Discriminative Deep Metric Learning for Face Verification in the Wild

Junlin Hu[1], Jiwen Lu[2]*, Yap-Peng Tan[1]
[1]School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore
[2]Advanced Digital Sciences Center, Singapore
jhu007@e.ntu.edu.sg, jiwen.lu@adsc.com.sg, eyptan@ntu.edu.sg

Abstract

This paper presents a new discriminative deep metric learning (DDML) method for face verification in the wild. Different from existing metric learning-based face verification methods which aim to learn a Mahalanobis distance metric to maximize the inter-class variations and minimize the intra-class variations, simultaneously, the proposed DDML trains a deep neural network which learns a set of hierarchical nonlinear transformations to project face pairs into the same feature subspace, under which the distance of each positive face pair is less than a smaller threshold and that of each negative pair is higher than a larger threshold, respectively, so that discriminative information can be exploited in the deep network. Our method achieves very competitive face verification performance on the widely used LFW and YouTube Faces (YTF) datasets.

1. Introduction

Overt the past two decades, a large number of face recognition methods have been proposed in the literature [23, 41], and most of them have achieved satisfying recognition performance under controlled conditions. However, their performance drops heavily when face images are captured in the wild because large intra-class variations usually occur in this scenario. Face recognition can be mainly classified into two tasks: face identification and face verification. The former aims to recognize the person from a set of gallery face images or videos and find the most similar one to the probe sample. The latter is to determine whether a given pair of face images or videos is from the same person or not. In this paper, we consider the second one where face images contain significant variations caused by varying lighting, expression, pose, resolution, and background.

Recently, many approaches have been proposed to improve the face verification performance in unconstrained environments [6, 9, 13, 28, 34, 37], and these methods can be roughly divided into two categories: feature descriptor-based and metric learning-based. For the first category, a robust and discriminative descriptor is usually employed to represent each face image as a compact feature vector, where different persons are expected to be separated as much as possible in the feature space. Typical face feature descriptors include SIFT [22], LBP [1], probabilistic elastic matching (PEM) [21], and fisher vector faces [28]. For the second category, a new distance metric is usually learned from the labeled training samples to effectively measure the similarity of face samples, under which the similarity of positive pairs is enlarged and that of negative

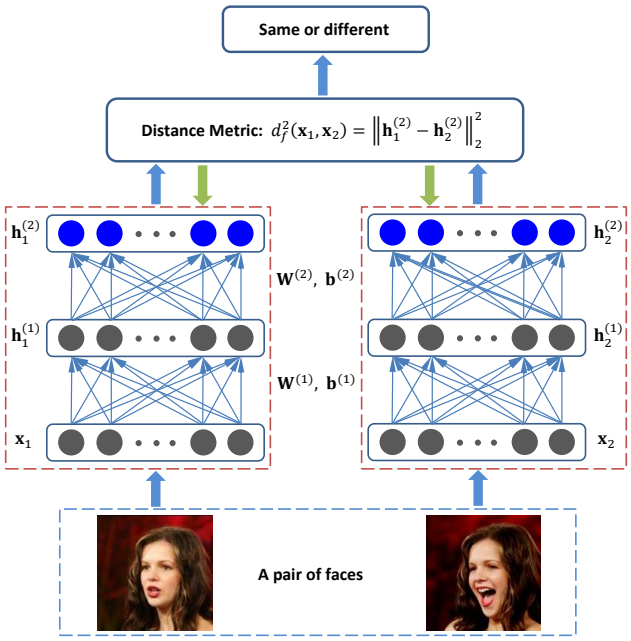

Figure 1. The flowchart of proposed DDML method for face verification. For a given pair of face images $\mathbf{x}_1$ and $\mathbf{x}_2$, we map them into the same feature subspace as $\mathbf{h}_1^{(2)}$ and $\mathbf{h}_2^{(2)}$ by using a set of hierarchical nonlinear transformations, where the similarity of their outputs at the most top level is computed and used to determine whether the face pair is from the same person or not.

*Corresponding author.

pairs is reduced as much as possible. Representative metric learning algorithms include logistic discriminant metric learning (LDML) [9], cosine similarity metric learning (CSML) [26], pairwise constrained component analysis (PCCA) [25], and pairwise-constrained multiple metric learning (PMML) [6].

