Discriminative Deep Metric Learning for Face and Kinship Verification

Jiwen Lu, *Senior Member, IEEE*, Junlin Hu, and Yap-Peng Tan, *Senior Member, IEEE*

***Abstract*— This paper presents a new discriminative deep metric learning (DDML) method for face and kinship verification in wild conditions. While metric learning has achieved reasonably good performance in face and kinship verification, most existing metric learning methods aim to learn a single Mahalanobis distance metric to maximize the inter-class variations and minimize the intra-class variations, which cannot capture the nonlinear manifold where face images usually lie on. To address this, we propose a DDML method to train a deep neural network to learn a set of hierarchical nonlinear transformations to project face pairs into the same latent feature space, under which the distance of each positive pair is reduced and that of each negative pair is enlarged. To better use the commonality of multiple feature descriptors to make all the features more robust for face and kinship verification, we develop a discriminative deep multi-metric learning method to jointly learn multiple neural networks, under which the correlation of different features of each sample is maximized, and the distance of each positive pair is reduced and that of each negative pair is enlarged. Extensive experimental results show that our proposed methods achieve the acceptable results in both face and kinship verification.**

***Index Terms*— Face verification, kinship verification, deep learning, deep metric learning, multi-feature learning.**

I. Introduction

A NUMBER of face recognition methods have been proposed in recent years [1], [2], and most of them have achieved encouraging performance under controlled conditions. However, their performance drops heavily when face images were captured in the wild because large intra-class variations usually occur in this scenario. Face recognition can be mainly classified into two categories: face identification and face verification. The first aims to recognize the person from a set of gallery face samples to find the most matched one to the probe sample. The second is to determine whether a given pair of face samples is from the same person or not. In this paper, we focus on the second one where face samples were acquired in unconstrained environments and a variety of variations on lighting, expression, pose, resolution, and background are contained in the captured face samples.

Recently, kinship verification via face images has been an emerging problem in face analysis [3]–[6]. Unlike face verification which aims to verify whether a pair of face samples is from the same person or not, the objective of kinship verification is to determine whether there is a kin relation between two persons from their faces. Generally, the kin is defined as a relation between two persons who are biologically related with overlapping genes. Hence, there are four representative kin relations in kinship verification: father-son (F-S), father-daughter (F-D), mother-son (M-S) and mother-daughter (M-D). In this work, we focus on verifying these four kin relations from facial images which are captured in wild conditions.

In this paper, we propose a new discriminative deep metric learning (DDML) method for face and kinship verification. Fig. 1 illustrates the basic idea of our proposed method. Unlike existing metric learning methods, our DDML builds a deep neural network to learn a set of hierarchical nonlinear transformations to map face samples into discriminative feature spaces, under which the distance of each positive pair is reduced and that of each negative pair is enlarged, respectively. To better use multiple feature descriptors for face and kinship verification, we further develop a discriminative deep multi-metric learning (DDMML) method to jointly learn multiple neural networks to extract the common information. Experimental results on multiple face datasets show the effectiveness of the proposed methods.

The contributions of this paper are summarized as:

- We propose a discriminative deep metric learning (DDML) method for face and kinship verification. Unlike most existing metric learning methods which learn a linear transformation, DDML learns a set of hierarchical nonlinear transformations by a neural network to project face images into discriminative feature subspaces, so that both the nonlinear and scalability problems can be explicitly simultaneously addressed.
- We develop a discriminative deep multi-metric learning (DDMML) method to jointly learn multiple neural networks to exploit the common information to improve the verification performance. Unlike existing multi-metric learning methods which usually linearly combine several Mahalanobis distance metrics, DDMML collaboratively learns multiple neural networks so that the common and discriminative information can be extracted to make all the features more robust.

Manuscript received June 16, 2015; revised February 8, 2016 and December 26, 2016; accepted June 15, 2017. Date of publication June 20, 2017; date of current version July 6, 2017. This work was supported in part by the National Natural Science Foundation of China under Grant 61672306 and in part by the National 1000 Young Talents Plan Program. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Shiguang Shan. *(Corresponding author: Jiwen Lu.)*

J. Lu is with the Department of Automation, Tsinghua University, State Key Laboratory of Intelligent Technologies and Systems, and Tsinghua National Laboratory for Information Science and Technology (TNList), Beijing, 100084, China (e-mail: lujiwen@tsinghua.edu.cn).

J. Hu and Y.-P. Tan are with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798 (e-mail: jhu007@e.ntu.edu.sg; eyptan@ntu.edu.sg).

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

Digital Object Identifier 10.1109/TIP.2017.2717505

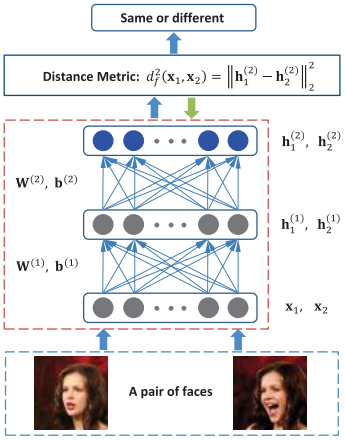


Fig. 1. The flowchart of DDML method for face verification. Given a pair of face images $\mathbf{x}_1$ and $\mathbf{x}_2$, we map them into the same feature space $\mathbf{h}_1^{(2)}$ and $\mathbf{h}_2^{(2)}$ by learning a set of hierarchical nonlinear transformations, where the similarity between their outputs at the top level of the network is computed to determine whether the pair is from the same person or not.

- We conduct extensive face and kinship verification experiments to demonstrate the efficacy of our methods.

This paper is an extended version of our previous conference work [7]. There are the following key extensions:

- We have extended discriminative deep metric learning (DDML) into discriminative deep multi-metric learning (DDMML) to jointly learn multiple neural networks to better combine multiple feature descriptors to improve the verification performance.
- Besides face verification, we have applied our proposed DDML and DDMML methods to kinship verification via face analysis. We conducted experiments on three widely used kinship face datasets.
- We have conducted more experimental results to evaluate the performance of our proposed methods including 1) more results on the LFW dataset under different settings, 2) more analysis of the proposed approaches, and 3) extensive comparisons with the state-of-the-art face and kinship verification methods.

II. Related Work

We briefly review four related topics: 1) face verification, 2) kinship verification, 3) metric learning, and 4) deep learning.

A. Face Verification

A variety of unconstrained face verification methods have been proposed in recent years [6], [8]–[18], and they can be mainly classified into two categories: appearance-based and geometry-based. Since geometric features usually ignore some discriminative information of human faces, appearance-based methods are more popular in unconstrained face verification. There are two key components in appearance-based face verification: feature representation and similarity measure. For feature representation, a robust hand-crafted or learned descriptor is usually employed, so that the inter-class variations are enlarged and intra-class variations are reduced. For similarity measure, an effective distance metric is usually learned from the labeled training samples, under which the similarity of positive pairs is enlarged and that of negative pairs is reduced as much as possible. Typical feature descriptors include scale-invariant feature transform (SIFT) [19], local binary pattern (LBP) [1], fisher vector faces [15], spatial face region descriptor (SFRD) [13] and discriminant face descriptor (DFD) [20]. And representative similarity measure methods include logistic discriminant metric learning (LDML) [10], cosine similarity metric learning (CSML) [21], and pairwise-constrained multiple metric learning (PMML) [13]. In this work, we contribute to the similarly measure component by presenting a new deep metric learning approach.

B. Kinship Verification

There have been a few seminal studies in kinship verification in recent years [3], [6], [22]–[25], and these methods can be mainly categorized into two classes: feature-based [3], [22], [23], [25] and model-based [6], [26]. Generally, feature-based methods aim to extract discriminative information to preserve stable kin-related characteristics. Representative methods in this category include skin color [3], histogram of gradient [3], [22], [24], Gabor wavelet [23], [24], gradient orientation pyramid [23], salient part [27], self-similarity [28], and dynamic spatio-temporal descriptor [25]. Model-based methods apply machine learning techniques to learn an effective classifier, such as metric learning [6], multiple kernel learning [23] and graph-based fusion [26].

C. Metric Learning

Many metric learning algorithms have been proposed over the past decade [4], [6], [10], [11], [13], [21], [29]–[32]. The common objective of these methods is to learn a good distance metric so that the similarity of each positive pair is enlarged and that of each negative pair is reduced. State-of-the-art metric learning methods include large margin nearest neighbor (LMNN) [33], information theoretic metric learning (ITML) [29], KISS metric embedding (KISSME) [30], and pairwise constrained component analysis (PCCA) [31]. However, these methods only learn a linear transformation to map samples into another feature space, which may not be powerful enough to capture the nonlinear manifold. To address this limitation, the kernel trick is usually adopted to map samples into a high-dimensional feature space under which a discriminative distance metric is learned [34]. However, these methods cannot explicitly obtain the nonlinear mapping functions, which usually suffer from the scalability problem. Unlike these methods, in this work, we propose a deep metric learning approach to learn hierarchical nonlinear mappings to address both the nonlinear and scalability problems simultaneously.

D. Deep Learning

Deep learning has received increasing interests in computer vision and machine learning in recent years, and a number

of such methods have been proposed in the literature [12], [35]–[42]. Existing deep learning methods can be mainly categorized three classes: unsupervised, supervised and semi-supervised. Representative deep learning models included deep stacked auto-encoder [40], [43], [44], deep convolutional neural networks (DCNN) [42], [45], [46], and deep belief network [36]. Among these methods, DCNN has achieved exciting performance in many computer vision applications such as image classification [41], human action recognition [42], and face verification [47]. While many attempts have been made on deep learning in feature engineering, little progress has been made in metric learning with a deep architecture.

III. The Proposed Methods

We first briefly review conventional Mahalanobis distance metric learning, and then present the proposed DDML and DDMML methods, as well as their implementation details.

A. Mahalanobis Distance Metric Learning

Let $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N] \in \mathbb{R}^{d\times N}$ be the training set, where $\mathbf{x}_i \in \mathbb{R}^d$ is the ith training sample and N is the total number of training samples, $1 \le i \le N$. The conventional Mahalanobis distance metric learning aims to seek a square matrix $\mathbf{M} \in \mathbb{R}^{d\times d}$ from the training set $\mathbf{X}$, under which the distance between any two samples $\mathbf{x}_i$ and $\mathbf{x}_j$ is computed as:

$$d_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)^T \mathbf{M}(\mathbf{x}_i - \mathbf{x}_j)}. \tag{1}$$

Since $d_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_j)$ is a distance, it is nonnegative, symmetrical and satisfies the triangle inequality. Hence, $\mathbf{M}$ can be decomposed by as:

$$\mathbf{M} = \mathbf{W}^T\mathbf{W}, \tag{2}$$

where $\mathbf{W} \in \mathbb{R}^{p\times d}$, and $p \le d$.

