Information Sciences 621 (2023) 341–353

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Multiple metric learning via local metric fusion

Xinyao Guo [a], Lin Li [a], Chuangyin Dang [b,*], Jiye Liang [a], Wei Wei [a]

[a] 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
[b] Department of Advanced Design and Systems Engineering, City University of Hong Kong, Kowloon, Hong Kong

ARTICLE INFO

Article history:
Received 26 April 2022
Received in revised form 20 November 2022
Accepted 24 November 2022
Available online 30 November 2022

Keyword:
Metric learning
Multiple metric learning
Metric fusion

ABSTRACT

Adaptive distance metric learning based on the characteristics of data can significantly improve the learner's performance. Due to the limitations of single metric learning for heterogeneous data, multiple local metric learning has become an essential representative tool to describe local properties of data. Most existing multiple metric learning algorithms need to perform metric learning on a pre-obtained instance division. However, the number of clusters in the pre-obtained division affects the effectiveness of metric learning. To tackle this problem, we propose a Multiple Metric Learning via Local Metric Fusion (MML-LMF) framework, which unifies local metric learning and fusion of similar local metrics into one metric and adaptively determines the number of local metrics. As an application of the MML-LMF framework to pairwise constraints, we devise a MML-LMF algorithm by constructing a concrete optimization model and acquiring a closed-form solution to the model. The experimental results on several benchmarks, person re-identification, and face verification datasets show that the performance of the proposed algorithm is superior to that of the existing state-of-the-art global and multiple metric learning algorithms.

1. Introduction

It is well known that machine learning algorithms are sensitive to a given distance or similarity measure [1–3]. However, it is often difficult to capture the underlying semantic space of a given problem with standard metrics such as the Euclidean distance. To overcome this deficiency, several metric learning algorithms have been developed in the literature [4–6]. Mahalanobis distance metric learning is one of the most common metric learning algorithms [7–9], which is equivalent to the Euclidean distance under a linear transformation. In many applications, one can replace the Euclidean distance with the Mahalanobis distance.

Generally, metric learning algorithms that learn a uniform metric are called single metric learning or global metric learning. The typical single metric learning algorithms include NCA (Neighbourhood Component Analysis) [10], LMNN (Large Margin Nearest Neighbor) [11], GMML (Geometric Mean Metric Learning) [12], GNSML (Global Nonlinear Smooth Metric Learning) [13], and CDML (Curvilinear Distance Metric Learning) [14]. Although the single metric learning algorithms are effective for data with simple structures, they fail to learn the distance for datasets with complex nonlinear structures. To address this issue, several strategies have been proposed in the literature [15,16], which includes kernel tricks, deep

* Corresponding author.
E-mail addresses: 1303590343@qq.com (X. Guo), lilynn1116@163.com (L. Li), mecdang@cityu.edu.hk (C. Dang), ljy@sxu.edu.cn (J. Liang), weiwei@sxu.edu.cn (W. Wei).

https://doi.org/10.1016/j.ins.2022.11.118
0020-0255/

embedding, and multiple metric learning. The kernel tricks map the input data into a high-dimensional feature space, in which a linear transformation can effectively separate the data [17,18]. Nevertheless, the kernel tricks suffer from enormous computational cost and have difficulty in choosing the kernel, which significantly limits their applications. The deep embedding has powerful feature representation capability, but it lacks interpretability and requires high data scaling. As a result of its good interpretability and low computational cost, multiple metric learning has received much attention.

Multiple metric learning yields a good balance between the fitting power and the complexity of the model and is thus suitable for more complex nonlinear data. Given the number of metrics, different metrics can be attained with multiple metric learning based on particular instances [19,20], local clusters [21], and selected bases [22]. MMLMNN (Multiple Metric Large Margin Nearest Neighbor) [11] is an extension of LMNN, which learns each class's corresponding local linear transformation while making similar and dissimilar instances with larger intervals. SCML (Sparse Compositional Metric Learning) [23] uses Fisher discriminant analysis to extract basic metrics in different local data regions and learns sparse linear combinations of these basic metrics for each local region. CMML (Clustered Multi-Metric Learning) [18] first uses clustering methods to cluster data into multiple clusters and then constructs triple constraints to learn a local metric for each cluster, where the local metric should be consistent with the global metric as much as possible to reduce the risk of overfitting. However, these algorithms require both specifying the number of local metrics and learning a single local metric for a cluster according to a fixed data partition in advance. To get rid of these requirements, a LIFT (Local Metrics Facilitated Transformation) framework was proposed in [21]. The LIFT framework learns the global and multiple local metrics jointly. Specifically, each local metric is derived from the global metric through a local bias optimization that takes into account the local properties of data. The LIFT framework learns a metric for every cluster, but it needs to specify the number of clusters in advance. Nonetheless, the changes in the number of clusters can cause a significant increase in the computational cost of LIFT.

To tackle the deficiencies with the existing multiple metric learning algorithms discussed above, we propose a Multi-Metric Learning via Local Metric Fusion (MML-LMF) framework, which unifies learning local metrics for constraints and fuses similar local metrics into one metric. The basic idea of the framework is as follows. Suppose that we initially obtain N local metrics based on N pairs of constraints. As similar local metrics are continuously fused into one metric, the number of different local metrics decreases. The framework uses a classifier to verify the classification accuracy of the current number of local metrics, records the classification accuracy on the classifier at each change of the number of local metrics, and chooses the number of local metrics according to the optimal classification accuracy. Based on the MML-LMF framework, we acquire a concrete optimization model with pairwise constraints and derive a closed-form solution to this optimization model. Furthermore, we have carried out extensive experiments on several benchmarks, person re-identification, and face verification datasets to demonstrate the performance of the proposed algorithm. The main contributions of this paper can be summarized as follows:

- We propose a novel Multi-Metric Learning via Local Metric Fusion (MML-LMF) framework. This kind of local metrics in terms of constraints can enlarge the parameter space, thus allowing a better fitting power for the complex data distribution.
- Using pairwise constraints, we construct a concrete optimization model based on the MML-LMF framework. Furthermore, we obtain a closed-form solution to the optimization model.
- We have carried out extensive experiments on several benchmarks, person-identification, and face verification datasets. The results show that the proposed algorithm outperforms the existing state-of-the-art algorithms.

2. Preliminaries

2.1. Notations

Let $\mathscr{D} = \{(x_i, y_i)\}_{i=1}^{N}$ denote a training dataset of N instances, where $x_i \in \mathbb{R}^d$ and $y_i \in \{1, 2, \cdots, C\}$ (C is the number of classes). The side information extracted from $\mathscr{D}$ is composed of pairwise constraints or triplet constraints. Let $\mathscr{T}_0 = \{(x_i, x_j, q_{ij})\}$ denote the set of all pairwise constraints for N instances, where q_{ij} indicates whether the corresponding pairs belong to the same class or not. Let $(i, j) \leftrightarrow t$ be a one-to-one mapping, where t indicates that (x_i, x_j) is the tth constraint in $\mathscr{T}_0$. If $y_i = y_j$, then $q_{ij} = q_t = 1$; otherwise, $q_{ij} = q_t = -1$. Let $|\mathscr{T}_0|$ be the number of pairwise constraints. We have $|\mathscr{T}_0| = N(N-1)/2$. Note that $x_i - x_j = v_{ij} = v_t$. Given two instances x_i and x_j, the Mahalanobis distance is defined as: $d_M^2(x_i, x_j) = (x_i - x_j)^T M (x_i - x_j) = \|L(x_i - x_j)\|_2^2 = \|L v_t\|_2^2$, where M is a positive semi-definite (PSD) matrix with $M = LL^T$.