In this paper, we contribute to the second category and propose a new discriminative deep metric learning (DDML) method for face verification in the wild, where the basic idea of our method is illustrated in Figure 1. Unlike most existing metric learning methods, our DDML builds a deep neural network which learns a set of hierarchical nonlinear transformations to project face pairs into other feature subspace, under which the distance of each positive face pair is less than a smaller threshold and that of each negative pair is higher than a larger threshold, respectively, so that discriminative information is exploited for the verification task. Experimental results on the widely used LFW and YouTube Faces (YTF) datasets are presented to show the effectiveness of the proposed method.

2. Related Work

Metric Learning: Many metric learning algorithms have been proposed over the past decade, and some of them have been successfully applied to address the problem of face verification in the wild [3, 6, 7, 9, 29]. The common objective of these methods is to learn a good distance metric so that the distance between positive face pairs is reduced and that of negative pairs is enlarged as much as possible. However, most existing metric learning methods only learn a linear transformation to map face samples into a new feature space, which may not be powerful enough to capture the nonlinear manifold where face images usually lie on. To address this limitation, the kernel trick is usually adopted to first map face samples into a high-dimensional feature space and then learn a discriminative distance metric in the high-dimensional space [31, 38]. However, these methods cannot explicitly obtain the nonlinear mapping functions, which usually suffer from the scalability problem. Different from these metric learning methods, our proposed DDML learns a set of hierarchical nonlinear transformations to project face pairs into one feature space in a deep architecture, where the nonlinear mappings are explicitly obtained. We also achieve the very competitive performance on the face verification in the wild problem with two existing publicly available datasets.

Deep Learning: In recent years, deep learning has received increasing interests in computer vision and machine learning, and a number of deep learning methods have been proposed in the literature [2, 10, 11, 13, 15, 18, 19, 20, 27, 30]. Generally, deep learning aims to learn hierarchical feature representations by building high-level features from low-level ones. Existing deep learning methods can be mainly categorized three classes: unsupervised, supervised and semi-supervised, and they have been successfully applied to many visual analysis applications such as object recognition [27], human action recognition [15, 18], and face verification [13]. While many attempts have been made on deep learning in feature engineering such as deep belief network [10], stacked auto-encoder [18], and convolutional neural networks [15], little progress has been made in metric learning with a deep architecture. More recently, Cai *et al.* [3] proposed a nonlinear metric learning method by combining the logistic regression and stacked independent subspace analysis. Differently, our proposed DDML method employs a network to learn the nonlinear distance metric where the back propagation algorithm can be used to train the model. Hence, our method is complementary to existing deep learning methods.

3. Proposed Approach

In this section, we first briefly review the conventional Mahalanobis distance metric learning, and then present the proposed DDML method, as well as its implementation details.

3.1. 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. 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 $\mathbf{x}_j$ can be 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 should have the properties of nonnegativity, symmetry, and triangle inequality. Hence, $\mathbf{M}$ is symmetric and positive semi-definite, and can be decomposed by as follows:

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

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

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 can see from Eq. (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 metric in the original space.

3.2. DDML

The conventional Mahalanobis distance metric learning methods [7] only seek a linear transformation, which cannot capture the nonlinear manifold where face images usually lie on, especially when face images are captured in unconstrained environments because there are usually large variations in this scenario. To address this limitation, the kernel trick is usually employed to implicitly map face samples into a high-dimensional feature space and then learn a discriminative distance metric in the high-dimensional space [31]. However, these methods cannot explicitly obtain the nonlinear mapping functions, which usually suffer from the scalability problem. Different from these previous metric learning methods, we propose a new deep metric learning method to learn hierarchical nonlinear mappings to address the nonlinear and scalability problems simultaneously.

As shown in Figure 1, we first construct a deep neural network to compute the representations of a face pair by passing them to multiple 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 of 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 $s : \mathbb{R} \mapsto \mathbb{R}$ is a nonlinear activation function which operates component-wisely, *e.g.*, 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. Similarly, 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:

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

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 distance can be measured by computing the squared Euclidean distance between the most top level representations, which is defined as follows:

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

It is desirable to exploit discriminative information for face representations of the most top level from our proposed DDML model, which is more effective to face 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. Figure 2 shows the basic idea of our proposed DDML method. Specifically, DDML aims to seek a nonlinear mapping f such that the distance $d_f^2(\mathbf{x}_i, \mathbf{x}_j)$ between $\mathbf{x}_i$ and $\mathbf{x}_j$ is smaller than a pre-specified threshold τ_1 in the transformed space if $\mathbf{x}_i$ and $\mathbf{x}_j$ are from the same subject ($\ell_{ij} = 1$), and larger than τ_2 in the transformed space if samples $\mathbf{x}_i$ and $\mathbf{x}_j$ are from different subjects ($\ell_{ij} = -1$), where the pairwise label ℓ_{ij} denotes the similarity or dissimilarity between a face pair $\mathbf{x}_i$ and $\mathbf{x}_j$, and $\tau_2 > \tau_1$.

