & \sum_{y_i\neq y_l} d_B^2(x_l,u_i)-d_B^2(x_i,u_i)\geq 1-\xi_{il}\\ & \xi_{il}\geq 0\\ & B\succeq 0.\end{aligned} \tag{10}$$

This model is similar to LMNN, we use an active-set decent algorithm to solve it. For details, please refer to the LMNN solution process. For simple notation, let $\tau_{il} = (u_i - x_l)(u_i - x_l)^T$. The gradient of Eq. (10) can be written as:

$$\bigtriangledown G(B) = \mu \sum_{i=1}^{N} \tau_{ii} + (1-\mu) \sum_{i=1}^{N} \sum_{l \in P_i} (\tau_{ii} - \tau_{il}), \tag{11}$$

where $l \in P_i$ if and only if the (i, l) triggers the hinge loss in the third part of Eq. (11). We update the gradient $\bigtriangledown G_{t+1}(B)$ at iteration $t+1$ from the gradient $\bigtriangledown G_t(B)$ at iteration t:

$$\begin{aligned} \bigtriangledown G_{t+1}(B) = \bigtriangledown G_t(B) - (1-\mu) \sum_{i=1}^{N} \sum_{l \in P_i^{(t)} - P_i^{(t+1)}} (\tau_{ii} - \tau_{il}) \\ + (1-\mu) \sum_{i=1}^{N} \sum_{l \in P_i^{(t+1)} - P_i^{(t)}} (\tau_{ii} - \tau_{il}), \end{aligned} \tag{12}$$

where $P_i^{(t)} - P_i^{(t+1)}$ is no longer an active constraint, and $P_i^{(t+1)} - P_i^{(t)}$ just becomes an active constraint. The flowchart of the algorithm is shown in Algorithm 1.

In Algorithm 1, the main computational cost is in solving the problem (10). The time complexity of solving the problem (10) is $O(t_1 n^2 d^2)$, where t_1 is the iteration number about solving the problem (10). Thus, the time complexity of Algorithm 1 is in general no more than $O(t_2(t_1 n^2 d^2 + nd^2 + n^2))$, where t_2 is the iteration number about Algorithm 1. For LMNN method, the time complexity of solving is generally no more than $O(t_1 k n^2 d^2)$. Therefore, our method has the same order of magnitude as LMNN's method.

Algorithm 1 Metric Learning with Clustering-Based Constraints

Input: X: datasets, Y: label vector, μ, β: regularization parameter, w: weight matrix
Output: **B**: Mahalanobis matrix, **U**: cluster centers
Step 1: Set U = X;
Step 2: While(the object function value is not decreasing)
Step 3: Update the cluster centers using formula(8);
Step 4: Fusion of similar cluster centers using equation(9);
Step 5: Update the Matrix B by solving the problem(10);
Step 6: Compute the object function value by formula(6);
Step 7: End

4 Experiments

Extensive experiments have been carried out to demonstrate the effectiveness of ML-CC[1]. We first visualize the mechanism of the proposed ML-CC on a synthetic dataset. Then we compare the performance of the proposed ML-CC method with the NCA [8], MCC [10], MCML [7], LMNN [17], PFLMNN [18], RVML [15], AML [32], and ASTCML [23] on nine UCI datasets. Moreover, the parametric sensitivity of ML-CC is analyzed. Finally, we validate the effectiveness of the method on face dataset.

4.1 Experiments on synthetic dataset

To demonstrate the effectiveness of the proposed method, we validate the performance with the classical nonlinear synthetic dataset Two-moon. The dataset contains 200 points across two classes. In Fig. 2a, different colors represent different classes. The black and blue circles are the cluster centers corresponding to the light green and light yellow data points, respectively.

In this experiment, we use Two-moon to learn the clustering centers and metrics with ML-CC-1, which is a one-iteration version of ML-CC. In this case, the data points are clustered only once, and the metric is learned based on the clustering results. It is easy to confirm from Fig. 2b–d that as the value of β increases, more and more cluster centers overlap each other, and when $\beta = 100$, the cluster centers of each class overlap to one point. This result shows that the number of cluster centers can be adjusted by the value of β.