Then, $d_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_j)$ can be rewritten as

$$\begin{aligned} d_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_j) &= \sqrt{(\mathbf{x}_i - \mathbf{x}_j)^T \mathbf{M}(\mathbf{x}_i - \mathbf{x}_j)} \\ &= \sqrt{(\mathbf{x}_i - \mathbf{x}_j)^T \mathbf{W}^T\mathbf{W}(\mathbf{x}_i - \mathbf{x}_j)} \\ &= \|\mathbf{W}\mathbf{x}_i - \mathbf{W}\mathbf{x}_j\|_2. \end{aligned} \tag{3}$$

We see from (3) that learning a Mahalanobis distance metric $\mathbf{M}$ is equivalent to seeking a linear transformation $\mathbf{W}$ which projects each sample $\mathbf{x}_i$ into a low-dimensional subspace, under which the Euclidean distance of two samples in the transformed space is equal to the Mahalanobis distance in the original space.

B. DDML

As shown in Fig. 1, we first construct a deep neural network to compute the representations of a face pair by passing them to multiple stacked layers of nonlinear transformations. Assume there are $M+1$ layers in our designed network, and $p^{(m)}$ units in the mth layer, where $m = 1, 2, \cdots, M$. For a given face sample $\mathbf{x} \in \mathbb{R}^d$, the output at the first layer is $\mathbf{h}^{(1)} = s(\mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)}) \in \mathbb{R}^{p^{(1)}}$, where $\mathbf{W}^{(1)} \in \mathbb{R}^{p^{(1)}\times d}$ is a projection matrix to be learned in the first layer, $\mathbf{b}^{(1)} \in \mathbb{R}^{p^{(1)}}$ is a bias vector, and $s : \mathbb{R}^d \mapsto \mathbb{R}^d$ is a nonlinear activation function which operates component-wisely. Representative such functions include the *tanh* or *sigmoid* function. Then, we use the output of the first layer $\mathbf{h}^{(1)}$ as the input of the second layer. Similarly, the output of the second layer can be computed as $\mathbf{h}^{(2)} = s(\mathbf{W}^{(2)}\mathbf{h}^{(1)} + \mathbf{b}^{(2)}) \in \mathbb{R}^{p^{(2)}}$, where $\mathbf{W}^{(2)} \in \mathbb{R}^{p^{(2)}\times p^{(1)}}$, $\mathbf{b}^{(2)} \in \mathbb{R}^{p^{(2)}}$, and s are the projection matrix, bias, and nonlinear activation function of the second layer, respectively. The output of the mth layer is $\mathbf{h}^{(m)} = s(\mathbf{W}^{(m)}\mathbf{h}^{(m-1)} + \mathbf{b}^{(m)}) \in \mathbb{R}^{p^{(m)}}$, and the output of the most top level can be computed as:

$$f(\mathbf{x}) = \mathbf{h}^{(M)} = s\big(\mathbf{W}^{(M)}\mathbf{h}^{(M-1)} + \mathbf{b}^{(M)}\big) \in \mathbb{R}^{p^{(M)}}, \tag{4}$$

where the mapping $f : \mathbb{R}^d \mapsto \mathbb{R}^{p^{(M)}}$ is a parametric nonlinear function determined by the parameters $\mathbf{W}^{(m)}$ and $\mathbf{b}^{(m)}$, where $m = 1, 2, \cdots, M$.

Given a pair of face samples $\mathbf{x}_i$ and $\mathbf{x}_j$, they can be finally represented as $f(\mathbf{x}_i) = \mathbf{h}_i^{(M)}$ and $f(\mathbf{x}_j) = \mathbf{h}_j^{(M)}$ at the top level when they are passed through the $(M+1)$-layer deep network, and their similarity can be measured by computing the squared Euclidean distance between the most top level representations, which is defined as follows:

$$d_f^2(\mathbf{x}_i, \mathbf{x}_j) = \big\|f(\mathbf{x}_i) - f(\mathbf{x}_j)\big\|_2^2. \tag{5}$$

It is desirable to exploit discriminative information for face representations at the most top level in our DDML model, which is more effective to verification. To achieve this, we expect the distances between positive pairs are smaller than those between negative pairs and develop a large margin framework to formulate our method. Fig. 2 illustrates the basic idea of our proposed DDML method. There are three face samples in the original feature space, which are used to generate two pairs of face images, where two of them form a positive pair (two circles) and two of them form the negative pair (one circle in the center and one triangle), respectively. In the original face feature space, the distance between the positive pair is larger than that between the negative pair which may be caused by the large intra-personal variations such as varying expressions, illuminations, and poses, especially when face images are captured in the wild. This is harmful to face verification because it causes an error. To address this, DDML aims to seek a nonlinear mapping f such that the distance of the positive pair is less than a small threshold τ_1 ($\tau_1 > 0$) and that of the negative pair is higher than a large threshold τ_2 ($\tau_2 > \tau_1$) of the most top level of our DDML model, respectively, so that more discriminative information can be exploited and the face pair can be easily verified.

To reduce the parameter numbers in our experiments, we only employ one threshold τ ($\tau > 1$) to connect τ_1 and τ_2, and enforce the margin between $d_f^2(\mathbf{x}_i, \mathbf{x}_j)$ and τ is larger than 1 by using the following constraint:

$$\ell_{ij}\big(\tau - d_f^2(\mathbf{x}_i, \mathbf{x}_j)\big) > 1, \tag{6}$$

where we have $\tau_1 = \tau - 1$ and $\tau_2 = \tau + 1$. With this constrain, there is a margin between each positive and negative pairs in the learned feature space. The pairwise label ℓ_{ij} denotes the similarity or dissimilarity of a face pair $\mathbf{x}_i$ and $\mathbf{x}_j$, i.e., $\ell_{ij} = 1$

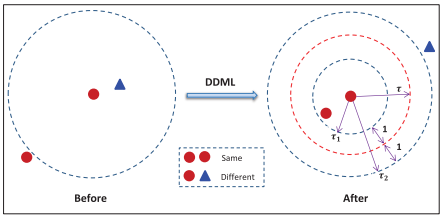


Fig. 2. Illustration of the basic idea of the proposed DDML method.

if $\mathbf{x}_i$ and $\mathbf{x}_j$ are from the same subject, and $\ell_{ij} = -1$ if $\mathbf{x}_i$ and $\mathbf{x}_j$ are from different subjects.

By applying the above constrain in (6) to each positive and negative pair in the training set, we formulate the DDML as the following optimization problem:

$$\arg\min_{f} J = \frac{1}{2}\sum_{i,j} g\Big(1 - \ell_{ij}\big(\tau - d_f^2(\mathbf{x}_i, \mathbf{x}_j)\big)\Big) + \frac{\lambda}{2}\sum_{m=1}^{M}\Big(\big\|\mathbf{W}^{(m)}\big\|_F^2 + \big\|\mathbf{b}^{(m)}\big\|_2^2\Big), \quad (7)$$

where $g(z) = \frac{1}{\beta}\log\big(1 + \exp(\beta z)\big)$ is the generalized logistic loss function [31], which is a smoothed approximation of the hinge loss function $[z]_+ = \max(z, 0)$, β is a sharpness parameter, $\|\mathbf{A}\|_F$ represents the Frobenius norm of the matrix $\mathbf{A}$, λ is a regularization parameter. There are two terms in our objective function (7), where the first term defines the logistic loss of training samples and the second term represents the regularizer of the projection matrix and bias, respectively.

To solve the optimization problem in (7), we use the stochastic sub-gradient descent scheme to obtain the parameters $\{\mathbf{W}^{(m)}, \mathbf{b}^{(m)}\}$, where $m = 1, 2, \cdots, M$. The gradient of the objective function J with respect to the parameters $\mathbf{W}^{(m)}$ and $\mathbf{b}^{(m)}$ is computed as follows:

$$\frac{\partial J}{\partial \mathbf{W}^{(m)}} = \sum_{i,j}\Big(\Delta_{ij}^{(m)}\mathbf{h}_i^{(m-1)^T} + \Delta_{ji}^{(m)}\mathbf{h}_j^{(m-1)^T}\Big) + \lambda\ \mathbf{W}^{(m)}, \quad (8)$$

$$\frac{\partial J}{\partial \mathbf{b}^{(m)}} = \sum_{i,j}\Big(\Delta_{ij}^{(m)} + \Delta_{ji}^{(m)}\Big) + \lambda\ \mathbf{b}^{(m)}, \quad (9)$$

where $\mathbf{h}_i^{(0)} = \mathbf{x}_i$ and $\mathbf{h}_j^{(0)} = \mathbf{x}_j$, which are the original inputs of our network. For the other layers $m = 1, 2, \cdots, M-1$, we update them as follows:

$$\Delta_{ij}^{(M)} = g'(c)\ \ell_{ij}\ \Big(\mathbf{h}_i^{(M)} - \mathbf{h}_j^{(M)}\Big) \odot s'\Big(\mathbf{z}_i^{(M)}\Big), \quad (10)$$

$$\Delta_{ji}^{(M)} = g'(c)\ \ell_{ij}\ \Big(\mathbf{h}_j^{(M)} - \mathbf{h}_i^{(M)}\Big) \odot s'\Big(\mathbf{z}_j^{(M)}\Big), \quad (11)$$

$$\Delta_{ij}^{(m)} = \Big(\mathbf{W}^{(m+1)^T}\Delta_{ij}^{(m+1)}\Big) \odot s'\Big(\mathbf{z}_i^{(m)}\Big), \quad (12)$$

$$\Delta_{ji}^{(m)} = \Big(\mathbf{W}^{(m+1)^T}\Delta_{ji}^{(m+1)}\Big) \odot s'\Big(\mathbf{z}_j^{(m)}\Big), \quad (13)$$

where the operation $\odot$ denotes the element-wise multiplication, and c and $\mathbf{z}_i^{(m)}$ are defined as follows:

$$c \triangleq 1 - \ell_{ij}\big(\tau - d_f^2(\mathbf{x}_i, \mathbf{x}_j)\big), \quad (14)$$

$$\mathbf{z}_i^{(m)} \triangleq \mathbf{W}^{(m)}\mathbf{h}_i^{(m-1)} + \mathbf{b}^{(m)}. \quad (15)$$

Algorithm 1 DDML

Input: Training set: $\mathbf{X} = \{(\mathbf{x}_i, \mathbf{x}_j, \ell_{ij})\}$, number of network layers $M+1$, threshold τ, learning rate μ, iterative number I_t, parameter λ, and convergence error ε.
Output: Weights and biases: $\{\mathbf{W}^{(m)}, \mathbf{b}^{(m)}\}_{m=1}^{M}$.
// *Initialization:*
Initialize $\{\mathbf{W}^{(m)}, \mathbf{b}^{(m)}\}_{m=1}^{M}$ according to (41).
// *Optimization by back propagation:*
for $t = 1, 2, \cdots, I_t$ **do**
  Randomly select a sample pair $(\mathbf{x}_i, \mathbf{x}_j, \ell_{ij})$ in $\mathbf{X}$.
  Set $\mathbf{h}_i^{(0)} = \mathbf{x}_i$ and $\mathbf{h}_j^{(0)} = \mathbf{x}_j$, respectively.
  // *Forward propagation*
  for $m = 1, 2, \cdots, M$ **do**
    Do forward propagation to get $\mathbf{h}_i^{(m)}$ and $\mathbf{h}_j^{(m)}$.
  end
  // *Computing gradient*
  for $m = M, M-1, \cdots, 1$ **do**
    Obtain gradient according to (8) and (9).
  end
  // *Back propagation*
  for $m = 1, 2, \cdots, M$ **do**
    Update $\mathbf{W}^{(m)}$ and $\mathbf{b}^{(m)}$ according to (16) and (17).
  end
  Calculate J_t using (7).
  If $t > 1$ and $|J_t - J_{t-1}| < \varepsilon$, go to **Return**.
end
Return: $\{\mathbf{W}^{(m)}, \mathbf{b}^{(m)}\}_{m=1}^{M}$.