2.2. The Local Metrics Facilitated Transformation (LIFT) framework

The global Mahalanobis metric M_0 describes a uniform type of feature relationship between instances. Most algorithms first divide the instances and then describe the different localities (instance clusters) of data using a single metric M_k. Instead of learning M_k, Ye et al. [21] learn a global metric plus a local metric bias for each locality, i.e., $M_k = M_0 + \Delta M_k$.

To eliminate the computational cost of ensuring the positive semi-definiteness of M, Ye et al. attempt to learn multiple transformations $L_k = L_0 + \Delta L_k$, where L_k, L_0, and ΔL_k are the corresponding transformation matrices of M_k, M_0, and ΔM_k, respectively. When data are clustered into K clusters, they define the kth cluster as $\mathscr{N}_k = \{(x_i^k, y_j^k)\}_{i=1}^{N_k}$, where N_k is the number

of instances in the kth cluster. Let $\mathscr{T}_k = \{(x_i^k, x_j^k, q_{ij}^k)\}$ be the set of pairwise constraints extracted from $\mathscr{N}_k$ with $|\mathscr{T}_k| = N_k(N_k-1)/2$. The framework of LIFT is as follows:

$$\min_{\mathscr{L}} \ \mu_1 \sum_{(x_i,x_j,q_{ij})\in\mathscr{T}_0} l(q_{ij}(\gamma - d^2_{L_0}(x_i,x_j))) + \lambda_1\|L_0\|_F^2 + \mu_2\sum_{k=1}^{K}\sum_{(x_i^k,x_j^k,q_{ij}^k)\in\mathscr{T}_k} l(q_{ij}^k(\gamma - d^2_{L_0+\Delta L_k}(x_i^k,x_j^k))) + \lambda_2\|\Delta L_k\|_F^2,$$

where γ is a pre-defined non-negative threshold value, $\mathscr{L} = \{L_0, \{\Delta L_k\}_{k=1}^K\}, \mu_1, \mu_2, \lambda_1$, and λ_2 are non-negative trade-off hyper-parameters, $\|\cdot\|_F$ is the matrix Frobenius norm, and $l(\cdot)$ is a convex non-increasing loss function.

In LIFT, once the global metric L_0 is competent for measuring local losses compared with local metric $L_0 + \Delta L_k$, the local metric bias ΔL_k becomes zero. The number of local metrics will thus be small so that the model complexity of LIFT is reduced. However, getting a low loss with a single metric for a local cluster may be challenging when the number of clusters is small. When the number of clusters is large enough, it also needs a high computational cost to ensure that different local metrics are dissimilar. Therefore the number of clusters will significantly affect the performances of the algorithms based on the LIFT framework.

3. Learning Multi-Metric via Local Metric Fusion

This section proposes a Multi-Metric Learning via Local Metric Fusion (MML-LMF) framework. As an application of this framework to pairwise constraints, we acquire a much easier multiple metric learning optimization model, which has a closed-form solution.

3.1. The MML-LMF Framework

Let $\mathscr{T}$ be a set of constraints extracted from $\mathscr{D}$, where the constraints can be pairwise constraints, triple constraints, quadruplet constraints, etc. With this set of constraints, we obtain the most fine-grained local metric and learn a metric L_t for tth constraint $\mathscr{T}^t$ in $\mathscr{T}$. Let $L_t = L_0 + \Delta L_t$, where L_0 is a global metric and ΔL_t metric is found with

$$\min_{\Delta L_t} l(f(\mathscr{T}^t, L_0 + \Delta L_t, \gamma)) + \lambda\|\Delta L_t\|_F^2, \tag{1}$$

where γ is a pre-defined non-negative threshold value, λ is a non-negative trade-off hyper-parameter, $l(\cdot)$ is a convex non-increasing loss function, and $f(\cdot)$ is an operator defined on the constraint $\mathscr{T}^t$, which describes how close the current constraint is to the threshold γ under the metric $L_0 + \Delta L_t$. For example, in pairwise constraints, $f(\cdot) = q_t(\gamma - \|(L_0 + \Delta L_t)v_t\|_2^2)$ defines the distance between similar (dissimilar) instances in comparison with a certain threshold γ.

The above procedure learns a global metric L_0 on the constraint set $\mathscr{T}$, which can well distinguish most of the constraints. To deal with those undistinguishable constraints, the MML-LMF framework further requires aggregating the constraints with similar non-zero metric biases. In the aggregation process, the number of different local metrics decreases so that the complexity of the model is reduced. The MML-LMF framework can be stated as follows:

$$\min_{\mathscr{L}} \sum_{t=1}^{|\mathscr{T}|} l(f(\mathscr{T}^t, L_0, \gamma)) + \lambda_1\|L_0\|_F^2 + \mu\sum_{t=1}^{|\mathscr{T}|} l(f(\mathscr{T}^t, L_0 + \Delta L_t, \gamma))) + \lambda_2\|\Delta L_t\|_F^2 + \lambda_3 \sum_{\substack{1\leqslant j_1<j_2\leqslant|\mathscr{T}| \\ \Delta L_{j_1},\Delta L_{j_2}\neq 0}} w_{\{j_1 j_2\}}\|\Delta L_{j_1} - \Delta L_{j_2}\|_F^2 \tag{2}$$

where $\mathscr{L} = \{L_0, \{\Delta L_t\}_{t=1}^{|\mathscr{T}|}\}, \mu, \lambda_1, \lambda_2$, and λ_3 are non-negative trade-off hyper-parameters. $w_{\{j_1 j_2\}} = exp(-\delta\|\Delta L_{j_1} - \Delta L_{j_2}\|_F^2)$ is a non-negative weight (δ is a pre-defined non-negative value), which can be used to control the scope of aggregating metric biases, and j_1 and j_2 mean the j_1th and j_2th constraints in $\mathscr{T}$, respectively.

3.2. An Application of the MML-LMF Framework to Pairwise Constraints

An intuitive drawback of the MML-LMF framework is the low efficiency due to the enormous number of constraints. To overcome this drawback, this section employs a simple and efficient constraint selection method [24,25] to select pairwise constraints. Furthermore, we attain an optimal solution approach in which sub-optimization problems are transformed into closed-form solutions. The approach only involves matrix addition and multiplication operations. Finally, the computational complexity of the algorithm is analyzed.

Under the global metric L_0 for pairwise constraints, most of the constraints already satisfy the loss $l(\cdot)$, i.e., similar instances are less than or equal to a certain threshold, and dissimilar instances greater than or equal to a certain threshold, where we take $l(x) = \max\{1, -x\}$ and $\gamma = 2$. The proposed model learns a metric bias ΔL_t for the tth constraint $\mathscr{T}^t = (x_i, x_j, q_t)$ in an unsatisfied loss, i.e., $\mathscr{T}^t \in \mathscr{T}_0$ and $\Delta L_t \neq 0$, where the unsatisfied loss implies that $l(x) = -x$. Let $\mathscr{T}_u = \{\mathscr{T}^t | \mathscr{T}^t \in \mathscr{T}_0$ and $\Delta L_t \neq 0\}$. We fuse the metrics with similar metric biases into one metric and obtain the following optimization problem:

$$\min_{\{\Delta L_t\}_{t=1}^{|\mathscr{T}_u|}} \sum_{t=1}^{|\mathscr{T}_u|} l(q_t(\gamma - \|(L_0+\Delta L_t)v_t\|_2^2)) + \lambda_2\|\Delta L_t\|_F^2 + \lambda_3 \sum_{1\leqslant j_1<j_2\leqslant|\mathscr{T}_u|} w_{\{j_1 j_2\}}\|\Delta L_{j_1} - \Delta L_{j_2}\|_F^2,$$

where $v_t = x_i - x_j, l(x) = -x$, and j_1 and j_2 mean the j_1th and j_2th constraints in $\mathscr{T}_u$, respectively. The purpose of the last term is to enforce similarity among $\{\Delta L_t\}_{t=1}^{|\mathscr{T}_u|}$. When $\Delta L_{j_1} = \Delta L_{j_2} = \cdots = \Delta L_{j_k}$, the k constraints form a cluster, and thus generate a local metric. Therefore, the last term can effectively control the number of local metrics to avoid the overfitting. Especially the algorithm generates the local metric from the perspective of constraints and has more parameters, which allows a better fitting capability for complex data distributions.