Moreover, when the weight parameter K adopts different values, it also has a significant impact on the result and limits the upper limit of the cluster centers merger (as indicated in Fig. 3a–d). K controls the size of the fused nearest neighbors and B controls the degree of nearest neighbor fusion. To facilitate the parameter adjustment, we adopt a larger value of β ($\beta = 100$) to maximize the integration of the cluster centers. By adjusting the value of the weight K, the upper limit of the algorithm cluster center's fusion degree increases.

4.2 Experiments on classification

To compare the performances of the different methods on the classification task, we adapt the 3-NN classification

[1] https://github.com/array12138/metric-leanring.

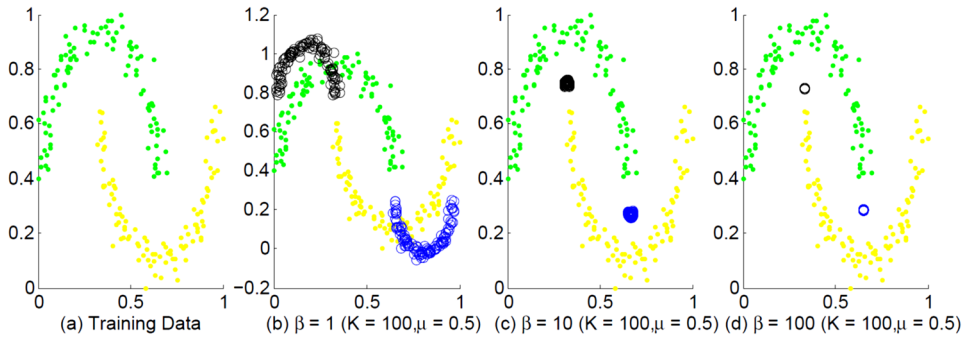

Fig. 2 The cluster centers change as parameter β gets larger. ($K = 100$, $\mu = 0.5$)

Table 1 Classification accuracy of 3-nearest neighbor classifiers

Methods	Auto 25,6,205	Balance 4,3,625	Glass 9,6,214	German 20,2,1000	Heart 13,2,270	Monk1 6,2,432	Pima 8,2,768	Verhicle 18,4,846	Wilt 5,2,4839
kNN	0.532	0.787	0.585	0.677	0.778	0.800	0.732	0.728	0.628
NCA	0.516	0.926	0.600	*0.717*	0.765	0.954	0.719	0.720	*0.862*
MCC	0.532	0.894	0.585	0.700	0.778	0.777	0.740	0.728	0.708
MCML	*0.548*	0.920	0.600	*0.717*	0.778	0.508	*0.749*	0.748	0.764
LMNN	0.500	0.787	0.585	0.670	*0.803*	0.908	0.710	*0.780*	0.842
ITML	0.516	0.894	0.539	0.700	0.790	0.790	0.697	0.744	0.820
RVML	*0.548*	0.888	0.523	0.713	0.753	0.639	0.706	0.610	0.718
FLMNN	0.532	0.819	0.585	0.703	*0.803*	0.931	0.732	0.768	0.836
AML	0.516	0.782	*0.615*	0.687	0.790	0.777	0.732	0.740	0.802
ASTCML	0.532	*0.931*	*0.615*	0.657	0.790	*0.962*	0.719	*0.780*	0.764
ML-CC-1	**0.597**	**0.947**	*0.615*	0.683	*0.803*	0.923	0.727	0.752	**0.866**
ML-CC	**0.597**	**0.947**	**0.631**	**0.723**	**0.815**	**0.977**	**0.753**	**0.791**	*0.862*

based on the learned metrics to perform the experimental analysis. The experiments have been conducted on nine datasets from the *UCI* machine learning repository, including *Auto*, *Balance*, *Glass*, *German*, *Heart*, *Monk*1, *Pima*, *Verhicle*, *Verhicle*, and *Wilt*. The performance of the proposed ML-CC is compared with ten classical methods of metric learning (kNN, NCA, MCC, MCML, LMNN, ITML, RVML, FLMNN, AML and ASTCML).