Then, the parameters $\mathbf{W}^{(m)}$ and $\mathbf{b}^{(m)}$ can be updated by using the following gradient descent algorithm until convergence:

$$\mathbf{W}^{(m)} = \mathbf{W}^{(m)} - \mu\frac{\partial J}{\partial \mathbf{W}^{(m)}}, \quad (16)$$

$$\mathbf{b}^{(m)} = \mathbf{b}^{(m)} - \mu\frac{\partial J}{\partial \mathbf{b}^{(m)}}, \quad (17)$$

where μ is the learning rate, which is set to 0.001 in our experiments.

Algorithm 1 summarizes the detailed procedure of the proposed DDML method.

C. DDMML

DDML learns one neural networks from a single feature representation and cannot deal with multiple feature representations directly. In face and kinship verification, it is easy to extract multiple features for each face image for multiple feature fusion. However, these features extracted from the same face image are usually highly correlated to each other even if they could characterize face images from different aspects [48]. For multiple feature fusion, these highly correlated information should be preserved because they usually reflect the intrinsic information of samples. An important principle to perform multi-feature metric learning is to jointly learn multiple distance metrics by preserving the correlation between different feature pairs.

Previous studies [48]–[50] have shown that canonical correlation analysis (CCA) is an effective technique to fuse bi-modal features by maximizing their correlation, where a pair of projections are learned to map features in the original

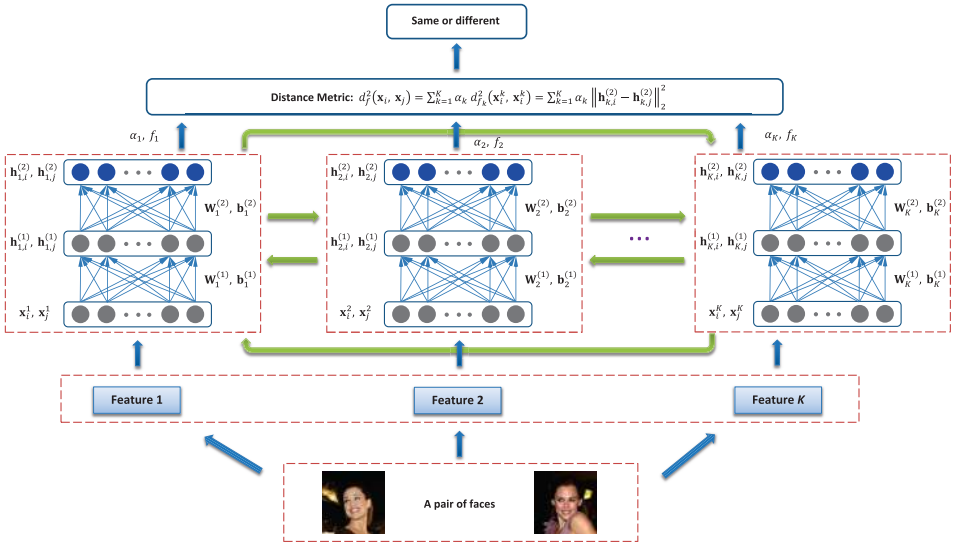


Fig. 3. The flowchart of DDMML method for face verification. For a given pair of face images $\mathbf{x}_i$ and $\mathbf{x}_j$, we first extract K features for each image and map them into K different feature subspaces $\mathbf{h}_{k,i}^{(2)}$ and $\mathbf{h}_{k,j}^{(2)}$ $(k = 1, 2, \cdots, K)$ by jointly learning K sets of hierarchical nonlinear transformations, where the similarity of their outputs at the most top level is adaptively combined and weighted to determine whether the given pair is from the same person or not.

space into a latent space. There are two key advantages for CCA-based multiple feature fusion: 1) the effects of noise can be largely reduced and the signal-to-noise ratio (SNR) is enlarged [50], and 2) the most correlated information across multiple features is exploited and preserved [48]. Therefore, a number of CCA-based information fusion methods have been proposed in recent years, and some of them have been successfully applied in different face analysis tasks. Motivated by the success of CCA, we propose a DDMML method to learn K mapping $\{f_k\}_{k=1}^K$ under which discriminative information is exploited in each feature space individually, and difference of feature representations of each pair of face samples is enforced to be as small as possible, which is consistent to the canonical correlation analysis-based multiple feature fusion approach. Fig. 3 illustrates the basic idea of the DDMML approach.

Let $\mathbf{X}_k = [\mathbf{x}_1^k, \mathbf{x}_2^k, \cdots, \mathbf{x}_N^k]$ be the feature set from the kth feature representation in the training set, where $\mathbf{x}_i^k \in \mathbb{R}^{d_k}$ is the ith training sample from the kth feature set, and N is the total number of training samples. The squared Euclidean distance between a pair of samples $\mathbf{x}_i^k$ and $\mathbf{x}_j^k$ in the kth feature set at the most top level is computed as:

$$d_{f_k}^2(\mathbf{x}_i^k, \mathbf{x}_j^k) = \left\| f_k(\mathbf{x}_i^k) - f_k(\mathbf{x}_j^k) \right\|_2^2, \tag{18}$$

where the mapping $f_k : \mathbb{R}^{d_k} \mapsto \mathbb{R}^{p_k^{(M_k)}}$ is a parametric nonlinear function determined by the parameters $\mathbf{W}_k^{(m)}$ and $\mathbf{b}_k^{(m)}$, where $m = 1, 2, \cdots, M_k$.

According to (7), the DDML model in the kth feature space can be rewritten as:

$$\arg\min_{f_k} J_k = \frac{1}{2} \sum_{i,j} g\Big(1 - \ell_{ij}\big(\tau_k - d_{f_k}^2(\mathbf{x}_i^k, \mathbf{x}_j^k)\big)\Big) + \frac{\lambda_k}{2} \sum_{m=1}^{M_k} \Big(\left\| \mathbf{W}_k^{(m)} \right\|_F^2 + \left\| \mathbf{b}_k^{(m)} \right\|_2^2 \Big), \tag{19}$$

where λ_k is a regularization parameter, and τ_k $(\tau_k > 1)$ is a threshold.

Since there are K features in the training set, we aim to learn K networks jointly by DDMML, where f_k is the kth mapping for the kth feature. Moreover, the learning procedure should satisfy the following two characteristics: 1) The discriminative information from each single feature is exploited as much as possible; 2) The differences of different feature representations for each sample in the learned networks are minimized.

To address this, we formulate DDMML as the following optimization problem:

$$\begin{aligned} \min_{f_1, \cdots, f_K} \; & J = \sum_{k=1}^{K} \alpha_k \; J_k \\ & + \omega \sum_{\substack{k,u=1 \\ k<u}}^{K} \sum_{i,j} \Big(d_{f_k}(\mathbf{x}_i^k, \mathbf{x}_j^k) - \pi_{ku} \; d_{f_u}(\mathbf{x}_i^u, \mathbf{x}_j^u) \Big)^2, \\ \text{s.t.} \; & \sum_{k=1}^{K} \alpha_k = 1, \quad \alpha_k \geq 0, \; \omega > 0, \end{aligned} \tag{20}$$

where α_k is a nonnegative weighting parameter to reflect different importance of different features, ω weights the pairwise difference of the distance between two samples $\mathbf{x}_i$ and $\mathbf{x}_j$ at the most top layer of the network by using mappings f_k and f_u, and π_{ku} is a balancing factor. Since the difference of the sample $\mathbf{x}_i$ from the kth and uth $(1 \leq k, u \leq K, k \neq u)$ feature representations relies on the mappings f_k and f_u, which could be of different dimensions, it is infeasible to compute them directly. Here we use an alternative constrain to reflect the relationship of different feature representations. Since the difference of $\mathbf{x}_i^k$ and $\mathbf{x}_i^u$, and that of $\mathbf{x}_j^k$ and $\mathbf{x}_j^u$ are expected to be minimized as much as possible, the distance between $\mathbf{x}_i^k$ and $\mathbf{x}_j^k$, and that of $\mathbf{x}_i^u$ and $\mathbf{x}_j^u$ are also expected to be as small as possible.

Having obtained multiple mappings $\{f_k\}_{k=1}^K$, the distance between two samples $\mathbf{x}_i$ and $\mathbf{x}_j$ can be computed as:

$$d_f^2(\mathbf{x}_i, \mathbf{x}_j) = \sum_{k=1}^{K} \alpha_k \; d_{f_k}^2(\mathbf{x}_i^k, \mathbf{x}_j^k). \tag{21}$$

The trivial solution to (21) is $\alpha_k = 1$, which corresponds to the minimum J_k over different feature representations, and $\alpha_k = 0$ otherwise, which cannot exploit the common property from multiple feature descriptors.

To address this shortcoming, we modify α_k to be α_k^r $(r > 1)$, then revisit the new objective function as:

$$\begin{aligned} \min_{f_1,\cdots,f_K} \; & J = \sum_{k=1}^{K} \alpha_k^r \; J_k \\ & + \omega \sum_{\substack{k,u=1 \\ k<u}}^{K} \sum_{i,j} \Big(d_{f_k}(\mathbf{x}_i^k, \mathbf{x}_j^k) - \pi_{ku} \; d_{f_u}(\mathbf{x}_i^u, \mathbf{x}_j^u) \Big)^2, \\ \text{s.t.} \; & \sum_{k=1}^{K} \alpha_k = 1, \quad \alpha_k \geq 0, \; \omega > 0. \end{aligned} \tag{22}$$

To our best knowledge, there is no closed-form solution to the optimization problem in (22) because we aim to learn K nonlinear mapping functions and the weighting vector simultaneously. In this work, we use an alternating minimization algorithm. The approach is to fix $\boldsymbol{\alpha}$ and $f_1, \cdots, f_{k-1}, f_{k+1}, \cdots, f_K$, update f_k, and then fix $f_1, f_2, \cdots, f_K$, update $\boldsymbol{\alpha}$.

Step 1 (Fix $\boldsymbol{\alpha}$ and $\boldsymbol{f}_1, \cdots, \boldsymbol{f}_{k-1}, \boldsymbol{f}_{k+1}, \cdots, \boldsymbol{f}_K$, update $\boldsymbol{f}_k$): Having fixed $\boldsymbol{\alpha}$ and $f_1, \cdots, f_{k-1}, f_{k+1}, \cdots, f_K$, equation (22) can be rewritten as:

$$\begin{aligned} \min_{\mathbf{f}_k} \; & J = \alpha_k^r \; J_k \; + \; A_k \\ & + \omega \sum_{\substack{u=1 \\ u \neq k}}^{K} \sum_{i,j} \Big(d_{f_k}(\mathbf{x}_i^k, \mathbf{x}_j^k) - \pi_{ku} \; d_{f_u}(\mathbf{x}_i^u, \mathbf{x}_j^u) \Big)^2, \end{aligned} \tag{23}$$

where A_k is a constant term.