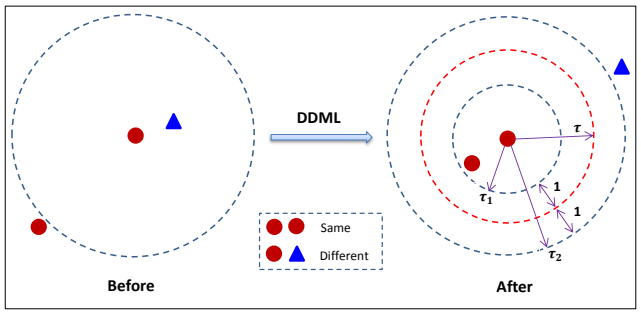

Figure 2. Intuitive illustration of the proposed DDML method. There are three face samples in the original feature space, which are used to generate two pairs of face images, where two of them form a positive pair (two circles) and two of them form the negative pair (one circle in the center and one 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 scenario is harmful to face verification because it causes an error. When our DDML method is applied, the distance of the positive pair is less than a smaller threshold τ_1 and that of the negative pair is higher than a larger threshold τ_2 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 number of parameters 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:

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

where $\tau_1 = \tau - 1$ and $\tau_2 = \tau + 1$. With this constrain, there is a margin between each positive and negative pairs in the learned feature space, as shown in Figure 2.

By applying the above constrain in Eq. (6) to each positive and negative pair in the training set, we formulate our

DDML as the following optimization problem:

$$\begin{aligned}\arg\min_f \ J &= J_1 + J_2 \\ &= \frac{1}{2}\sum_{i,j} g\Big(1-\ell_{ij}\big(\tau - d_f^2(\mathbf{x}_i,\mathbf{x}_j)\big)\Big) \\ &+ \frac{\lambda}{2}\sum_{m=1}^{M}\Big(\big\|\mathbf{W}^{(m)}\big\|_F^2 + \big\|\mathbf{b}^{(m)}\big\|_2^2\Big) \quad (7)\end{aligned}$$

where $g(z) = \frac{1}{\beta}\log\big(1+\exp(\beta z)\big)$ is the generalized logistic loss function [25], 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 $\mathbf{A}$, and λ is a regularization parameter. There are two terms J_1 and J_2 in our objective function, where J_1 defines the logistic loss and J_2 represents the regularization term, respectively.

To solve the optimization problem in Eq. (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 $\mathbf{b}^{(m)}$ can be computed as follows:

$$\begin{aligned}\frac{\partial J}{\partial \mathbf{W}^{(m)}} &= \sum_{i,j}\Big(\Delta_{ij}^{(m)}\mathbf{h}_i^{(m-1)^T} + \Delta_{ji}^{(m)}\mathbf{h}_j^{(m-1)^T}\Big) \\ &+ \lambda\,\mathbf{W}^{(m)} \quad (8)\\ \frac{\partial J}{\partial \mathbf{b}^{(m)}} &= \sum_{i,j}\Big(\Delta_{ij}^{(m)} + \Delta_{ji}^{(m)}\Big) + \lambda\,\mathbf{b}^{(m)} \quad (9)\end{aligned}$$

where $\mathbf{h}_i^{(0)} = \mathbf{x}_i$ and $\mathbf{h}_j^{(0)} = \mathbf{x}_j$, which are from the original inputs of our network. For all other layers $m = 1, 2, \cdots, M-1$, we have the following updating equations:

$$\begin{aligned}\Delta_{ij}^{(M)} &= g'(c)\ell_{ij}\Big(\mathbf{h}_i^{(M)} - \mathbf{h}_j^{(M)}\Big)\odot s'\Big(\mathbf{z}_i^{(M)}\Big) \quad (10)\\ \Delta_{ji}^{(M)} &= g'(c)\ell_{ij}\Big(\mathbf{h}_j^{(M)} - \mathbf{h}_i^{(M)}\Big)\odot s'\Big(\mathbf{z}_j^{(M)}\Big) \quad (11)\\ \Delta_{ij}^{(m)} &= \Big(\mathbf{W}^{(m+1)^T}\Delta_{ij}^{(m+1)}\Big)\odot s'\Big(\mathbf{z}_i^{(m)}\Big) \quad (12)\\ \Delta_{ji}^{(m)} &= \Big(\mathbf{W}^{(m+1)^T}\Delta_{ji}^{(m+1)}\Big)\odot s'\Big(\mathbf{z}_j^{(m)}\Big) \quad (13)\end{aligned}$$

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

$$\begin{aligned}c &\triangleq 1-\ell_{ij}\big(\tau - d_f^2(\mathbf{x}_i,\mathbf{x}_j)\big) \quad (14)\\ \mathbf{z}_i^{(m)} &\triangleq \mathbf{W}^{(m)}\mathbf{h}_i^{(m-1)} + \mathbf{b}^{(m)} \quad (15)\end{aligned}$$

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

$$\begin{aligned}\mathbf{W}^{(m)} &= \mathbf{W}^{(m)} - \mu\frac{\partial J}{\partial \mathbf{W}^{(m)}} \quad (16)\\ \mathbf{b}^{(m)} &= \mathbf{b}^{(m)} - \mu\frac{\partial J}{\partial \mathbf{b}^{(m)}} \quad (17)\end{aligned}$$

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 Eq. (20).
// Optimization by back prorogation:
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 by back propagation according to Eqs. (8) and (9).
  end
  // Back propagation
  for $m = 1, 2, \cdots, M$ **do**
    Update $\mathbf{W}^{(m)}$ and $\mathbf{b}^{(m)}$ according to Eqs. (16) and (17).
  end
  Calculate J_t using Eq (7).
  If $t > 1$ and $|J_t - J_{t-1}| < \varepsilon$, go to **Return**.
end
Return: $\{\mathbf{W}^{(m)}, \mathbf{b}^{(m)}\}_{m=1}^{M}$.

where μ is the learning rate.

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

3.3. Implementation Details

In this subsection, we detail the nonlinear activation functions and the initializations of $\mathbf{W}^{(m)}$ and $\mathbf{b}^{(m)}$, $1 \le m \le M$ in our proposed DDML method.

Activation Function: There are many nonlinear activation functions which could be used to determine the output of the nodes in our deep metric learning network. In our experiments, we use the `tanh` as the activation function because it has demonstrated better performance in our experiments. The `tanh` function and its derivative are computed as follows:

$$\begin{aligned}s(z) &= \tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}} \quad (18)\\ s'(z) &= \tanh'(z) = 1 - \tanh^2(z) \quad (19)\end{aligned}$$

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) [32] are two popular initialization methods in deep learning. In our experiments, we utilize a simple normalized random initialization method in [8], where the bias $\mathbf{b}^{(m)}$ is initialized as $\mathbf{0}$, and the weight of each layer is initialized as the following uniform distribution:

$$\mathbf{W}^{(m)} \sim U\big[-\frac{\sqrt{6}}{\sqrt{p^{(m)}+p^{(m-1)}}}, \frac{\sqrt{6}}{\sqrt{p^{(m)}+p^{(m-1)}}}\big] \quad (20)$$

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

4. Experiments

To evaluate the effectiveness of our proposed DDML method, we perform unconstrained face verification experiments on the challenging LFW [14] and YTF [34] databases. The following settings describe the details of the experiments and results.

4.1. Datasets and Experimental Settings

The LFW dataset [14] contains more than 13000 face images of 5749 subjects collected from the web with large variations in expression, pose, age, illumination, resolution, 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 use the *image restricted* setting where only the pairwise label information is required to train our method. We follow the standard evaluation protocol on the "View 2" dataset [14] 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 two types of LFW dataset for our evaluation: the LFW-a dataset[1] and "funneled" version[2]. For the LFW-a dataset, we crop each image into 80×150 to remove the background information, and then extract two features: Dense SIFT (DSIFT) [22] and LBP [1]. Regarding the "funneled" version, we use Sparse SIFT (SSIFT) descriptors provided by [9]. These three features are summarized as follows for each face image:

- DSIFT: We densely sample SIFT descriptors on each 16×16 patch without overlapping and obtain 45 SIFT descriptors. Then, we concatenate these SIFT descriptors to form a 5760-dimensional feature vector.
- LBP: We divide each image into 8×15 non-overlapping blocks, where the size of each block is 10×10. We extract a 59-dimensional uniform pattern LBP feature for each block and concatenate them to form a 7080-dimensional feature vector.
- SSIFT: The SSIFT descriptors are computed at the nine fixed landmarks with three different scales, and then they are concatenated into a 3456-dimensional feature vector [9].