In a practical training phase, it is optional to use all pairwise constraints in $\mathscr{T}_0$ for metric learning. Most algorithms construct similarity constraints with k_1 nearest neighbors of the same label and dissimilar constraints with k_2 nearest neighbors of different labels, where k_1 and k_2 usually belong to $\{1,2,3\}$. A similar strategy for constructing pairwise or triplet constraints can be found in [24,25]. Let $\mathscr{T}_{k_1\&k_2}$ denote the constraint set consisting of each instance x_i and its k_1 and k_2 nearest neighbors. We modify the constraint set $\mathscr{T}_u$ as $\hat{\mathscr{T}}_u = \{\mathscr{T}^t | \mathscr{T}^t \in \mathscr{T}_{k_1\&k_2}, \Delta L_t \neq 0\}$. For the tth ΔL_t, the optimization model is:

$$\min_{\Delta L_t} q_t(\|(L_0+\Delta L_t)v_t\|_2^2 - \gamma) + \lambda_2\|\Delta L_t\|_F^2 + \lambda_3 \sum_{j\neq t}^{|\hat{\mathscr{T}}_u|} w_{tj}\|\Delta L_t - \Delta L_j\|_F^2, \tag{3}$$

where $v_t = x_i - x_j, \mathscr{T}^t = (x_i, x_j, q_t)$ is the tth constraint in $\hat{\mathscr{T}}_u$, and w_{tj} is an abbreviation of $w_{\{t,j\}}$.

3.3. Optimization

To solve the problem (3), one needs to initialize the metric bias $\{\Delta L_t\}_{t=1}^{|\hat{\mathscr{T}}_u|}$ and solve the following optimization problem,

$$\min_{\Delta L_t} q_t(\|(L_0+\Delta L_t)v_t\|_2^2 - \gamma) + \lambda_2\|\Delta L_t\|_F^2. \tag{4}$$

Taking the derivative of formula (4), we obtain:

$$\Delta L_t = -q_t(\lambda_2 I + q_t v_t v_t^T)^{-1} v_t v_t^T L_0, \tag{5}$$

where I is an identity matrix. An application of the Sherman-Morrison formula to $(\lambda_2 I + q_t v_t v_t^T)^{-1}$ yields

$$\Delta L_t = -\frac{q_t}{\lambda_2}(I - \frac{q_t v_t v_t^T}{\lambda_2 + q_t v_t^T v_t}) v_t v_t^T L_0. \tag{6}$$

After the initialization of ΔL_t, we get from the formula (3)

$$\Delta L_t = \frac{1}{\alpha}(I - \frac{q_t v_t v_t^T}{\alpha + q_t v_t^T v_t})(\lambda_3 \sum_{j\neq t}^{|\hat{\mathscr{T}_u}|} w_{jt}\Delta L_j - q_t v_t v_t^T L_0), \tag{7}$$

where $\alpha = \lambda_2 + \lambda_3 \sum_{j\neq t}^{|\hat{\mathscr{T}_u}|} w_{tj}$.

As the metrics merge continuously, there will be more and more identical metrics. Let G be the constraints set such that $\forall \mathscr{T}^{t_1}, \mathscr{T}^{t_2} \in G$, we have $\Delta L_{t_1} = \Delta L_{t_2} \neq 0$. Then $\sum_{t=1}^{|G|} \partial f(\Delta L_t)/\partial \Delta L_t = 0$. Thus,

$$\Delta L_G = \Delta L_t = (\hat{\alpha} I + \sum_{t=1}^{|G|} q_t v_t v_t^T)^{-1} \cdot (\lambda_3 \sum_{j=1}^{|\hat{\mathscr{T}}_u|} w_{Gj}\Delta L_j - \sum_{t=1}^{|G|} q_t v_t v_t^T L_0), \tag{8}$$

where j indicates the jth constraint in $\hat{\mathscr{T}}_u$, $\hat{\alpha} = \lambda_2|G| + \lambda_3 \sum_{j=1}^{|\hat{\mathscr{T}}_u|} w_{Gj}$ and $w_{Gj} = \sum_{t=1}^{|G|} w_{tj}$. The matrix inversion process in the formula (8) can be replaced by the iteration in Lemma 1.

Lemma 1. $K_t = (\hat{\alpha} I + \sum_{t=1}^{|G|} q_t v_t v_t^T)^{-1}$ in the formula (8) can be updated as

$$\begin{cases} K_0 = \frac{1}{\hat{\alpha}} I, \\ K_1 = (K_0^{-1} + q_1 v_1 v_1^T)^{-1} = K_0 - \frac{q_1 K_0 v_1 v_1^T K_0}{1 + q_1 v_1^T K_0 v_1}, \\ \cdots, \\ K_t = K_{t-1} - \frac{q_t K_{t-1} v_t v_t^T K_{t-1}}{1 + q_t v_t^T K_{t-1} v_t}, \end{cases}$$

where K_t can be obtained by t iterations and the derivation is simply a repeated application of the Sherman-Morrison formula.

When the two metric biases ΔL_{G_1} and ΔL_{G_2} are similar, we need to fuse them into a new metric bias ΔL_G, i.e.

$$\Delta L_G = \frac{|G_1|\Delta L_{G_1} + |G_2|\Delta L_{G_2}}{|G_1| + |G_2|}. \tag{9}$$

To judge whether two metric biases should be merged, we use a small threshold on $\|\Delta L_{G_1} - \Delta L_{G_2}\|_F^2$, which we set to be $1.05 \cdot \min_{ij}\|\Delta L_{G_i} - \Delta L_{G_j}\|_F^2$.

3.4. Global Metric Learning of MML-LMF

When learning global metric L_0, the $l(\cdot)$ for global and local regularizers should be different. To facilitate the optimization process, we choose the smooth hinge loss $l_s(\cdot)$ for global metric learning, which is defined as:

$$l_s(x) = \begin{cases} 0 & if\ x > 1 \\ \frac{1}{2}(x-1)^2 & if\ 0 \leqslant x \leqslant 1 \\ \frac{1}{2} - x & if\ x < 0. \end{cases}$$

When λ_2 is sufficiently large and $\lambda_3 = 0$, the MML-LMF in formula (2) degenerates to a global distance metric learning algorithm. With $l_s(\cdot)$, the optimization problem formula (2) becomes:

$$\min_{L_0} \sum_{t=1}^{|\mathscr{T}_0|} l_s(q_t(\gamma - \|L_0 v_t\|_2^2)) + \lambda_1 \|L_0\|_F^2. \tag{10}$$

For the global metric learning, the solution can be simply attained by the gradient descent method, which is similar to that of LMNN. The summary of the application of the MML-LMF framework to pairwise constraints is presented in Algorithm 1.

3.5. Computational Complexity

In Algorithm 1, the computational complexity of MML-LMF is mainly concentrated in the loop process of the metric fusion, where the number of metrics determines the number of loop iterations. The maximum number of metrics is nk, where k is the number of selected nearest neighbors. The computational complexity of formula (8) and formula (9) are $o(nkd^3)$ and $o(n^2k^2d^2)$, respectively. The algorithm terminates when the number of metrics equals t. Therefore, the computational complexity of algorithm 1 is $o(t(nkd^3 + n^2k^2d^2))$. With the help of the global metric, the number of metrics t is much smaller than nk. Therefore, the efficiency of the algorithm is relatively high.