We have compared all the methods with the proposed method over 5-fold cross-validation. In each trial, 70% of the data points are randomly selected as the training set; the remaining are used for testing, where the dataset Wilt has a training set and a testing set. The distributions of all classes in the training set and testing set are the same as those in the original dataset. In our experiments, we have performed metric learning on the training set and used a 3-NN classifier

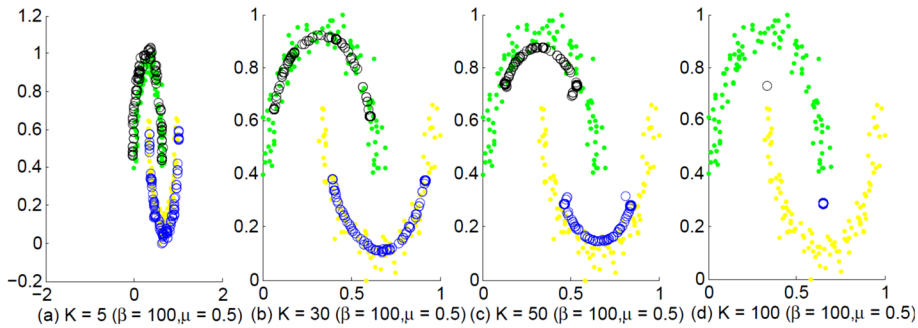

Fig. 3 The cluster centers change as parameter K gets larger. ($\beta = 100$, $\mu = 0.5$)

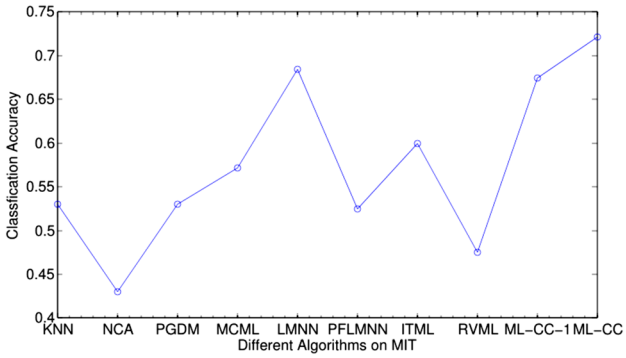

Fig. 4 3-NN classification based on different metric learning methods for the MIT face image dataset

to classify the testing set based on the learned metrics to verify the metric learning algorithm's performance.

The parameter β is set to 100 in the proposed method. The weight parameter K is adjusted so that $w_{ij} = 1$ if j is a neighbor of i and have the same label, otherwise $w_{ij} = 0$. The parameters K and μ in the proposed method are adjusted by searching the grid $\{2, 4, 8, 16, 32 \ldots, \} \times \{0.1, 0.3, \ldots, 0.9\}$. The stopping condition of the proposed model solving process is that the objective function value no longer decreases or the number of iterations reaches the maximum number of iterations we allow.

The parameters for the comparative algorithms are also carefully tuned to achieve their optimal performance. In Table 1, the three numbers below each dataset correspond to the number of dimensionalities, number of classes, and the number of samples. We highlight the best results in each dataset in boldface, and sub-optimal results in light blue. The experimental results indicate that ML-CC has significant advantages on most datasets, which suggests that clustering data with cluster centers as target samples and clusters as target neighbors can better exploit the local properties of data. Besides, ML-CC-1 also shows acceptable performance, which indicates that the metric obtained based on the nearest neighbor structure of single cluster mining can also perform well for classification.

4.3 Experiments on face images

We use the *MIT CBCL Face Database*[2] to validate the proposed classification method. The database is divided into a training set and a testing set. The training set contains 3240 synthetic face images with ten different subjects, each containing 324 images, and the testing set contains 2000 images with ten different subjects, each containing 200 images.