Similar to DDML, we also use the stochastic sub-gradient descent scheme to obtain the parameters $\{\mathbf{W}_k^{(m)}, \mathbf{b}_k^{(m)}\}$ in f_k, where $m = 1, 2, \cdots, M_k$. Specifically, the gradient of the objective function J with respect to the parameters $\mathbf{W}_k^{(m)}$ and $\mathbf{b}_k^{(m)}$ can be computed as:

$$\begin{aligned} \frac{\partial J}{\partial \mathbf{W}_k^{(m)}} = & \sum_{i,j} \Big(\Delta_{k,ij}^{(m)} \; \mathbf{h}_{k,i}^{(m-1)^T} + \Delta_{k,ji}^{(m)} \; \mathbf{h}_{k,j}^{(m-1)^T} \Big) \\ & + \lambda_k \; \mathbf{W}_k^{(m)}, \end{aligned} \tag{24}$$

$$\frac{\partial J}{\partial \mathbf{b}_k^{(m)}} = \sum_{i,j} \Big(\Delta_{k,ij}^{(m)} + \Delta_{k,ji}^{(m)} \Big) + \lambda_k \; \mathbf{b}_k^{(m)}, \tag{25}$$

where $\mathbf{h}_{k,i}^{(0)} = \mathbf{x}_i^k$ and $\mathbf{h}_{k,j}^{(0)} = \mathbf{x}_j^k$, which are the original inputs of our networks. For the other layers $m = 1, 2, \cdots, M_k - 1$, we have the following updating rules:

$$\Delta_{k,ij}^{(M_k)} = \rho_k \left[\Big(\mathbf{h}_{k,i}^{(M_k)} - \mathbf{h}_{k,j}^{(M_k)} \Big) \odot s' \Big(\mathbf{z}_{k,i}^{(M_k)} \Big) \right], \tag{26}$$

$$\Delta_{k,ji}^{(M_k)} = \rho_k \left[\Big(\mathbf{h}_{k,j}^{(M_k)} - \mathbf{h}_{k,i}^{(M_k)} \Big) \odot s' \Big(\mathbf{z}_{k,j}^{(M_k)} \Big) \right], \tag{27}$$

$$\Delta_{k,ij}^{(m)} = \Big(\mathbf{W}_k^{(m+1)^T} \Delta_{k,ij}^{(m+1)} \Big) \odot s' \Big(\mathbf{z}_{k,i}^{(m)} \Big), \tag{28}$$

$$\Delta_{k,ji}^{(m)} = \Big(\mathbf{W}_k^{(m+1)^T} \Delta_{k,ji}^{(m+1)} \Big) \odot s' \Big(\mathbf{z}_{k,j}^{(m)} \Big), \tag{29}$$

where the operation $\odot$ denotes the element-wise multiplication, and ρ_k, c_k and $\mathbf{z}_{i,k}^{(m)}$ are defined as follows:

$$\rho_k \triangleq \alpha_k^r \; g'(c_k) \; \ell_{ij} + \omega \sum_{\substack{u=1 \\ u \neq k}}^{K} \Big(1 - \pi_{ku} \; \frac{d_{f_u}(\mathbf{x}_i^u, \mathbf{x}_j^u)}{d_{f_k}(\mathbf{x}_i^k, \mathbf{x}_j^k)} \Big), \tag{30}$$

$$c_k \triangleq 1 - \ell_{ij} \big(\tau - d_{f_k}^2(\mathbf{x}_i^k, \mathbf{x}_j^k) \big), \tag{31}$$

$$\mathbf{z}_{k,i}^{(m)} \triangleq \mathbf{W}_k^{(m)} \mathbf{h}_{k,i}^{(m-1)} + \mathbf{b}_k^{(m)}. \tag{32}$$

Then, $\mathbf{W}_k^{(m)}$ and $\mathbf{b}_k^{(m)}$ can be updated with the following gradient descent algorithm until convergence:

$$\mathbf{W}_k^{(m)} = \mathbf{W}_k^{(m)} - \mu_k \frac{\partial J}{\partial \mathbf{W}_k^{(m)}}, \tag{33}$$

$$\mathbf{b}_k^{(m)} = \mathbf{b}_k^{(m)} - \mu_k \frac{\partial J}{\partial \mathbf{b}_k^{(m)}}. \tag{34}$$

Step 2 (Fix $\boldsymbol{f}_1, \boldsymbol{f}_2, \cdots, \boldsymbol{f}_K$, update $\boldsymbol{\alpha}$): We construct a Lagrange function as follows:

$$\begin{aligned} La(\boldsymbol{\alpha}, \eta) = & \sum_{k=1}^{K} \alpha_k^r \; J_k \\ & + \omega \sum_{\substack{k,u=1 \\ k<u}}^{K} \sum_{i,j} \Big(d_{f_k}(\mathbf{x}_i^k, \mathbf{x}_j^k) - \pi_{ku} \; d_{f_u}(\mathbf{x}_i^u, \mathbf{x}_j^u) \Big)^2 \\ & - \eta \Big(\sum_{k=1}^{K} \alpha_k - 1 \Big). \end{aligned} \tag{35}$$

Let $\frac{\partial La(\boldsymbol{\alpha}, \eta)}{\partial \alpha_k} = 0$ and $\frac{\partial La(\boldsymbol{\alpha}, \eta)}{\partial \eta} = 0$, we have

$$\frac{\partial La(\boldsymbol{\alpha}, \eta)}{\partial \alpha_k} = r \; \alpha_k^{r-1} \; J_k - \eta = 0, \tag{36}$$

$$\frac{\partial La(\boldsymbol{\alpha}, \eta)}{\partial \eta} = \sum_{k=1}^{K} \alpha_k - 1 = 0. \tag{37}$$

According to (36) and (37), α_k can be updated as:

$$\alpha_k = \frac{\big(1/J_k \big)^{1/(r-1)}}{\sum_{k=1}^{K} \big(1/J_k \big)^{1/(r-1)}}. \tag{38}$$

We repeat the above two steps until DDMML converges. **Algorithm 2** summarizes the detailed procedure of the proposed DDMML method.

D. Implementation Details

In this subsection, we discuss the nonlinearity activation functions and initializations of $\mathbf{W}^{(m)}$ and $\mathbf{b}^{(m)}$, $1 \leq m \leq M$ in our proposed DDML and DDMML methods.

Algorithm 2 DDMML

Input: Training set $\mathbf{X} = \{\mathbf{X}_k\}_{k=1}^K = \{(\mathbf{x}_i^k, \mathbf{x}_j^k, \ell_{ij})\}_{k=1}^K$ from K views; Number of network layers $\{M_k + 1\}_{k=1}^K$; Threshold $\{\tau_k\}_{k=1}^K$; Learning rate $\{\mu_k\}_{k=1}^K$; Iterative number I_t; Parameters $\{\lambda_k\}_{k=1}^K$, ω and r; Convergence error ε.
Output: Parameters: $\{\mathbf{W}_k^{(m)}, \mathbf{b}_k^{(m)}\}_{m=1}^M$ and α_k, $k = 1, 2, \cdots, K$.
// Initialization:
Initialize $\{\mathbf{W}_k^{(m)}, \mathbf{b}_k^{(m)}\}_{m=1}^M$ according to (41), and $\alpha_k = 1/K$ for $k = 1, 2, \cdots, K$.
// Optimization by back propagation:
for $t = 1, 2, \cdots, I_t$ **do**
  Randomly select a sample pair $(\mathbf{x}_i, \mathbf{x}_j, \ell_{ij})$ in $\mathbf{X}$.
  for $k = 1, 2, \cdots, K$ **do**
    Set $\mathbf{h}_{k,i}^{(0)} = \mathbf{x}_i^k$ and $\mathbf{h}_{k,j}^{(0)} = \mathbf{x}_j^k$, respectively.
    // Forward propagation
    for $m = 1, 2, \cdots, M_k$ **do**
      Do forward propagation to get $\mathbf{h}_{k,i}^{(m)}$ and $\mathbf{h}_{k,j}^{(m)}$.
    end
    Obtain ρ_k by (30).
    // Computing gradient
    for $m = M_k, M_k - 1, \cdots, 1$ **do**
      Obtain gradient according to (24) and (25).
    end
    // Back propagation
    for $m = 1, 2, \cdots, M_k$ **do**
      Update $\mathbf{W}_k^{(m)}$ and $\mathbf{b}_k^{(m)}$ by (33) and (34).
    end
  end
  Compute $\{\alpha_k\}_{k=1}^K$ according to (38).
  Calculate J_t using Eq (22).
  If $t > 1$ and $|J_t - J_{t-1}| < \varepsilon$, go to **Return**.
end
Return: $\{\mathbf{W}_k^{(m)}, \mathbf{b}_k^{(m)}\}_{m=1}^M$, and α_k, $k = 1, 2, \cdots, K$.

1) Activation Function: There are many nonlinearity activation functions which could be used to determine the output of the nodes in our deep metric learning model. In our experiments, we use the *tanh* function as the activation function because we found it achieved better performance than others in View 1 of LFW dataset [8]. There are two views in LFW dataset [8], View 1 and View 2. View 1 is used for model selection and algorithm development and View 2 is used for performance reporting. In View 1 of LFW, the training set contains 1100 positive pairs and 1100 negative pairs, and the test set contains 500 positive pairs and 500 negative pairs. We evaluated three nonlinear activation functions (i.e., sigmoid, ns-sigmoid, and tanh) in View 1 of LFW. We find the tanh function reports the best accuracy in View 1 of LFW, therefore we employ the tanh function as nonlinear activation function for all datasets. The *tanh* function and its derivative can be computed as follows:

$$s(z) = \tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}, \tag{39}$$

$$s'(z) = \tanh'(z) = 1 - \tanh^2(z). \tag{40}$$

2) Initialization: The initializations of $\mathbf{W}^{(m)}$ and $\mathbf{b}^{(m)}$ ($1 \leq m \leq M$) are important to the gradient descent based method in our deep neural networks. Random initialization and denoising autoencoder (DAE) [51] are two popular initialization methods in deep learning. In our experiments, we utilize a simple normalized random initialization method in [52], where the bias $\mathbf{b}^{(m)}$ is initialized as $\mathbf{0}$, and the weight of each layer is initialized as the following uniform distribution:

$$\mathbf{W}^{(m)} \sim U\left[-\frac{\sqrt{6}}{\sqrt{p^{(m)} + p^{(m-1)}}}, \frac{\sqrt{6}}{\sqrt{p^{(m)} + p^{(m-1)}}}\right], \tag{41}$$

where $p^{(0)}$ is the dimension of input layer and $1 \leq m \leq M$.

IV. Experiments

To evaluate the effectiveness of our DDML and DDMML methods, we perform unconstrained face and kinship verification experiments on the LFW [8], YTF [16], KinFaceW-I [5], KinFaceW-II [5], and TSKinFace [53] datasets.