Algorithm 1: MML-LMF

Input: X: data matrix; Y: label vector; $\lambda_1, \lambda_2, \lambda_3, \gamma = 2; k_1, k_2$: neighborhood numbers.

Output: $\{\Delta L_t\}_{t=1}^{|\hat{\mathscr{T}}_u|}$: metric biases.

1: Learning the global metric L_0 by solving optimization problems (10).
2: Select partial pairwise constraints set $\hat{\mathscr{T}}_u$ based on the number of similar nearest neighbors k_1 and the number of dissimilar nearest neighbors k_2.
3: Initialize the metric biases $\{\Delta L_t\}_{t=1}^{\hat{\mathscr{T}}_u}$ using formula (6).
4: Calculate the weight matrix $w_{ij} = exp(-\delta\|\Delta L_i - \Delta L_j\|_F^2)$, where $\delta = 1$.
5: **while** true**do**
6: Update the metric biases ΔL_t using formula (8).
7: Fusion of similar metric biases is performed using formula (9).
8: Let $|\Delta L_t|$ be number of different metric biases.
9: if($|\Delta L_t|$==2) or ($|\Delta L_t| <$ threshold)
10: break;
11: end
12: **end while**

4. Experiments on Benchmark Datasets

This section investigates the performances of some representative single metric learning and multiple metric learning algorithms on 15 benchmark datasets: Auto(205, 25, 6), Balance(625, 4, 3), Breast(699, 10, 2), Cars(392,8,3), Chess(3196, 36, 2), Cleve(303, 13, 4), Glass(214, 9, 6), Heart(270, 13, 2), ILPD(583, 10, 2), Letter(20000, 16, 26), Pima(768, 8, 2), Segment(2310, 19, 7), Solar(323, 12, 6), Vote(435, 16, 2), and Wilt(4839, 5, 2), where the numbers in each parenthesis are the number of instances, features, and classes.

We have tested all the algorithms on the 15 benchmark datasets. Twenty runs evaluate the accuracy of these algorithms. At each run, every dataset is randomly partitioned into two parts with 70% as the training set and 30% as the testing set. We have tuned the parameters for each run to obtain optimal results. In the MML-LMF[1] algorithm, we set $\lambda_1 = 1$ and ignore the global regularization term $\|L_0\|_F^2$, since the local metric learning tasks can also serve the purpose of the regularization term. Furthermore, we neglect the fusion regularization term and set $\lambda_3 = 1$ as we want local metrics to fuse slowly. The process of metric fusion makes a trade-off between discriminative ability and performance. We mainly adjust the parameters in the selection method for constraints and the parameter λ_2. In the implementation, we classify the parameters of the selection method for constraints into 9 pairs according to the number of similar nearest neighbors k_1 and the number of dissimilar nearest neighbors k_2, i.e., $(k_1, k_2) : \{(1,1),(1,2),(1,3),\cdots,(3,3)\}$. The parameter λ_2 controls the sparsity of local metrics and is tuned from $\{0.001, 0.01, 0.1, 1\}$.

When calculating the metric corresponding to each constraint, each instance corresponds to multiple metrics and constraints. Let us denote the metrics set as $\mathscr{L}_{x_i}$ and the constraints set as $\mathscr{T}_{x_i}$ for x_i. We evaluate the metrics for x_i by selecting one metric L from $\mathscr{L}_{x_i}$ so that the loss of the $k_1 + k_2$ constraints is minimized, i.e.

$$\min_{L \in \mathscr{L}_{x_i}} \sum_{t=1}^{|\mathscr{T}_{x_i}|} l(q_t(\gamma - \|L v_t\|_2^2)), \tag{11}$$

where $l(x) = \max\{1, -x\}$. The distance between a testing instance x_t and a training instance x_i is denoted as $d_L^2(x_i, x_t)$. We evaluate the performance of the proposed algorithm with the 3NN classifier.

We report the experimental results on classification accuracy in Table 1 and mark the highest accuracy for each dataset in bold. Each item in the table represents the mean of 20 runs, with the variance of 20 runs in parentheses. The bottom row shows the average rank order for each algorithm. The compared algorithms have been optimized by tuning the parameters according to their corresponding literature. Our algorithm performs much better on benchmark datasets than those representative algorithms. From the average rank order of the LIFT and the MML-LMF, one can see that the MML-LMF framework achieves better trade-offs between discriminative ability and performance than the LIFT framework does.

4.1. Fusion Regularization Term in MML-LMF

In the MML-LMF, we fuse some similar metrics into one metric to reduce its complexity. Thus we show the accuracy change of the 3NN classifier in the process of merging metrics to illustrate the effect of metrics fusion on the MML-LMF. Regarding random partitions and simplicity of display, we randomly select 5 of the 20 runs and choose the parameters corresponding to the optimal results. In Fig. 1, we illustrate the effect of the number of metrics on the classification performance for four datasets, which shows that the partition of data significantly affects the number of optimal metrics. Moreover, the given number of metrics also affects the performance of MML-LMF. The results above further validates that MML-LMF can provide a better trade-off between discriminative ability and performance.

4.2. Visualization

Here we use tSNE [26,27] to visualize the results of MML-LMF, plotted in Figs. 2(a)-(d). The Figs. 2(a)-(d) show the projection visualization effects on Balance and Chess, respectively. When learning each metric for the selected constraints, MML-LMF picks the metric L_i for each instance x_i with (11). The distance between two instances in the training set is $d(x_i, x_j) = \|L_i x_i - L_j x_j\|_2^2$. The MML-LMF is equivalent to the Euclidean distance after different linear transformations between two instances. The original instance class membership is shown in Figs. 2(a), (c). The instance after trained metrics is shown in Figs. 2(b), (d). One can see that most of the similar instances are clustered compared with the original instance class membership.

4.3. Parameter Sensitivity

In the MML-LMF, the main factors affecting the model metric are the selection method of constraints and the local metric regularizer's coefficient λ_2. We have used only a simple constraint selection method, that is to say, we select only the k_1 similar and k_2 dissimilar nearest neighbors of the instance to construct the pairwise constraints, where k_1 and k_2 are selected in $\{1, 2, 3\}$. Furthermore, the parameter λ_2 controls the number of non-zero local metric biases. When λ_2 becomes large enough, all local metric biases are zero. λ_2 is selected in $\{0.001, 0.01, 0.1, 1\}$. We show the effect of parameters (k_1, k_2) and parameter λ_2 on the classification accuracy for 12 datasets in Fig. 3.

The experimental results show that the selection method of constraints is remarkably effective. At the same time, the regularization term of local metric biases also significantly affects the performance of MML-LMF in most datasets. From Figs. 3 (b), (g), and (h), one can see that the more the constraints are selected, the more the performance may be reduced instead. Therefore, the selection method of constraints is the core issue in MML-LMF. Moreover, the experimental results in Figs. 3(a),

[1] https://github.com/array12138/MML-LMF/tree/main

Table 1
Classification accuracy of classical global and multiple metric learning algorithms on different data sets.