We represent each image as a 400-dimensional feature vector, which is a straightened gray vector, by image cropping to visualize the results. We select the first three classes in the training set (972 face images) and the first three classes in the testing set (600 face images). The parameters in the proposed method, such as K and μ, are tuned by searching the grid $\{2, 4, 8, 16, 32...\} \times \{0.1, 0.3, ..., 0.9\}$, $\beta = 100$, and the maximum value of K is set to the number of samples in the smallest class.

After obtaining the metric, we apply the metric to the 3-NN to classify the images in the testing set, and the experimental results are displayed in Fig. 4. From the results, one can see that for the AML method, it is difficult to obtain results in an acceptable time, and ML-CC obtains the best classification accuracy.

4.4 Parametric sensitivity

In the proposed ML-CC, three parameters β, μ, and K can affect the model performance. The parameter β controls the number of cluster centers. When β is sufficiently large, the number of cluster centers becomes small.

To reduce the size of the parameter, we directly set β to 100. The parameter μ is a trade-off parameter. The weight parameter K controls the upper limit of cluster center overlap. When β is sufficiently large, as K increases, the number of cluster centers decreases until they overlap into one point. In Fig. 5, we change the value of K, μ and record the average classification accuracy on the validation set in the 5-fold cross-validation for nine datasets, where the parameter K varies within the set [2,4,8,16,32,64] and the parameter μ within the set [0.1,0.3,0.5,0.7,0.9]. From the experimental results, μ has a low impact on the model and K has a high impact on the model. This is because tuning up K helps mine more important constraints for dataset, and thus it makes the metric learning model to improve the quality of the learned metric.

5 Conclusion

This paper proposes a novel metric learning method called ML-CC, which exploits the learned cluster centers to generate a large margin constraint for metric learning. Because the number of different cluster centers is considerably smaller

[2] http://cbcl.mit.edu/software-datasets/heisele/facerecognition-database.html.