A. Datasets and Experimental Settings

The LFW dataset [8] contains more than 13000 face images of 5749 subjects captured from the web with variations in expression, pose, age, illumination, resolution, background, and so on. There are two training paradigms for supervised learning on this dataset: 1) *image restricted* and 2) *image unrestricted*. In our experiments, we evaluate our proposed methods on both of these two settings. We follow the standard evaluation protocol on the "View 2" dataset [8] which includes 3000 matched pairs and 3000 mismatched pairs. The dataset is divided into 10 folds, and each fold consists of 300 matched (positive) pairs and 300 mismatched (negative) pairs. We use three types of the LFW dataset for feature extraction in our evaluation: the original LFW, LFW-a, and "funneled" LFW. For the original LFW dataset, we employ the over-completed high-dimensional LBP (HDLBP) provided by [54] as the feature representation for each image. Specifically, it densely samples multi-scale LBP features at each landmarks and then these features are concatenated into a high-dimensional feature vector (more than 120000 dimension). For the LFW-a dataset, we crop each image into 80×150 from center of this image to remove the background information and extract the histogram of oriented gradients (HOG) [55] from two different scales for each image. Specifically, we first divide each image into 16×30 non-overlapping blocks, where the size of each block is 5×5. Then, we divide each image into 8×15 non-overlapping blocks, where the size of each block is 10×10. Subsequently, we extract a 9-dimensional HOG feature for each block and concatenate them to form a 5400-dimensional feature vector. For the "funneled" LFW dataset, we use the Sparse SIFT (SSIFT) descriptor provided by [10]. Specifically, these SSIFT descriptors are computed at the nine fixed landmarks with three different scales, and are concatenated into a 3456-dimensional feature vector. As suggested in [11], [21], and [56], we also use the square root of each feature and evaluate the performance of our DDMML method when all the six different feature descriptors are used to learn the model. For each feature descriptor, we apply whitened PCA (WPCA) to reduce it into an appropriate dimension to further remove the redundancy. Specifically,

TABLE I
THE DIMENSION PARAMETERS OF OUR DDML AND DDMML IN OUR EXPERIMENTS

Layer	LFW			YTF			KinFaceW-I & II			
	SSIFT $k=1$	HOG $k=2$	HDLBP $k=3$	CSLBP $k=1$	FPLBP $k=2$	LBP $k=3$	LBP $k=1$	DSIFT $k=2$	HOG $k=3$	LPQ $k=4$
$\mathbf{W}_k^{(1)}$	200×300	200×300	250×400	150×200	150×200	150×200	150×200	150×200	150×200	150×200
$\mathbf{b}_k^{(1)}$	200×1	200×1	250×1	150×1	150×1	150×1	150×1	150×1	150×1	150×1
$\mathbf{W}_k^{(2)}$	150×200	150×200	200×250	80×150	80×150	80×150	100×150	100×150	100×150	100×150
$\mathbf{b}_k^{(2)}$	150×1	150×1	200×1	80×1	80×1	80×1	100×1	100×1	100×1	100×1

the WPCA projection was computed on the training data of each cross-validation split. For the image unrestricted setting, we generated 14500 positive pairs and 14500 negative pairs to learn the parameters of our models. In our experiments, we select one fold as the testing set and use face images in the other nine folds as the training set. For positive pairs, if the number of face images for a person in the training set is larger than 10, we randomly select 10 images for this person and use all these 10 images to construct 45 positive pairs. If the number of face images of this person is smaller than 10, we use all face images of this person to generate positive pairs if there are more than one face image for this person. For negative pairs, we randomly choose two face images from two different persons from the training set to construct a negative pair and we sample 14500 negative pairs finally. We repeat this procedure 10 times to train our models on the training set and evaluate them in the testing set.

The YTF dataset [16] contains 3425 videos of 1595 different persons collected from the YouTube website. There are large variations in pose, illumination, and expression in each video, and the average length of each video clip is 181.3 frames. In our experiments, we follow the standard evaluation protocol [16] and test our method for unconstrained face verification with 5000 video pairs. These pairs are equally divided into 10 folds, and each fold has 250 intra-personal pairs and 250 inter-personal pairs. We adopt the *image restricted* protocol to evaluate our method. For this dataset, we directly use the provided three feature descriptors [16] including LBP, Center-Symmetric LBP (CSLBP) [16] and Four-Patch LBP (FPLBP) [9]. Since all face images have been aligned by the detected facial landmarks, we average all the feature vectors within one video clip to form a mean feature vector in our experiments. Lastly, we use WPCA to project each mean vector into a vector with appropriate dimension. For both LFW and YTF datasets, we follow the standard protocol in [8] and [16] and use two measures including the mean verification accuracy (%) with standard error and the receiving operating characteristic (ROC) curve from the ten-fold cross validation to validate our methods.

The KinFaceW-I [5] and KinFaceW-II [5] are two kinship face datasets collected from the public figures or celebrities. For each person in these two datasets, face image of his/her parent or child was also collected. There are four kinship relations in these two datasets: F-S, F-D, M-S and M-D. In KinFaceW-I, there are 156, 134, 116, and 127 pairs of kinship images for these four relations. In KinFaceW-II, each relation contains 250 pairs of kinship images. For these two datasets, we directly use each aligned 64×64 image for feature extraction. For each face image, we extract four feature descriptors: 1) LBP: we divide each image into 8×8 non-overlapping blocks, where the size of each block is 8×8. We extract a 59-dimensional uniform pattern LBP feature for each block and concatenate them to form a 3776-dimensional feature vector; 2) Dense SIFT (DSIFT): we densely sample SIFT descriptors on each 16×16 patch with stepsize of 8 pixels and obtain 49 SIFT descriptors. Then, we concatenate these SIFT descriptors to form a 6272-dimensional feature vector; 3) HOG: we first divide each image into 16×16 non-overlapping blocks, where the size of each block is 4×4. Then, we divide each image into 8×8 non-overlapping blocks, where the size of each block is 8×8. Subsequently, we extract a 9-dimensional HOG feature for each block and concatenate them to form a 2880-dimensional feature vector; and 4) Local phase quantization (LPQ) [57]: we first divide each image into 4×4 non-overlapping blocks, where the size of each block is 16×16. Then, we extract a 256-bin LPQ histogram with window size of 3, 5 and 7 for each block respectively, and finally concatenate them to form a 12288-dimensional feature vector. Following the same experimental settings in [5], we adopt the five-fold cross validation strategy under the image restricted setting and the mean verification rate is used for the kinship verification performance evaluation.

For DDML, we train a deep network with three layers ($M = 2$), and the threshold τ, learning rate μ and regularization parameter λ are empirically set as 3, 10^{-3}, 10^{-2} for all experiments, respectively. For DDMML, we jointly train multiple deep networks where each corresponds to one network in DDML, and the parameters are set as: $\omega = 10^{-2}$, $r = 2$, $M_k = 2$, $\tau_k = 3$, $\mu_k = 10^{-3}$, $\lambda_k = 10^{-2}$ and $\pi_{ku} = 1$, where $k = 1, 2, \cdots, K$. The parameters $\{\mathbf{W}^{(m)}, \mathbf{b}^{(m)}\}_{m=1}^{M}$ of DDML and $\{\mathbf{W}_k^{(m)}, \mathbf{b}_k^{(m)}\}_{m=1}^{M}$ of DDMML were set as shown in Table I. In addition, we also evaluated DDMML with different balancing factor π_{ku} in set $\{0.4, 0.6, 0.8, 1, 1.5, 2\}$ on the LFW dataset under the image-restricted setting, and the mean verification rate (%) of the DDMML is in the range of $[93.12, 93.36]$ for the various π_{ku}. Therefore, we simply set $\pi_{ku} = 1$ in our experiments.

Having trained our DDML and DDMML models, we apply them to each test face pair to measure their similarity. Given each pair of test images, we first compute their distance d_f^2 using our DDML and DDMML and then predict them as a matched (positive) pair if d_f^2 is smaller than a threshold, and vice versa. In DDMML, we first compute the distance d_f^2

TABLE II

COMPARISONS (%) WITH SHALLOW METRIC LEARNING METHODS ON LFW UNDER THE IMAGE RESTRICTED SETTING

Feature	DDML	DSML
SSIFT	86.98 ± 0.43	83.67 ± 0.35
SSIFT (sqrt)	87.83 ± 0.29	84.55 ± 0.29
HOG	87.35 ± 0.58	85.22 ± 0.62
HOG (sqrt)	88.38 ± 0.77	86.40 ± 0.55
HDLBP	91.37 ± 0.39	88.75 ± 0.37
HDLBP (sqrt)	92.55 ± 0.36	90.47 ± 0.28
Feature	DDMML	DSMML
All	$\mathbf{93.28 \pm 0.39}$	91.10 ± 0.38

TABLE III

COMPARISONS (%) WITH SHALLOW METRIC LEARNING METHODS ON LFW UNDER THE IMAGE UNRESTRICTED SETTING

Feature	DDML	DSML
SSIFT	88.42 ± 0.52	85.35 ± 0.43
SSIFT (sqrt)	88.92 ± 0.45	85.63 ± 0.42
HOG	88.08 ± 0.62	86.25 ± 0.54
HOG (sqrt)	89.18 ± 0.69	87.60 ± 0.61
HDLBP	92.62 ± 0.35	90.05 ± 0.35
HDLBP (sqrt)	93.70 ± 0.37	91.57 ± 0.33
Feature	DDMML	DSMML
All	$\mathbf{94.50 \pm 0.35}$	92.07 ± 0.40

between $\mathbf{x}_i$ and $\mathbf{x}_j$ using (21), then we compare it with the threshold $\sum_{k=1}^{K} \alpha_k \ \tau_k$ for verification.

B. Experimental Comparison on LFW

1) Deep vs. Shallow Metric Learning: We first compare our methods with shallow metric learning methods. Two shallow metric learning methods called discriminative shallow metric learning (DSML) and discriminative shallow multi-metric learning (DSMML) are constructed, where only one layer ($M = 1$) is learned in our methods and the activation function is $s(z) = z$. For example, DSML is the linear case of DDML, i.e., the linear activation function $s(z) = z$ and no hidden layer ($M = 1$) are adopted in the DDML. Tables II and III record the verification rates with standard error of these methods when different feature descriptors are used. We see that our DDML and DDMML consistently outperform DSML and DSMML in terms of the mean verification rate. This is because DDML and DDMML learn hierarchical nonlinear transformations while DSML and DSMML only learn a linear transformation, so that DDML and DDMML can better model the nonlinear relationship of face samples.

2) Comparison With the State-of-the-Art Face Verification Methods: We compare our methods with the state-of-the-art face verification methods on the LFW dataset under the image restricted and unrestricted settings. Tables IV and V show the mean verification rate with standard error and Fig. 4 shows the ROC curves of different methods, respectively. We see that our DDML and DDMML improve the current state-of-the-art by 1.45% and 2.18% in terms of the mean verification rate under the image restricted paradigm, and 0.52% and 1.32% under the image unrestricted paradigm without using outside training data, respectively.