As suggested in [16, 26, 36], we also use the square root of each feature and evaluate the performance of our DDML method when all the six different feature descriptors are combined. For each feature descriptor, we apply Whitened PCA (WPCA) to project it into a 500-dimensional feature vector to further remove the redundancy.

The YTF dataset [34] 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 [34] 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. Similar to LFW, we also adopt the *image restricted* protocol to evaluate our method. For this dataset, we directly use the provided three feature descriptors [34] including LBP, Center-Symmetric LBP (CSLBP) [34] and Four-Patch LBP (FPLBP) [35]. 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 400-dimensional feature vector.

For our DDML method, we train a deep network with three layers ($M = 2$), and the threshold τ, the learning rate μ and regularization parameter λ are empirically set as 3, 10^{-3}, 10^{-2} for all experiments, respectively. To further improve the verification accuracy, we further fuse multiple features in the score level. Assume there are K feature descriptors extracted for each face sample, we can get K similarity scores (or distances) by our DDML method. Then, we concatenate these cores into a K-dimensional vector, and then take the mean of this vector as the final similarity for verification. Following the standard protocol in [14, 34], we use two measures including the mean classification accuracy with standard error and the receiving operating characteristic (ROC) curve from the ten-fold cross validation to validate our method.

4.2. Experimental Comparison on LFW

Deep vs. Shadow Metric Learning: We first compare our method with the discriminative shadow metric learning (DSML) method. DSML means only one layer is considered in our model where M and the activation function are 1 and $s(z) = z$. Table 1 records the verification rate with standard error of these two methods when different feature descriptors are used. We see that our DDML consistently outperforms DSML in terms of the mean verification rate.

[1] Available: http://www.openu.ac.il/home/hassner/data/lfwa/.
[2] Available: http://vis-www.cs.umass.edu/lfw/.

Table 1. Comparison of the mean verification rate and standard error (%) with the shadow metric learning method on the LFW dataset under the image restricted setting.

Feature	DDML	DSML
DSIFT (original)	86.78 ± 2.09	83.68 ± 2.06
DSIFT (square root)	87.25 ± 1.62	84.42 ± 1.80
LBP (original)	85.47 ± 1.85	81.88 ± 1.90
LBP (square root)	87.02 ± 1.62	84.08 ± 1.21
SSIFT (original)	86.98 ± 1.37	84.02 ± 1.47
SSIFT (square root)	$\mathbf{87.83 \pm 0.93}$	84.52 ± 1.38
All features	$\mathbf{90.68 \pm 1.41}$	87.45 ± 1.45

Table 2. Comparisons of the mean verification rate and standard error (%) with the state-of-the-art results on the LFW dataset under the image restricted setting, where NoD denotes the number of descriptors used in each method.

Method	NoD	Accuracy
PCCA (SIFT) [25]	1	83.80 ± 0.40
CSML+SVM [26]	6	88.00 ± 0.37
PAF [39]	1	87.77 ± 0.51
STFRD+PMML [6]	8	89.35 ± 0.50
Fisher vector faces [28]	1	87.47 ± 1.49
DDML (SSIFT)	1	$\mathbf{87.83 \pm 0.93}$
DDML (combined)	6	$\mathbf{90.68 \pm 1.41}$

This is because DDML learns hierarchical nonlinear transformations while DSML only learns a linear transformation, so that DDML can better discover the nonlinear relationship of samples in the learned distance metric.

Comparison with the State-of-the-Art Methods: We compare our method with the state-of-the-art methods on the LFW dataset[3]. These compared methods can be categorized two classes: 1) metric learning based methods such as LDML [9], PCCA [25], CSML+SVM [26], DML-eig combined [40], and STFRD+PMML [6]; and 2) descriptor based methods such as Multiple LE+comp [4], Pose Adaptive Filter (PAF) [39], and Fisher vector faces [28]. Table 2 lists the verification rate with standard error and Figure 3 shows the ROC curves of different methods on this dataset, respectively. We clearly see that our DDML is very competitive with the state-of-the-art methods in terms of the mean verification rate under the image restricted setting.