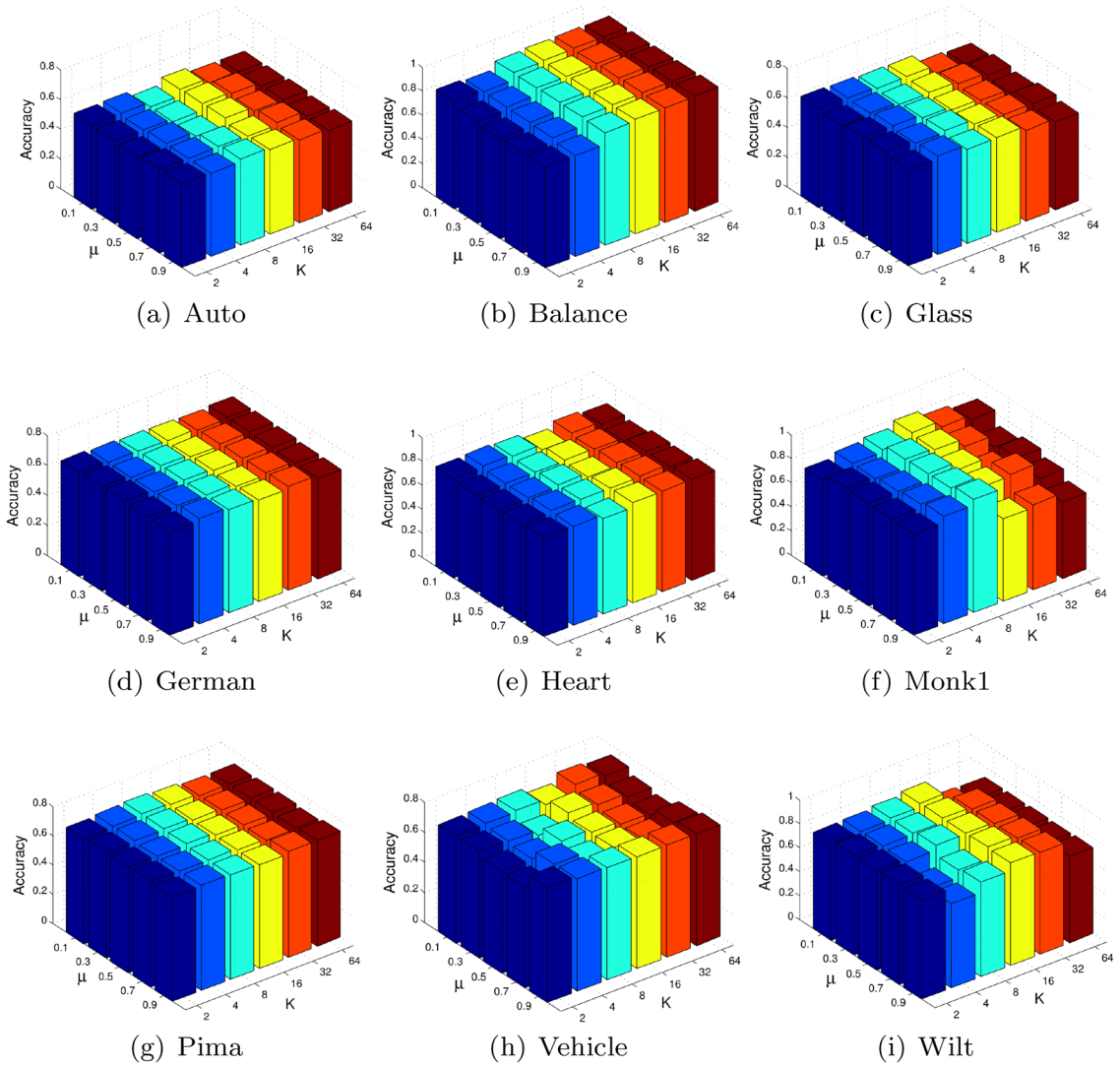

Fig. 5 Sensitivity analysis of parameters under different data sets

than the number of data points, the proposed method can significantly reduce the search space for solving the optimization problem. Our study shows that updating the cluster centers by clustering and constructing a large marginal constraint derived from the cluster centers with a considerable residual constraint can improve metric learning. However, our current approach still suffers from an enormous computational burden. An interesting future issue would be to study how to use clustering methods to construct simpler and more effective constraints between the generated cluster centers and the real samples.

Acknowledgements This work was supported by the National Natural Science Foundation of China (Nos. 61976184, 62006147, 61772323), the Projects of Key Research and Development Plan of Shanxi Province (201903D121162) and the 1331 Engineering Project of Shanxi Province, China.

References

1. Dong Y, Shi W, Du B, Hu X, Zhang L (2021) Asymmetric weighted logistic metric learning for hyperspectral target detection. IEEE Trans Cybern 1–14
2. Lv J, Kang Z, Lu X, Xu Z (2021) Pseudo-supervised deep subspace clustering. IEEE Trans Image Process 30:5252–5263
3. Jin Y, Li C, Li Y, Peng P, Giannopoulos GA (2021) Model latent views with multi-center metric learning for vehicle re-identification. IEEE Trans Intell Transp Syst 22(3):1919–1931
4. Nguyen HV, Bai L (2010) Cosine similarity metric learning for face verification. In: Asian conference on computer vision, pp 709–720
5. Norouzi M, Fleet DJ, Salakhutdinov RR (2012) Hamming distance metric learning. In: Advance neural information processing systems, pp 1061–1069
6. Xu J, Luo L, Deng C et al (2018) Multi-level metric learning via smoothed Wasserstein distance. In: Proceedings of the 27th international joint conference on artificial intelligence, pp 2919–2925