Methods	KNN	NCA	LMNN	GMML	GNSML	ANML	MMLMNN	SCML	CMML	LIFT	MML-LMF
Auto	.550(.060)	.573(.071)	.583(.047)	.552(.057)	.584(.071)	.656(.041)	.557(.245)	.601(.054)	.573(.043)	.592(.044)	**.683**(.065)
Balance	.806(.027)	.936(.027)	.847(.026)	.924(.017)	.922(.017)	**.944**(.007)	.896(.017)	.903(.020)	.927(.015)	.922(.018)	.939(.022)
Breast	.956(.009)	.951(.015)	.963(.011)	.967(.011)	.961(.009)	.961(.008)	.963(.011)	.952(.005)	.964(.009)	.964(.012)	**.971**(.011)
Cars	.831(.031)	**.925**(.040)	.880(.028)	.845(.032)	.878(.030)	.871(.035)	.904(.022)	.923(.014)	.884(.034)	.874(.027)	.892(.020)
Chess	.783(.214)	.982(.006)	.974(.007)	.937(.009)	.954(.007)	**.985**(.002)	.979(.005)	.971(.003)	.980(.006)	.980(.005)	**.985**(.004)
Cleve	.729(.031)	.707(.043)	.743(.050)	.752(.031)	.753(.037)	**.785**(.031)	.730(.031)	.678(.052)	.740(.029)	.712(.044)	.768(.031)
Glass	.659(.058)	.658(.082)	.689(.056)	.687(.039)	.702(.053)	**.769**(.040)	.720(.045)	.655(.035)	.679(.039)	.673(.062)	.733(.051)
Heart	.816(.037)	.797(.044)	.825(.027)	.834(.028)	.830(.030)	.840(.040)	.807(.039)	.803(.028)	.836(.040)	.840(.043)	**.864**(.034)
ILPD	.658(.035)	.674(.040)	.661(.031)	.693(.033)	.677(.037)	.693(.043)	.664(.028)	.682(.015)	.692(.026)	.717(.019)	**.728**(.023)
Letter	.954(.000)	.960(.032)	.969(.000)	.951(.000)	.972(.000)	.971(.000)	.965(.000)	.970(.000)	.969(.000)	.977(.000)	**.982**(.000)
Pima	.727(.031)	.727(.026)	.730(.025)	.738(.020)	.752(.026)	.753(.016)	.734(.023)	.713(.016)	.750(.021)	**.765**(.018)	**.766**(.022)
Segment	.949(.009)	.957(.010)	.962(.006)	.948(.009)	.958(.006)	**.971**(.005)	.967(.008)	.969(.004)	.955(.007)	.966(.008)	**.971**(.005)
Solar	.610(.044)	.607(.047)	.642(.052)	.622(.034)	.632(.045)	.635(.034)	.645(.040)	**.676**(.039)	.639(.038)	.649(.038)	.665(.051)
Vote	.917(.021)	.939(.024)	.945(.016)	.929(.019)	.947(.014)	.957(.010)	.956(.014)	.921(.009)	.953(.015)	**.960**(.014)	**.959**(.017)
Wilt	.696(.000)	.853(.023)	.842(.000)	.722(.000)	.802(.000)	.838(.000)	**.862**(.000)	.854(.000)	.798(.000)	.806(.000)	**.862**(.000)
Avg Rank	10.00	7.53	6.53	7.4	6.00	3.53	5.93	6.80	5.80	4.67	**1.80**

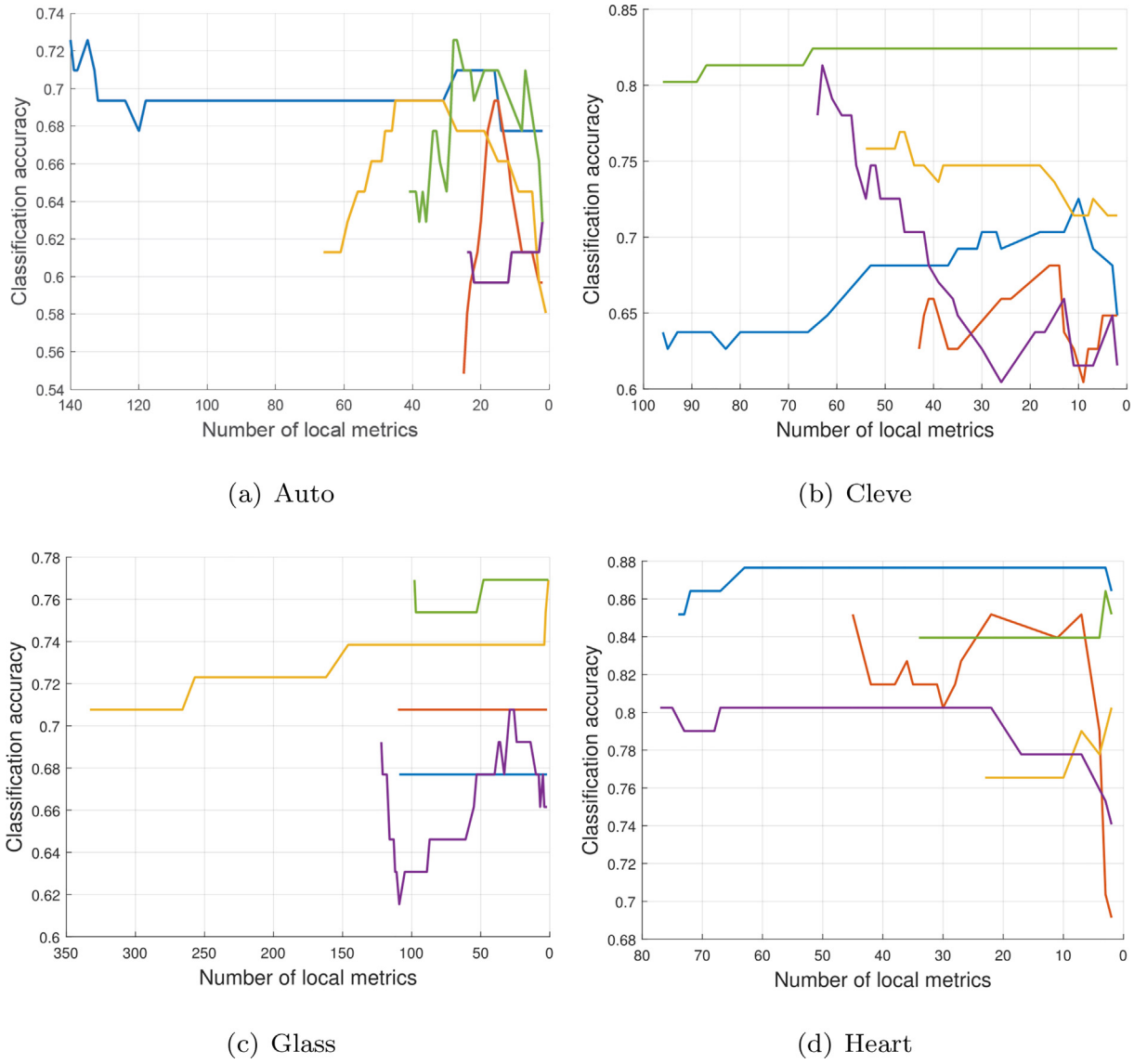

Fig. 1. Change of the classification performance and the number of local metrics. Different colors indicate that different training sets are used.

(b), and (g) show that different datasets have different sensitivities to the sparsity of local biases, so the optimal values should be obtained at different λ_2 for the datasets. Hence it is also crucial to adjust λ_2 to control the sparsity of the local metric biases.

4.4. Experiments on Person Re-identification

We have also evaluated our algorithm on two benchmark person re-identification datasets: VIPeR [28] and PRID450S [29]. The VIPeR dataset contains individuals, each of which has two images. Each group of images is taken from a horizontal view but with very different orientations. There are 450 image pairs in the PRID450S dataset, which are collected from two different static surveillance cameras. Fig. 4 gives some example images of each of these datasets.