TABLE IV

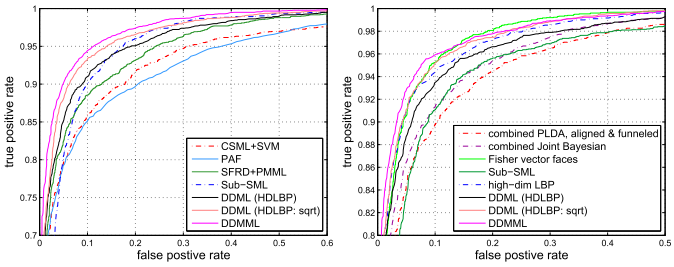
COMPARISONS (%) WITH STATE-OF-THE-ART FACE VERIFICATION METHODS ON LFW UNDER THE IMAGE RESTRICTED SETTING, WHERE NOD DENOTES THE NUMBER OF DESCRIPTORS USED

Method	NoD	Accuracy
CSML+SVM, aligned [21]	6	88.00 ± 0.37
PAF [14]	1	87.77 ± 0.51
STFRD+PMML [13]	8	89.35 ± 0.50
Sub-SML [58]	6	89.73 ± 0.38
VMRS [59]	10	91.10 ± 0.59
DDML (HOG: sqrt)	1	88.38 ± 0.77
DDML (HDLBP: sqrt)	1	92.55 ± 0.36
DDMML	6	$\mathbf{93.28 \pm 0.39}$

TABLE V

COMPARISONS OF THE MEAN ACCURACY (%) WITH THE STATE-OF-THE-ART RESULTS ON LFW UNDER THE IMAGE UNRESTRICTED SETTING, WHERE NOD DENOTES THE NUMBER OF DESCRIPTORS USED

Method	NoD	Accuracy
Combined PLDA [60]	3	90.07 ± 0.51
Combined Joint Bayesian [61]	4	90.90 ± 1.48
Sub-SML [58]	6	90.75 ± 0.64
VMRS [59]	10	92.05 ± 0.45
Fisher vector faces [15]	1	93.03 ± 1.05
High-dim LBP [54]	1	93.18 ± 1.07
DDML (HDLBP: sqrt)	1	93.70 ± 0.37
DDMML	6	$\mathbf{94.50 \pm 0.35}$



(a) (b)

Fig. 4. ROC comparison between our methods and the state-of-the-art face verification methods on LFW under the (a) image restricted and (b) image unrestricted settings, respectively.

3) Comparison With Deep Learning Methods: We compare our DDML and DDMML with several recently proposed deep learning based face verification methods: CDBN [12], DNLML-ISA [32], ConvNet-RBM [45], DeepFace [47], and DeepID2 [62]. Table VI shows the verification performance of different deep learning-based face verification methods on LFW. We see that our DDML and DDMML outperform the other deep learning methods without outside data. The reason is that CDBN is a unsupervised deep learning method and our methods are supervised, so that more discriminative information can be exploited in our DDML and DDMML methods. Compared with DNLML-ISA which uses a stacked architecture, our DDML and DDMML adopt a fully-connection architecture to design the network, which can explore better hierarchical information due to the back-propagation strategy. Compared with ConvNet-RBM, our methods employ a large margin strategy to train the model, which is more effective than the RBM model used in ConvNet-RBM. Since there are

TABLE VI

PERFORMANCE COMPARISONS (%) WITH EXISTING DEEP LEARNING-BASED FACE VERIFICATION METHODS ON LFW

Method	Setting	Accuracy	Outside data
CDBN [12]	restricted	86.88 ± 0.62	No
CDBN+Hand-crafted [12]	restricted	87.77 ± 0.62	No
DNLML-ISA, combined [32]	unrestricted	92.28 ± 0.42	No
ConvNet-RBM [45]	unrestricted	91.75	No
ConvNet-RBM [45]	unrestricted	92.52	Yes ($>87.6K$ outside images)
DeepFace [47]	restricted	97.15 ± 0.84	Yes ($4.4M$ outside images)
DeepFace [47]	unrestricted	97.25 ± 0.81	Yes ($4.4M$ outside images)
DeepID2 [62]	unrestricted	$\mathbf{99.15 \pm 0.13}$	Yes ($0.2M$ outside images)
DDMML	restricted	93.28 ± 0.39	No
DDMML	unrestricted	94.50 ± 0.35	No

TABLE VII

COMPARISON WITH DEEPFACE AND DEEPID2 ON LFW DATASET UNDER THE UNRESTRICTED WITH LABELED OUTSIDE DATA SETTING

Method	Feature	Accuracy (%)
DDML	CNN	98.23 ± 0.22
DDMML	CNN, SSIFT, HOG, HDLBP	98.27 ± 0.20
DeepFace [47]	CNN	97.35 ± 0.25
DeepID2 [62]	CNN	$\mathbf{99.15 \pm 0.13}$

TABLE VIII

COMPARISONS OF THE MEAN ACCURACY (%) WITH THE SHALLOW METRIC LEARNING METHODS ON YTF UNDER THE IMAGE RESTRICTED SETTING

Feature	DDML	DSML
CSLBP	75.98 ± 0.89	73.26 ± 0.99
FPLBP	76.60 ± 1.71	73.46 ± 1.66
LBP	81.26 ± 1.63	78.14 ± 0.94
Feature	DDMML	DSMML
All	$\mathbf{82.54 \pm 1.58}$	79.38 ± 1.36

4.4M outside face images used in DeepFace, DeepFace can learn more discriminative and robust features for verification. However, our DDMML achieves the state-of-the-art performance on LFW when no outside data is used.

4) Comparison With DeepFace and DeepID2 on the LFW Dataset Under the Unrestricted With Labeled Outside Data Setting: We evaluated our DDML and DDMML methods using convolutional neural network (CNN) feature for face verification [47], [62], [63]. In our implementations, we employed the VGG-Face CNN model provided by [63] to compute the CNN feature descriptor. Specifically, for each face image in the LFW dataset, we first cropped a 150 × 150 region from its center, and then resized it into 224 × 224 to compute a 4096-dimensional CNN feature vector with the network. Moreover, each feature vector is reduced to the size of 300 by PCA for verification. Table VII shows the mean verification accuracy on the LFW dataset under the unrestricted with labeled outside data setting, respectively. We see that our DDML and DDMML are comparable to DeepFace [47], and DeepID2 [62] obtains the best accuracy on the LFW dataset under the unrestricted with labeled outside data setting.

C. Experimental Comparison on YTF

1) Deep vs. Shallow Metric Learning: We compare our methods with DSML and DSMML methods on the YTF dataset. Table VIII shows the verification rates with standard error when different feature descriptors are compared. Our DDML and DDMML consistently outperform DSML and DSMML in terms of the mean verification rate on YTF.

2) Comparison With the State-of-the-Art methods: We compare our methods with the state-of-the-art face verification methods on the YTF dataset. These compared methods include Matched Background Similarity (MBGS) [16], APEM [64], STFRD+PMML [13], MBGS+SVM⊖ [17], VSOF+OSS

TABLE IX

COMPARISONS OF THE MEAN ACCURACY (%) WITH THE STATE-OF-THE-ART FACE VERIFICATION METHODS ON YTF UNDER THE IMAGE RESTRICTED SETTING

Method	Accuracy
MBGS (LBP) [16]	76.40 ± 1.80
APEM (LBP) [64]	77.44 ± 1.46
APEM (fusion) [64]	79.06 ± 1.51
STFRD+PMML [13]	79.48 ± 2.52
MBGS+SVM⊖ (LBP) [17]	78.90 ± 1.90
VSOF+OSS (Adaboost) [65]	79.70 ± 1.80
PHL+SILD (LBP) [56]	80.20 ± 1.30
DDML (LBP)	81.26 ± 1.63
DDMML	$\mathbf{82.54 \pm 1.58}$

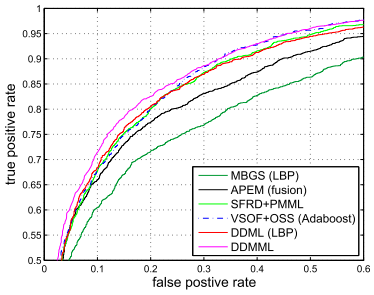


Fig. 5. ROC curves comparison of different methods on YTF under image restricted setting.

TABLE X

COMPARISON WITH CNN BASED METHODS ON THE YTF DATASET UNDER IMAGE UNRESTRICTED SETTING

Method	Feature	Accuracy (%)
DDML	CNN	94.58 ± 1.01
DDMML	CNN, CSLBP, FPLBP, LBP	$\mathbf{94.80 \pm 1.01}$
DeepFace [47]	CNN	91.4 ± 1.1
DeepID2+ [66]	CNN	93.2 ± 0.2

TABLE XI

PERFORMANCE COMPARISONS (%) WITH THE SHALLOW METRIC LEARNING METHODS ON KinFaceW-I AND KinFaceW-II DATASETS UNDER IMAGE-RESTRICTED SETTING

Method	Feature	KinFaceW-I					KinFaceW-II				
		F-S	F-D	M-S	M-D	Mean	F-S	F-D	M-S	M-D	Mean
DSML	LBP	70.8	67.2	72.5	74.0	71.1	72.4	64.3	67.6	71.2	68.9
DSML	DSIFT	70.0	70.9	73.9	78.1	73.2	75.6	63.8	70.0	74.7	71.0
DSML	HOG	73.9	69.1	70.8	76.9	72.7	74.9	66.5	73.1	73.4	72.0
DSML	LPQ	78.3	72.6	75.1	80.5	76.6	80.0	75.2	76.4	78.3	77.5
DSMML	All	80.4	75.5	77.6	82.1	78.9	83.2	76.0	79.0	81.0	79.8
DDML	LBP	78.4	71.9	75.8	75.8	75.5	81.4	73.8	78.1	77.2	77.6
DDML	DSIFT	78.0	75.9	76.5	83.3	78.4	82.5	75.7	79.1	79.2	79.1
DDML	HOG	80.5	72.8	75.4	81.2	77.5	80.9	75.7	78.8	77.0	78.1
DDML	LPQ	83.8	77.0	78.1	86.6	81.4	84.8	82.6	79.4	81.8	82.2
DDMML	All	**86.4**	**79.1**	**81.4**	**87.0**	**83.5**	**87.4**	**83.8**	**83.2**	**83.0**	**84.3**

TABLE XII

COMPARISONS OF THE MEAN VERIFICATION ACCURACY (%) ON THE TSKinFace DATASET

Method	Feature	FS	FD	MS	MD	FM-S	FM-D
DSML	LBP	74.5	69.9	71.1	71.0	75.8	73.9
DSML	DSIFT	75.2	71.5	71.9	72.3	76.3	75.8
DSML	HOG	75.8	70.0	70.3	71.6	77.1	76.0
DSML	LPQ	76.1	73.2	74.5	73.8	78.6	78.2
DSMML	All	79.3	76.5	76.2	77.1	82.0	81.6
DDML	LBP	76.0	73.8	75.7	76.3	80.2	79.5
DDML	DSIFT	78.6	75.1	75.2	76.8	81.5	81.0
DDML	HOG	79.2	75.0	76.9	76.1	81.9	81.3
DDML	LPQ	81.0	78.3	80.8	79.6	83.7	83.2
DDMML	All	86.6	82.5	83.2	84.3	88.5	87.1
RSBM-block-FS [53]	–	83.0	80.5	82.8	81.1	86.4	84.4
GMP [67]	–	**88.5**	**87.0**	**87.9**	**87.8**	**90.6**	**89.0**

(Adaboost) [65], and PHL+SILD [56]. Table IX and Fig. 5 show the mean verification rate with the standard error and ROC curves of DDML and DDMML and the state-of-the-art methods on the YTF dataset under image restricted setting, respectively. We see that the performance of our DDML with the LBP feature is 81.26 ± 1.63, which improves the PHL+SILD method by 1.0% in the gain of the mean verification rate. Moreover, the gain can be further improved 1.28% when DDMML is used.