Comparison with Existing Deep Learning Methods: We also compare our DDML with two recently proposed deep learning based face verification methods: CDBN [13] and DNLML-ISA [3]. Table 3 records the performance of different deep learning methods. We see that our DDML consistently outperforms the other two deep learning methods in terms of the mean verification rate. The reason is that CDBN is a unsupervised deep learning method and our method is supervised, such that more discriminative information can be exploited in our DDML. Compared with DNLML-ISA which uses a stacked architecture, our DDML adopts a convolutional architecture to design the network, which can explore better hierarchical information.

[3] Available: http://vis-www.cs.umass.edu/lfw/results.html.

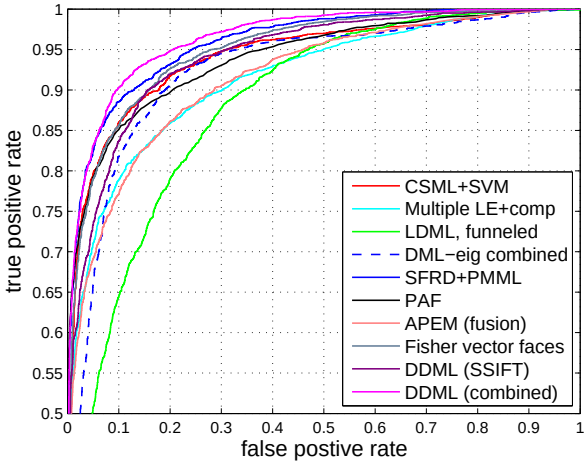

Figure 3. Comparisons of ROC curves between our DDML and the state-of-the-art methods on the LFW dataset under the image restricted setting.

Table 3. Comparisons of the mean verification rate and standard error (%) with different deep learning methods on the LFW dataset under the image restricted setting.

Method	NoD	Accuracy
CDBN [13]	6	86.88 ± 0.62
CDBN+Hand-crafted [13]	12	87.77 ± 0.62
DNLML-ISA (SSIFT) [3]	1	86.17 ± 0.40
DNLML-ISA [3]	8	88.50 ± 0.40
DDML (SSIFT)	1	$\mathbf{87.83 \pm 0.93}$
DDML (combined)	6	$\mathbf{90.68 \pm 1.41}$

4.3. Experimental Comparison on YTF

Deep vs. Shadow Metric Learning: We also compare our method with the DSML method on the YTF dataset. Table 4 records the verification rates with standard error of these two methods when different feature descriptors are compared. We see that our DDML consistently outperforms DSML in terms of the mean verification rate.

Comparison with the State-of-the-Art Methods: We compare our method with the state-of-the-art methods on the YTF dataset[4]. These compared methods include Matched Background Similarity (MBGS) [34], APEM [21], STFRD+PMML [6], MBGS+SVM⊖ [37], VSOF+OSS (Adaboost) [24], and PHL+SILD [16]. Table 5 and Figure 4 show the mean verification rate with the standard

[4] Available: http://www.cs.tau.ac.il/ wolf/ytfaces/results.html.

Table 4. Comparisons of the mean verification rate and standard error (%) with the shadow metric learning method on the YTF dataset 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	$\mathbf{81.26 \pm 1.63}$	78.14 ± 0.94
All features	$\mathbf{82.34 \pm 1.47}$	79.36 ± 1.22

Table 5. Comparisons of the mean verification rate and standard error (%) with the state-of-the-art results on the YTF dataset under the image restricted setting.