Since the acquisition of natural images is easily affected by viewpoints, pose, illumination, and occlusion, we use a novel descriptor method [30,31] based on a hierarchical distribution of pixel features. The method can effectively avoid the negative influence of a complex environment, while the extracted data features are publicly available [2].

[2] http://www.i.kyushu-u.ac.jp/ matsukawa/ReID.html

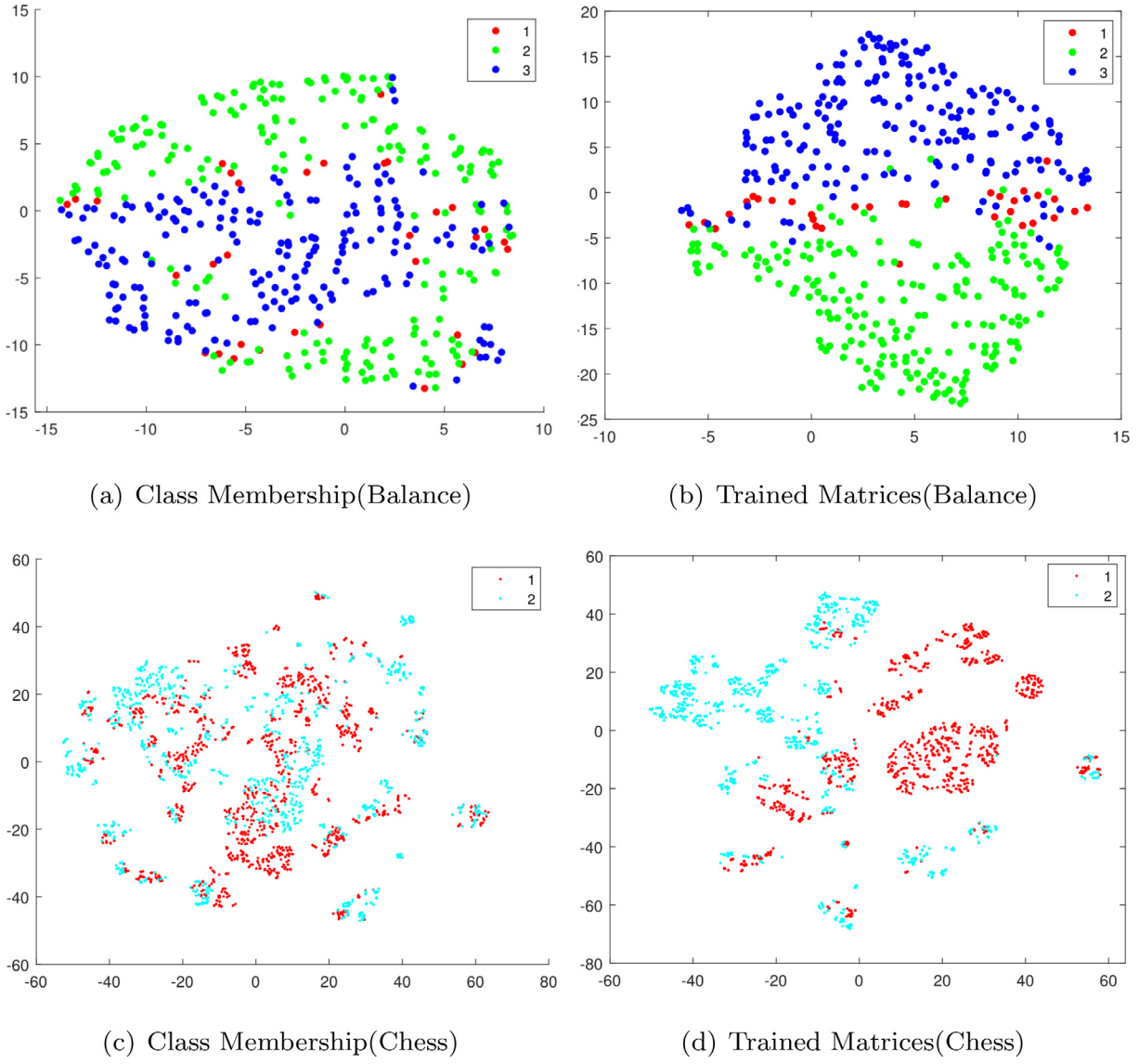

Fig. 2. Visualization effects on different datasets.

We carry out all evaluations in the single-shot experiment settings [32,33]. Firstly, we divide the individuals in datasets into two subsets, randomly selecting p individuals as the testing set and the remaining as the training set, where p is chosen to be 316, 225 for VIPeR, PRID450S, respectively. Then we reduce the dimensionality of the original datasets to 50 dimensions by using PCA. Following the same settings in [33], we randomly generate 10 groups of partitions. In each partition, one image of each person is randomly chosen as a probe image, and the rest are regarded as gallery images.

In addition, we set $\gamma = 4, k_1 = 1$, and the other parameters are the same as the ones in the experiments conducted on the Benchmark datasets. When calculating the distance between two instances according to multiple metrics, we follow the principle that the distance is $d_{M_{x_i}}(x_i, x_j)$, where M_{x_i} is the metric corresponding to x_i if we calculate how close x_j is to x_i. Following this principle, we evaluate the performance with the Cumulative Matching Characteristic (CMC) curves, which visualize the expectation of finding the correct person in the top 30 matches. We calculate the average results of 10 runs and show the experimental results in Fig. 5. From the result, one can see that our algorithm has a clear advantage on VIPeR and PRID450S over the compared algorithms.

4.5. Experiments on Face Verification

The face verification task aims to determine whether two face images belong to the same face. In this section, we carry out the MML-LMF algorithm on the LFW (Labeled Faces in the Wild) dataset for the face verification task. The algorithm's