3) Comparison With DeepFace and DeepID2+ [66] on YTF Dataset Under the Image Unrestricted Setting: Following the same VGG-Face CNN model [63] as used in the LFW dataset, we only extracted CNN features on the first 100 frames of each video at a single scale. For each face image in the YTF dataset, we first resized it to size of 200×200 pixels and cropped a 100×100 region from its center. Then, we resized it into 224×224 to compute a 4096-dimensional CNN feature vector. Finally, we average these CNN feature vectors of the first 100 frames for each video, and each video is represented by a 4096-dimensional feature vector. Moreover, each feature vector is reduced to the size of 200 by PCA before verification. Table X shows the mean verification accuracy of our proposed methods and several CNN based methods (e.g., DeepFace [47], DeepID2+ [66]) on the YTF dataset under the image unrestricted setting. We see that the performance of our DDML and DDMML methods are comparable to the current state-of-the-art results under the same setting.

D. Experimental Results on KinFaceW-I and KinFaceW-II

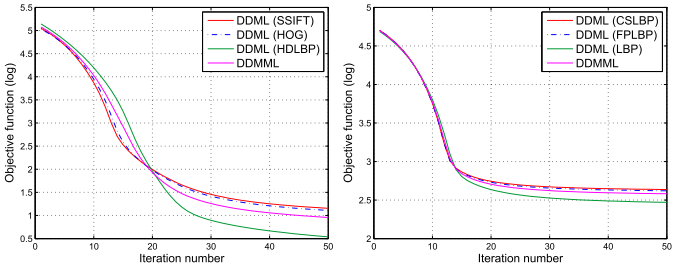
1) Deep vs. Shallow Metric Learning: We compare our methods with the DSML and DSMML methods on the KinFaceW-I and KinFaceW-II datasets. Table XI shows the mean verification rates when different feature descriptors are compared. As can be seen, our DDML and DDMML consistently outperform DSML and DSMML in terms of the mean verification rate on these two datasets.

E. Experimental Results on TSKinFace

The TSKinFace [53] dataset contains 513 Father-Mother-Son (FM-S) groups and 502 Father-Mother-Daughter (FM-D) groups, respectively. Following the same setting in [53], we adopted the cropped face images (i.e., 64×64 pixels) provided by the dataset creators and extracted LBP, DSIFT, HOG and LPQ features for each image. Then we employed the 5-fold cross-validation strategy to conduct kinship verification experiments with the same number of positive and negative groups for evaluation. Table XII lists the mean verification accuracy of our methods and two current state-of-the-art methods on the TSKinFace. We see that our DDMML method obtains the second best results for all relationships, and our methods are comparable to other methods on this dataset.

F. Effect of the Activation Function

We analyze the effect of the activation function in our DDML method. We compare the *tanh* function with two other popular activation functions: *sigmoid* and non-saturating



(a) (b)

Fig. 6. Convergence curves of DDML and DDMML on the (a) LFW and (b) YTF datasets, respectively.

sigmoid (*ns-sigmoid*).[1] Table XIII lists the performance of our DDML method with different activation functions on the LFW and YTF datasets, where the HDLBP (sqrt) and LBP features are used, respectively. We see that the *tanh* function performs the best and the *sigmoid* function performs the worst. The reason is that the output of the *tanh* activation function is symmetrical, where the output of the standard sigmoid activation function is asymmetrical. The asymmetry in the *sigmoid* function introduces a bias that pushes the activations of the later layers towards saturation, which is not a good position to obtain good training.

G. Convergence Analysis

We have evaluated the convergence of our DDML and DDMML methods. Fig. 6 plots the value of the objective function of DDML and DDMML versus different number of iterations on the LFW and YTF datasets under image restricted setting. In our experiments, the objective function values of DDML and DDMML are the sum of all 5400 training pairs on LFW and 4500 pairs on YTF, respectively. We see that the proposed DDML and DDMML methods converge in $30 \sim 40$ iterations on these two datasets.

H. Discussion

While deep features have achieved state-of-the-art performance on various visual applications, their performance can still be improved by combining them with conventional metric leaning or deep metric learning methods [63]. Table XIV shows performance comparison of deep features without and with metric learning on LFW and YTF datasets, where the results are directly taken from [63]. In SoftMax (L_2) [63], softmax log-loss is used to learn VGG-Face CNN feature [63], and Euclidean distance (L_2) is used to compare CNN features. In Embedding loss [63], a triplet loss training scheme (i.e., metric learning method) is used to tune the VGG-Face CNN model and learn a discriminative metric for face verification. From this Table, we can see that CNN feature with metric learning significantly improves the performance of CNN feature without metric learning scheme on both LFW and YTF datasets. More details of VGG-Face CNN feature, SoftMax (L_2), and Embedding loss can be found in [63].

[1]The sigmoid function is defined as $s(z) = 1/(1+e^{-z})$, and the ns-sigmoid function is defined as $z = s^3(z)/3 + s(z)$.

TABLE XIII

COMPARISONS (%) OF DIFFERENT ACTIVATION FUNCTIONS ON LFW AND YTF UNDER THE IMAGE RESTRICTED SETTING

Dataset	sigmoid	ns-sigmoid	tanh
LFW	81.93 ± 0.46	92.10 ± 0.43	$\mathbf{92.55 \pm 0.36}$
YTF	70.20 ± 1.26	80.78 ± 1.15	$\mathbf{81.26 \pm 1.63}$

TABLE XIV

THE EQUAL ERROR RATE (EER) (%) OF CNN FEATURE WITHOUT AND WITH METRIC LEARNING SCHEME ON LFW AND YTF DATASETS UNDER THE UNRESTRICTED WITH LABELED OUTSIDE DATA SETTING

Method	Feature Dims.	LFW	YTF
SoftMax (L_2) [63]	4096	97.27	92.8
Embedding loss [63]	1024	99.13	97.4

TABLE XV

THE MEAN VERIFICATION ACCURACY (%) OF DDML AND L_2 WITH CNN FEATURE ON LFW AND YTF DATASETS UNDER THE UNRESTRICTED WITH LABELED OUTSIDE DATA SETTING

Method	LFW	YTF
L_2	96.57 ± 0.26	90.33 ± 1.10
DDML	98.23 ± 0.22	94.58 ± 1.01

We also evaluated DDML and L_2 using CNN feature for face verification. In experiments, we employed the VGG-Face CNN model provided by [63] to compute the CNN features. Table XV shows the mean verification accuracy of DDML and L_2 using CNN feature on both LFW and YTF datasets under the unrestricted setting. We also see that DDML with CNN feature significantly improves the performance of CNN feature on two datasets. These results also show that it is necessary to learn a distance metric for deep features.

V. CONCLUSION

We have proposed a discriminative deep metric learning (DDML) method for face and kinship verification. To better use multiple features for face and kinship verification, we have also proposed a discriminative deep multi-metric learning (DDMML) method to extract the common information of multiple features to improve the verification performance. Our methods achieve the competitive or acceptable face and kinship verification performance on the widely used LFW, YTF, KinFaceW-I, KinFaceW-II, and TSKinFace datasets. How to apply our DDML and DDMML to other visual applications such as image classification, human activity recognition and visual tracking [68] is a proposing direction of our future work.

REFERENCES

[1] T. Ahonen, A. Hadid, and M. Pietikäinen, "Face description with local binary patterns: Application to face recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 12, pp. 2037–2041, Dec. 2006.

[2] J. Lu, Y.-P. Tan, and G. Wang, "Discriminative multimanifold analysis for face recognition from a single training sample per person," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 1, pp. 39–51, Jan. 2013.

[3] R. Fang, K. D. Tang, N. Snavely, and T. Chen, "Towards computational models of kinship verification," in *Proc. Int. Conf. Image Process.*, Sep. 2010, pp. 1577–1580.

[4] S. Xia, M. Shao, J. Luo, and Y. Fu, "Understanding kin relationships in a photo," *IEEE Trans. Multimedia*, vol. 14, no. 4, pp. 1046–1056, Aug. 2012.

[5] J. Lu, J. Hu, X. Zhou, Y. Shang, Y.-P. Tan, and G. Wang, "Neighborhood repulsed metric learning for kinship verification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 2594–2601.

[6] J. Lu, X. Zhou, Y.-P. Tan, Y. Shang, and J. Zhou, "Neighborhood repulsed metric learning for kinship verification," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 2, pp. 331–345, Feb. 2014.

[7] J. Hu, J. Lu, and Y.-P. Tan, "Discriminative deep metric learning for face verification in the wild," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 1875–1882.

[8] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller, "Labeled faces in the wild: A database for studying face recognition in unconstrained environments," Dept. Comput. Sci., Univ. Massachusetts Amherst, Amherst, MA, USA, Tech. Rep. 07-49, 2007.

[9] L. Wolf, T. Hassner, and Y. Taigman, "Descriptor based methods in the wild," in *Proc. Real-Life Images Workshop Eur. Conf. Comput. Vis. (ECCV)*, 2008, pp. 1–14.

[10] M. Guillaumin, J. Verbeek, and C. Schmid, "Is that you? Metric learning approaches for face identification," in *Proc. IEEE Int. Conf. Comput. Vis.*, Sep. 2009, pp. 498–505.

[11] L. Wolf, T. Hassner, and Y. Taigman, "Similarity scores based on background samples," in *Proc. Asian Conf. Comput. Vis.*, 2009, pp. 88–97.

[12] G. B. Huang, H. Lee, and E. G. Learned-Miller, "Learning hierarchical representations for face verification with convolutional deep belief networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 2518–2525.

[13] Z. Cui, W. Li, D. Xu, S. Shan, and X. Chen, "Fusing robust face region descriptors via multiple metric learning for face recognition in the wild," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 3554–3561.

[14] D. Yi, Z. Lei, and S. Z. Li, "Towards pose robust face recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 3539–3545.

[15] K. Simonyan, O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Fisher vector faces in the wild," in *Proc. Brit. Mach. Vis. Conf.*, 2013, pp. 1–12.

[16] L. Wolf, T. Hassner, and I. Maoz, "Face recognition in unconstrained videos with matched background similarity," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2011, pp. 529–534.

[17] L. Wolf and N. Levy, "The SVM-minus similarity score for video face recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 3523–3530.

[18] J. Hu, J. Lu, Y.-P. Tan, and J. Zhou, "Deep transfer metric learning," *IEEE Trans. Image Process.*, vol. 25, no. 12, pp. 5576–5588, Dec. 2016.

[19] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.

[20] Z. Lei, M. Pietikäinen, and S. Z. Li, "Learning discriminant face descriptor," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 2, pp. 289–302, Feb. 2014.