7. Globerson A, Roweis ST (2006) Metric learning by collapsing classes. In: Advance neural information processing systems, pp 451–458
8. Goldberger J, Hinton GE, Roweis ST et al (2005) Neighbourhood components analysis. In: Advance neural information processing systems, pp 513–520
9. Yang Z, Laaksonen J (2007) Regularized neighborhood component analysis. In: Scandinavian conference on image analysis, pp 253–262
10. Xing EP, Jordan MI, Russell SJ et al (2003) Distance metric learning with application to clustering with side-information. In: Advance neural information processing systems, pp 521–528
11. Fetaya E, Ullman S (2015). Learning local invariant mahalanobis distances. In: International conference on machine learning, pp 162–168
12. Wei S, Li Z, Zhang C (2017) Combined constraint-based with metric-based in semi-supervised clustering ensemble. Int J Mach Learn Cybern 9(7):1085–1100
13. Shuang X, Yang M, Zhou Y et al (2020) Partial label metric learning by collapsing classes. Int J Mach Learn Cybern 11(7):2453–2460
14. Ying Y, Li P (2012) Distance metric learning with eigenvalue optimization. J Mach Learn Res 13(1):1–26
15. Perrot M, Habrard A (2015). Regressive virtual metric learning. In: Advance neural information processing systems, pp 1810–1818
16. Zuo W, Wang F, Zhang D et al (2017) Distance metric learning via iterated support vector machines. IEEE Trans Image Process 26(10):4937–4950
17. Weinberger KQ, Saul LK (2009) Distance metric learning for large margin nearest neighbor classification. J Mach Learn Res 10(2):207–244
18. Song K, Nie F, Han J et al (2017) Parameter free large margin nearest neighbor for distance metric learning. In: Proceedings of the thirty-first AAAI conference on artificial intelligence, pp 2555–2561
19. Zhan D, Li M, Li Y et al (2009) Learning instance specific distances using metric propagation. In: Proceedings of the 26th annual international conference on machine learning, pp 1225–1232
20. Wang J, Woznica A, Kalousis A (2012) Learning neighborhoods for metric learning. In: Joint European conference on machine learning and knowledge discovery in databases, pp 223–236
21. Davis JV, Kulis B, Jain P et al (2007) Information-theoretic metric learning. In: Proceedings of the 24th international conference on machine learning, pp 209–216
22. Le Capitaine H (2018) Constraint selection in metric learning. Knowl-Based Syst 146:91–103
23. Wang X, Guo X, Wei W, Liang J (2021) Metric learning algorithm with adversarial sample triples constraints. CAAI Trans Intell Syst 16(1):30–37
24. Sui XL, Li X, Qian X et al (2018) Convex clustering with metric learning. Pattern Recogn 81:575–584
25. Kang Z, Lin Z, Zhu X, Xu W (2021) Structured graph learning for scalable subspace clustering: from single view to multiview. IEEE Trans Cybern
26. Kang Z, Pan H, Hoi SCH, Xu Z (2019) Robust graph learning from noisy data. IEEE Trans Cybern 50(5):1833–1843
27. Zhang L, Zhang Q, Du B, You J, Tao D (2017) Adaptive manifold regularized matrix factorization for data clustering. In: International joint conference on artificial intelligence, pp 3399–3405
28. Nie F, Cai G, Li X (2017) Multi-view clustering and semi-supervised classification with adaptive neighbours. Proc AAAI Conf Artif Intell 31:2408–2414
29. Li X, Yin H, Zhou K, Zhou X (2020) Semi-supervised clustering with deep metric learning and graph embedding. World Wide Web 23(2):781–798
30. Elezi I, Vascon S, Torcinovich A, Pelillo M, Leal-Taixé L (2020) The group loss for deep metric learning. In: European conference on computer vision. Springer, pp 277–294
31. Qi C, Zhang J, Jia H, Mao Q, Wang L, Song H (2021) Deep face clustering using residual graph convolutional network. Knowl-Based Syst 211:106561
32. Chen S, Gong C, Yang J et al (2018) Adversarial metric learning. In: Proceedings of the 27th international joint conference on artificial intelligence, pp 2021–2027