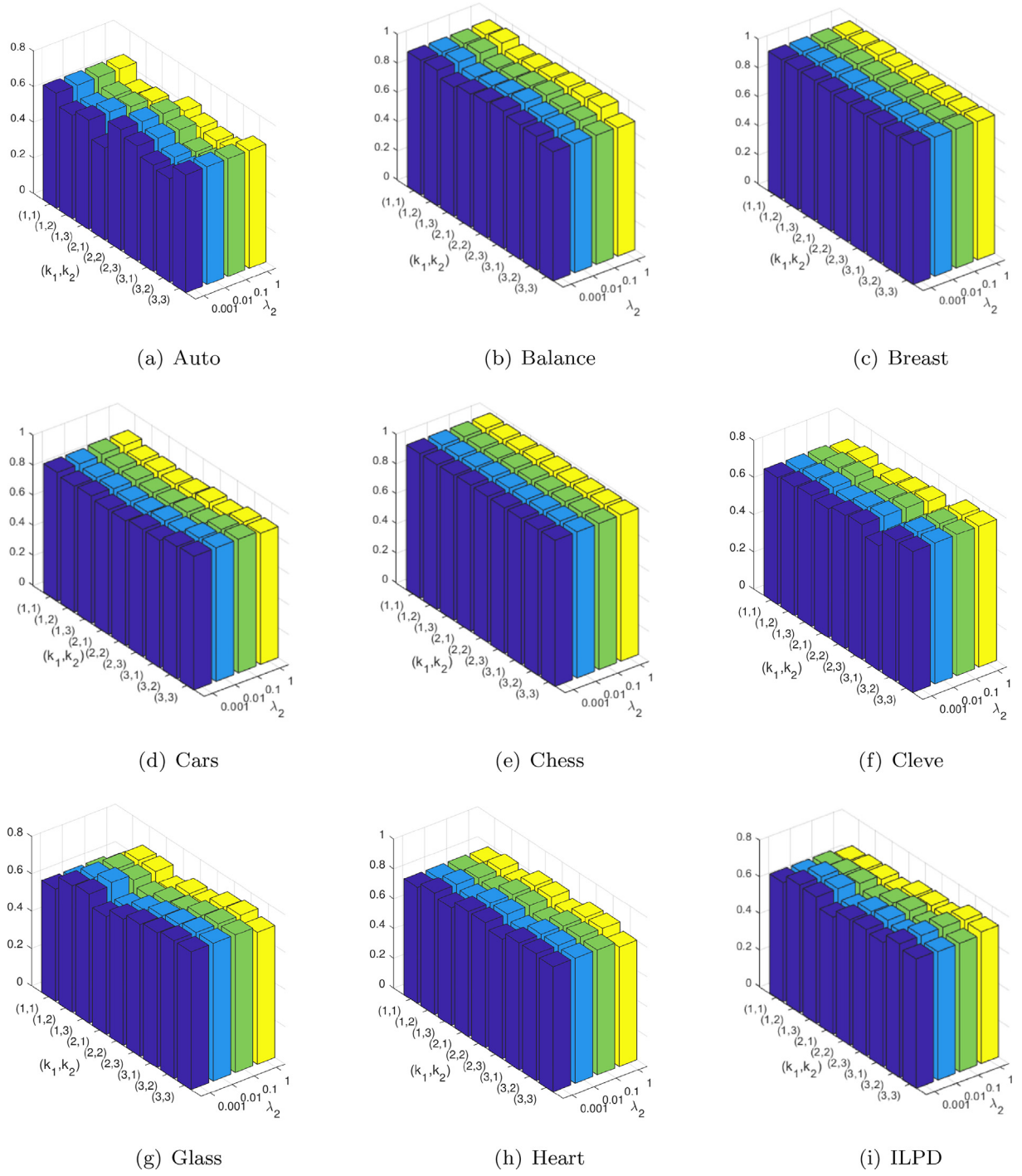

Fig. 3. Parameter Sensitivity on different datasets.

performance is evaluated with accuracy. Experiments show that the MML-LMF outperforms the state-of-the-art metric learning algorithms.

Our experiments first use a lightweight Python-based framework, deepface [3] to extract features. The deepface encapsulates a variety of classical face recognition networks, such as DeepFace [34], VGG-Face[35], Facenet [36], and OpenFace [37]. Then the

[3] https://github.com/serengil/deepface

(a) VIPeR (b) PRID450S

Fig. 4. Example images from the person re-identification datasets. For each dataset, images in the same column represent the same person.

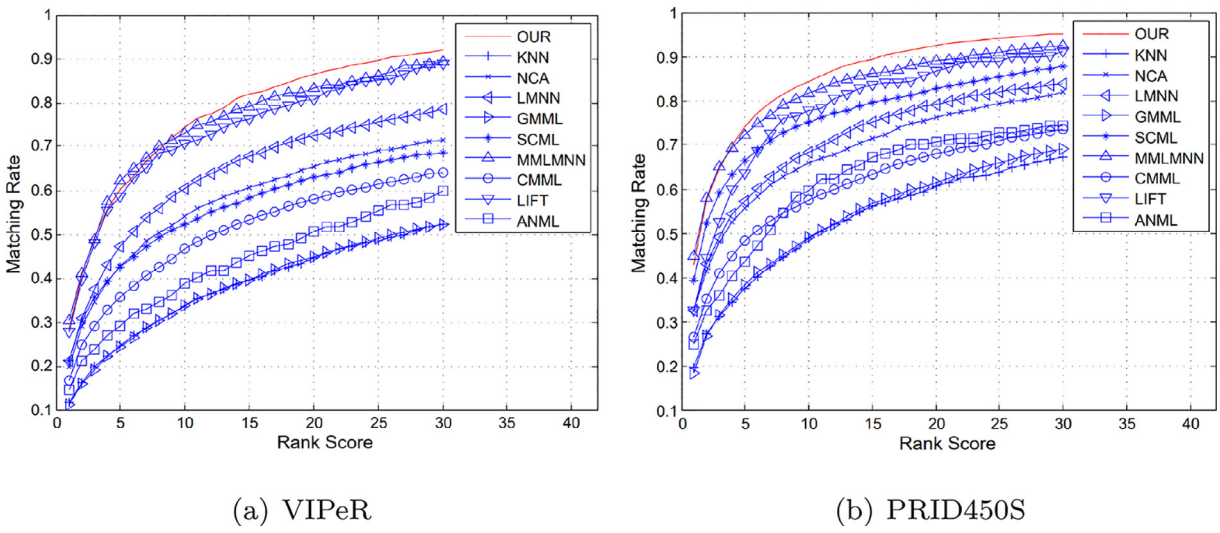

Fig. 5. The CMC curves of different algorithms on different datasets.

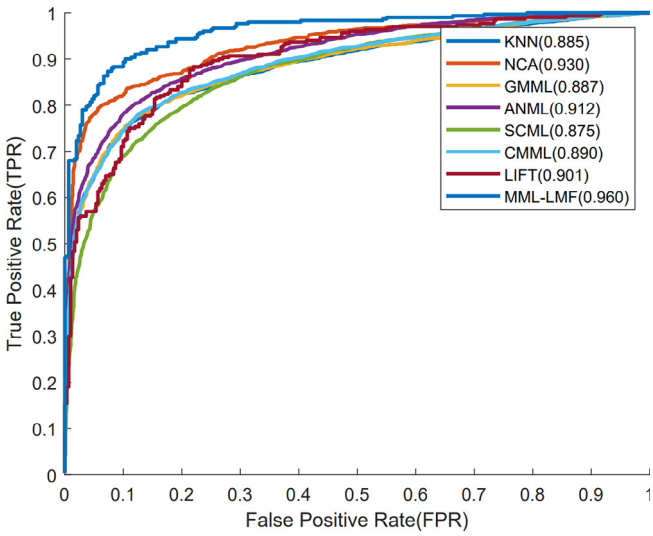

Fig. 6. ROC curves of all algorithms on LFW dataset. AUC values are presented in the legends.

VGG-Face feature is used to evaluate our algorithm, where the VGG-Face feature has 2622 dimensions. Subsequently, we use PCA to reduce the VGG-Face feature to 50.

The MML-LMF is evaluated with a ten-fold cross-validation approach, which is similar to the experimental setup in [25,38,14]. Each fold consists of 300 similar face pairs and 300 dissimilar face pairs. We choose one of the folds as the testing set and the rest as the training set. For SCML, CMML, and LIFT, we set the target nearest neighbor to 1 neighbor while using the face images in the training face pairs as the training set. Then the test instance selects the nearest training instance's metric as its metric. We plot the ROC curve by changing the thresholds of different distance metrics. Then the values of Area Under Curve (AUC) are calculated to quantitatively evaluate the performances of all comparators. We report the ROC curves and AUC values of KNN, NCA, GMML, ANML, SCML, CMML, LIFT, and MML-LMF in Fig. 6, where one can see that the proposed MML-LMF algorithm can achieve satisfactory verification accuracy, higher to the competing algorithms. Among them, the performance of SCML, CMML, and LIFT is relatively lower. Since these algorithms perform single metric learning on a cluster of data, the differences in faces in clusters could be huge, and it is difficult to capture the cluster structure with a single metric.

5. Conclusion

We have proposed a framework of multi-metric learning via local metric fusion, which has a significant advantage over the instance cluster learning metrics approaches in the literature. The MML-LMF can adaptively determine the number of local metrics before learning the local metrics. As a result of this framework, we have constructed a concrete optimization model with pairwise constraints and acquired a closed-form solution to the model. The experimental results show that MML-LMF is effective and performs better than the state-of-the-art multiple metric learning algorithms. In future work, we will extend the MML-LMF framework to triple constraints and develop a better constraint selection method.