[21] H. V. Nguyen and L. Bai, "Cosine similarity metric learning for face verification," in *Proc. Asian Conf. Comput. Vis.*, 2010, pp. 709–720.

[22] X. Zhou, J. Hu, J. Lu, Y. Shang, and Y. Guan, "Kinship verification from facial images under uncontrolled conditions," in *Proc. ACM Int. Conf. Multimedia*, 2011, pp. 953–956.

[23] X. Zhou, J. Lu, J. Hu, and Y. Shang, "Gabor-based gradient orientation pyramid for kinship verification under uncontrolled environments," in *Proc. ACM Int. Conf. Multimedia*, 2012, pp. 725–728.

[24] G. Somanath and C. Kambhamettu, "Can faces verify blood-relations?" in *Proc. IEEE Int. Conf. Biometrics, Theory, Appl. Syst.*, Sep. 2012, pp. 105–112.

[25] H. Dibeklioglu, A. A. Salah, and T. Gevers, "Like father, like son: Facial expression dynamics for kinship verification," in *Proc. IEEE Int. Conf. Comput. Vis.*, Jun. 2013, pp. 1497–1504.

[26] Y. Guo, H. Dibeklioglu, and L. van der Maaten, "Graph-based kinship recognition," in *Proc. Int. Conf. Pattern Recognit.*, Aug. 2014, pp. 4287–4292.

[27] G. Guo and X. Wang, "Kinship measurement on salient facial features," *IEEE Trans. Instrum. Meas.*, vol. 61, no. 8, pp. 2322–2325, Aug. 2012.

[28] N. Kohli, R. Singh, and M. Vatsa, "Self-similarity representation of weber faces for kinship classification," in *Proc. IEEE Int. Conf. Biometrics, Theory, Appl. Syst.*, Sep. 2012, pp. 245–250.

[29] J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon, "Information-theoretic metric learning," in *Proc. ICML*, 2007, pp. 209–216.

[30] M. Köstinger, M. Hirzer, P. Wohlhart, P. M. Roth, and H. Bischof, "Large scale metric learning from equivalence constraints," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 2288–2295.

[31] A. Mignon and F. Jurie, "PCCA: A new approach for distance learning from sparse pairwise constraints," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 2666–2672.

[32] X. Cai, C. Wang, B. Xiao, X. Chen, and J. Zhou, "Deep nonlinear metric learning with independent subspace analysis for face verification," in *Proc. 20th ACM Int. Conf. Multimedia*, 2012, pp. 749–752.

[33] K. Q. Weinberger, J. Blitzer, and L. K. Saul, "Distance metric learning for large margin nearest neighbor classification," in *Proc. Adv. Neural Inf. Process. Syst.*, 2005, pp. 1473–1480.

[34] D.-Y. Yeung and H. Chang, "A kernel approach for semisupervised metric learning," *IEEE Trans. Neural Netw.*, vol. 18, no. 1, pp. 141–149, Jan. 2007.

[35] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[36] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, 2006.

[37] Y. Bengio, "Learning deep architectures for AI," *Found. Trends Mach. Learn.*, vol. 2, no. 1, pp. 1–127, 2009.

[38] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," in *Proc. Int. Conf. Mach. Learn.*, 2009, pp. 609–616.

[39] G. W. Taylor, R. Fergus, Y. LeCun, and C. Bregler, "Convolutional learning of spatio-temporal features," in *Proc. Eur. Conf. Comput. Vis.*, 2010, pp. 140–153.

[40] Q. V. Le, W. Y. Zou, S. Y. Yeung, and A. Y. Ng, "Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2011, pp. 3361–3368.

[41] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[42] S. Ji, W. Xu, M. Yang, and K. Yu, "3D convolutional neural networks for human action recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 1, pp. 221–231, Jan. 2013.

[43] Y. Zhang, M. Shao, E. K. Wong, and Y. Fu, "Random faces guided sparse many-to-one encoder for pose-invariant face recognition," in *Proc. IEEE Int. Conf. Comput. Vis.*, Jun. 2013, pp. 2416–2423.

[44] M. Kan, S. Shan, H. Chang, and X. Chen, "Stacked progressive auto-encoders (SPAE) for face recognition across poses," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 1883–1890.

[45] Y. Sun, X. Wang, and X. Tang, "Hybrid deep learning for face verification," in *Proc. IEEE Int. Conf. Comput. Vis.*, Jun. 2013, pp. 1489–1496.

[46] Z. Zhu, P. Luo, X. Wang, and X. Tang, "Deep learning identity-preserving face space," in *Proc. IEEE Int. Conf. Comput. Vis.*, Jun. 2013, pp. 113–120.

[47] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "DeepFace: Closing the gap to human-level performance in face verification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 1701–1708.

[48] Y. Fu, L. Cao, G. Guo, and T. S. Huang, "Multiple feature fusion by subspace learning," in *Proc. ACM Int. Conf. Image Video Retr.*, 2008, pp. 127–134.

[49] M. Borga. (2001). *Canonical Correlation: A Tutorial*. [Online]. Available: http://people.imt.liu.se/magnus/cca

[50] M. B. Blaschko and C. H. Lampert, "Correlational spectral clustering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2008, pp. 1–8.

[51] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proc. Int. Conf. Mach. Learn.*, 2008, pp. 1096–1103.

[52] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2010, pp. 249–256.

[53] X. Qin, X. Tan, and S. Chen, "Tri-subject kinship verification: Understanding the core of A family," *IEEE Trans. Multimedia*, vol. 17, no. 10, pp. 1855–1867, Oct. 2015.

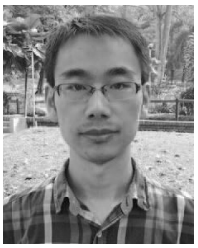
[54] D. Chen, X. Cao, F. Wen, and J. Sun, "Blessing of dimensionality: High-dimensional feature and its efficient compression for face verification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 3025–3032.

[55] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2005, pp. 886–893.

[56] M. Kan, D. Xu, S. Shan, W. Li, and X. Chen, "Learning prototype hyperplanes for face verification in the wild," *IEEE Trans. Image Process.*, vol. 22, no. 8, pp. 3310–3316, Aug. 2013.

[57] T. Ahonen, E. Rahtu, V. Ojansivu, and J. Heikkilä, "Recognition of blurred faces using local phase quantization," in *Proc. Int. Conf. Pattern Recognit.*, Dec. 2008, pp. 1–4.
[58] Q. Cao, Y. Ying, and P. Li, "Similarity metric learning for face recognition," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 2408–2415.
[59] O. Barkan, J. Weill, L. Wolf, and H. Aronowitz, "Fast high dimensional vector multiplication face recognition," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 1960–1967.
[60] P. Li, Y. Fu, U. Mohammed, J. H. Elder, and S. J. D. Prince, "Probabilistic models for inference about identity," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 1, pp. 144–157, Jan. 2012.
[61] D. Chen, X. Cao, L. Wang, F. Wen, and J. Sun, "Bayesian face revisited: A joint formulation," in *Proc. Eur. Conf. Comput. Vis.*, 2012, pp. 566–579.
[62] Y. Sun, Y. Chen, X. Wang, and X. Tang, "Deep learning face representation by joint identification-verification," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 1988–1996.
[63] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition," in *Proc. Brit. Mach. Vis. Conf.*, 2015, pp. 41.1–41.12.
[64] H. Li, G. Hua, Z. Lin, J. Brandt, and J. Yang, "Probabilistic elastic matching for pose variant face verification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 3499–3506.
[65] H. Méndez-Vázquez, Y. Martínez-Díaz, and Z. Chai, "Volume structured ordinal features with background similarity measure for video face recognition," in *Proc. Int. Conf. Biometrics*, Jun. 2013, pp. 1–6.
[66] Y. Sun, X. Wang, and X. Tang, "Deeply learned face representations are sparse, selective, and robust," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 2892–2900.
[67] Z. Zhang, Y. Chen, and V. Saligrama, "Group membership prediction," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 3916–3924.
[68] J. Hu, J. Lu, and Y.-P. Tan, "Deep metric learning for visual tracking," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 11, pp. 2056–2068, Nov. 2016.

Jiwen Lu (S'10–M'11–SM'15) received the B.Eng. degree in mechanical engineering and the M.Eng. degree in electrical engineering from the Xi'an University of Technology, Xi'an, China, in 2003 and 2006, respectively, and the Ph.D. degree in electrical engineering from Nanyang Technological University, Singapore, in 2012. From 2011 to 2015, he was a Research Scientist with the Advanced Digital Sciences Center, Singapore. He is currently an Associate Professor with the Department of Automation, Tsinghua University, Beijing, China. His current research interests include computer vision, pattern recognition, and machine learning. He has authored/co-authored over 150 scientific papers in these areas, where 46 were the IEEE transactions papers. He was a recipient of the National 1000 Young Talents Plan Program in 2015. He serves/has served as an Associate Editor of the *Pattern Recognition Letters*, *Neurocomputing*, and the IEEE ACCESS, a Managing Guest Editor of the *Pattern Recognition and Image and Vision Computing*, a Guest Editor of the *Computer Vision and Image Understanding*, and an elected member of the Information Forensics and Security Technical Committee of the IEEE Signal Processing Society. He is/was a Workshop Chair/Special Session Chair/Area Chair for over ten international conferences.



Junlin Hu received the B.Eng. degree from the Xi'an University of Technology, Xi'an, China, in 2008, and the M.Eng. degree from Beijing Normal University, Beijing, China, in 2012. He is currently pursuing the Ph.D. degree with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore. His research interests include computer vision, pattern recognition, and biometrics.

Yap-Peng Tan (S'95–M'97–SM'04) received the B.S. degree from National Taiwan University, Taipei, Taiwan, in 1993, and the M.A. and Ph.D. degrees from Princeton University, Princeton, NJ, in 1995 and 1997, respectively, all in electrical engineering. From 1997 to 1999, he was with Intel Corporation, Chandler, AZ, and Sharp Laboratories of America, Camas, WA. In 1999, he joined the Nanyang Technological University of Singapore, where he is currently an Associate Professor and an Associate Chair (Academic) of the School of Electrical and Electronic Engineering. His current research interests include image and video processing, content-based multimedia analysis, computer vision, pattern recognition, and data analytics. He served as the Chair of the Visual Signal Processing and Communications Technical Committee of the IEEE Circuits and Systems Society from 2012 to 2014, and the Chairman of the IEEE Signal Processing Singapore Chapter from 2009 to 2010. He was the Finance Chair of ICIP'2004, the General Co-Chair of ICME'2010, the Technical Program Co-Chair of ICME'2015, and the General Co-Chair of VCIP'2015. He is the Tutorial Co-Chair of ICME'2016 and the Technical Program Co-Chair of ICIP'2019. He has also served as an Associate Editor of the IEEE SIGNAL PROCESSING LETTERS, the IEEE TRANSACTIONS ON MULTIMEDIA, and the IEEE ACCESS, an Editorial Board Member of the *EURASIP Journal on Advances in Signal Processing* and the *EURASIP Journal on Image and Video Processing*, a Guest Editor for special issues of several journals, including the IEEE TRANSACTIONS ON MULTIMEDIA.