CRediT authorship contribution statement

Xinyao Guo: Conceptualization, Methodology, Software, Visualization, Writing - original draft. **Lin Li:** Project administration, Writing - review & editing. **Chuangyin Dang:** Investigation, Methodology. **Jiye Liang:** Supervision, Methodology. **Wei Wei:** Validation, Writing - review & editing, Software.

Data availability

All data are public data sets, which are easy to retrieve on the network. Meanwhile, the links are also attached, specifically: https://github.com/array12138/MML-LMF/tree/main

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work was partially supported by the National Natural Science Foundation of China (Nos. 61976184, 62006147, U21A20473), and the 1331 Engineering Project of Shanxi Province, China.

References

[1] W. Yan, Q. Sun, H. Sun, Y. Li, Joint dimensionality reduction and metric learning for image set classification, Information Sciences 516 (2020) 109–124.
[2] Q. Wang, W. Min, Q. Han, Z. Yang, X. Xiong, M. Zhu, H. Zhao, Viewpoint adaptation learning with cross-view distance metric for robust vehicle re-identification, Information Sciences 564 (2021) 71–84.
[3] G. Andresini, A. Appice, D. Malerba, Autoencoder-based deep metric learning for network intrusion detection, Information Sciences 569 (2021) 706–727.
[4] M. Kemertas, T. Aumentado-Armstrong, Towards robust bisimulation metric learning, Advances in Neural Information Processing Systems 34 (2021) 4764–4777.
[5] Z. Hu, D. Wu, F. Nie, R. Wang, Generalization bottleneck in deep metric learning, Information Sciences 581 (2021) 249–261.
[6] D. Wu, H. Wang, Z. Hu, F. Nie, Improved deep metric learning with local neighborhood component analysis, Information Sciences 617 (2022) 165–176.
[7] H.J. Ye, D.C. Zhan, X.-M. Si, Y. Jiang, Learning feature aware metric, in: Asian Conference on Machine Learning, 2016, pp. 286–301.
[8] S. Ying, Z. Wen, J. Shi, Y. Peng, J. Peng, H. Qiao, Manifold preserving: An intrinsic approach for semisupervised distance metric learning, IEEE Transactions on Neural Networks and Learning Systems 29 (7) (2017) 2731–2742.
[9] H.J. Ye, D.C. Zhan, Y. Jiang, et al, What makes objects similar: A unified multi-metric learning approach, IEEE Transactions on Pattern Analysis and Machine Intelligence 41 (5) (2018) 1257–1270.
[10] J. Goldberger, G.E. Hinton, S.T. Roweis, et al, Neighbourhood components analysis, Advances in Neural Information Processing Systems (2005) 513–520.
[11] K.Q. Weinberger, L.K. Saul, Distance metric learning for large margin nearest neighbor classification, Journal of Machine Learning Research 10 (2) (2009) 207–244.
[12] P. Zadeh, R. Hosseini, S. Sra, Geometric mean metric learning, in: International Conference on Machine Learning, 2016, pp. 2464–2471.

[13] Y. Peng, L. Hu, S. Ying, st al., Global nonlinear metric learning by gluing local linear metrics, in: Proceedings of the 2018 SIAM International Conference on Data Mining, 2018, pp. 423–431.
[14] S. Chen, L. Luo, J. Yang, et al, Curvilinear distance metric learning, in: International Conference on Neural Information Processing Systems, 2019, pp. 4223–4232.
[15] Y. Ruan, Y. Xiao, Z. Hao, et al, A nearest-neighbor search model for distance metric learning, Information Sciences 552 (2021) 261–277.
[16] X. Deng, Z. Zhang, Deep causal metric learning, in: Proceedings of the 39th International Conference on Machine Learning, Vol. 162, 2022, pp. 4993–5006.
[17] X. Li, Y. Bai, Y. Peng, et al, Nonlinear semi-supervised metric learning via multiple kernels and local topology, International Journal of Neural Systems 28 (02) (2018) 1750040.
[18] B. Nguyen, F.J. Ferri, C. Morell, et al, An efficient method for clustered multi-metric learning, Information Sciences 471 (2019) 149–163.
[19] E. Fetaya, S. Ullman, Learning local invariant mahalanobis distances, in: International Conference on Machine Learning, 2015, pp. 162–168.
[20] D.C. Zhan, M. Li, Y.F. Li, et al., Learning instance specific distances using metric propagation, in: International Conference on Machine Learning, 2009, pp. 1225–1232.
[21] H.J. Ye, D.C. Zhan, N. Li, et al, Learning multiple local metrics: Global consideration helps, IEEE transactions on Pattern Analysis and Machine Intelligence 42 (7) (2019) 1698–1712.
[22] J. Wang, A. Kalousis, A. Woznica, Parametric local metric learning for nearest neighbor classification, Advances in Neural Information Processing Systems (2012) 1601–1609.
[23] Y. Shi, A. Bellet, F. Sha, Sparse compositional metric learning, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 28, 2014, pp. 2078–2084.
[24] S.C. Hoi, W. Liu, S.-F. Chang, Semi-supervised distance metric learning for collaborative image retrieval and clustering, ACM Transactions on Multimedia Computing, Communications, and Applications 6 (3) (2010) 1–26.
[25] W. Zuo, F. Wang, D. Zhang, et al, Distance metric learning via iterated support vector machines, IEEE Transactions on Image Processing 26 (10) (2017) 4937–4950.
[26] L. Van der Maaten, G. Hinton, Visualizing data using t-sne, Journal of Machine Learning Research 9 (11) (2008) 2579–2605.
[27] L. Van Der Maaten, Accelerating t-sne using tree-based algorithms, The Journal of Machine Learning Research 15 (1) (2014) 3221–3245.
[28] D. Gray, H. Tao, Viewpoint invariant pedestrian recognition with an ensemble of localized features, in: European Conference on Computer Vision, 2008, pp. 262–275.
[29] C.C. Loy, T. Xiang, S. Gong, Time-delayed correlation analysis for multi-camera activity understanding, International Journal of Computer Vision 90 (1) (2010) 106–129.
[30] T. Matsukawa, T. Okabe, E. Suzuki, et al, Hierarchical gaussian descriptor for person re-identification, Computer Vision and Pattern Recognition (2016) 1363–1372.
[31] T. Matsukawa, T. Okabe, E. Suzuki, et al, Hierarchical gaussian descriptors with application to person re-identification, IEEE transactions on Pattern Analysis and Machine Intelligence 42 (9) (2019) 2179–2194.
[32] S. Gong, M. Cristani, C.C. Loy, et al, The re-identification challenge, Person Re-identification (2014) 1–20.
[33] C. Jose, F. Fleuret, Scalable metric learning via weighted approximate rank component analysis, in: European Conference on Computer Vision, 2016, pp. 875–890.
[34] Y. Taigman, M. Yang, M. Ranzato, et al, Deepface: Closing the gap to human-level performance in face verification, in: IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 1701–1708.
[35] O.M. Parkhi, A. Vedaldi, A. Zisserman, Deep face recognition, British Machine Vision Conference (2015) 1–12.
[36] F. Schroff, D. Kalenichenko, J. Philbin, Facenet: A unified embedding for face recognition and clustering, in: IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 815–823.
[37] T. Baltrusaitis, A. Zadeh, Y.C. Lim, et al, Openface 2.0: Facial behavior analysis toolkit, in: IEEE International Conference on Automatic Face and Gesture Recognition (FG 2018), 2018, pp. 59–66.
[38] Z. Huo, F. Nie, H. Huang, Robust and effective metric learning using capped trace norm: Metric learning via capped trace norm, in: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 1605–1614.