Method	Accuracy
MBGS (LBP) [34]	76.40 ± 1.80
APEM (LBP) [21]	77.44 ± 1.46
APEM (fusion) [21]	79.06 ± 1.51
STFRD+PMML [6]	79.48 ± 2.52
MBGS+SVM⊖ (LBP) [37]	79.48 ± 2.52
VSOF+OSS (Adaboost) [24]	79.70 ± 1.80
PHL+SILD (LBP) [16]	80.20 ± 1.30
DDML (LBP)	$\mathbf{81.26 \pm 1.63}$
DDML (combined)	$\mathbf{82.34 \pm 1.47}$

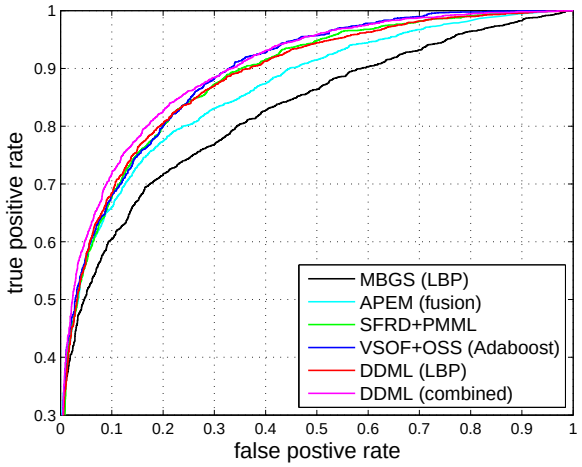

Figure 4. Comparisons of ROC curves between our work and the state-of-the-art methods on the image restricted YTF dataset.

error and ROC curves of our DDML and the state-of-the-art methods on the YTF dataset, respectively. We observe that the performance of our DDML with the LBP feature is 81.26 ± 1.63, which improves the current state-of-the-art method (PHL+SILD) by 1.0% in the gain of the mean verification rate. Moreover, the gain can be further improved 1.08% when three similarity scores are combined.

Comparison with Existing Video-based Face Recognition Methods: Lastly, we compare our DDML with existing video-based face recognition methods on the YTF dataset. These methods are Discriminant-analysis of Canonical Correlations (DCC) [17], Manifold-Manifold Distance (MMD) [33], Affine Hull based Image Set Distance (AHISD) [5], Convex Hull based Image Set Distance (CHISD) [5], and Sparse Approximated Nearest Points (SANP) [12]. Table 6 tabulates the mean verification rate with the standard error of our DDML and existing video-based face recognition methods on the YTF dataset. As seen in this table, our DDML significantly outperforms these video-based face recognition methods.

Table 6. Comparisons of the mean verification rate and standard error (%) with the existing video-based face verification methods on the YTF dataset under the image restricted setting.

Method	Accuracy
SANP [12]	63.74 ± 1.69
MMD [33]	64.96 ± 1.00
CHISD [5]	66.24 ± 1.70
AHISD [5]	66.50 ± 2.03
DCC [17]	70.84 ± 1.57
DDML (LBP)	$\mathbf{81.26 \pm 1.63}$
DDML (combined)	$\mathbf{82.34 \pm 1.47}$

Table 7. Comparisons of the proposed DDML method with different activation functions on the LFW and YTF datasets under the image restricted setting.

Dataset	sigmoid	ns-sigmoid	tanh
LFW	77.18 ± 1.82	85.80 ± 1.39	$\mathbf{87.83 \pm 0.93}$
YTF	70.20 ± 1.26	80.78 ± 1.15	$\mathbf{81.26 \pm 1.63}$

4.4. Effect of the Activation Function

In this subsection, 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 sigmoid` (`ns-sigmoid`)[5]. Table 7 lists the performance of our DDML method with different activation functions on the LFW and YTF datasets, where the SSIFT (square root) feature and LBP feature are used for the LFW and YTF datasets, respectively. We see from this table that the `tanh` function performs the best and the `sigmoid` function performs the worse in our DDML method.

4.5. Computational Time

Lastly, we report the computational time of our DDML method. Our hardware configuration comprises a 3.2-GHz CPU and a 8GB RAM. For each fold, the training time of our DDML are 33.8 and 27.4 seconds, and the testing time

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

are 0.1 and 0.1 seconds on the LFW and YTF datasets[6], respectively. Compared with most existing deep learning methods in [11, 13, 15, 18, 20], our deep learning method is more efficient, especially the training time of our model is much faster. Hence, our DDML complements well to the existing deep learning methods.

5. Conclusion

In this paper, we have presented a new discriminative deep metric learning (DDML) method for face verification in the wild. Our method achieves the very competitive verification performance on the widely used LFW and YTF datasets. How to apply our DDML method to other visual applications such as image classification and activity recognition is an interesting direction of future work.

Acknowledgement

Jiwen Lu is partially supported by the research grant for the Human Sixth Sense Program (HSSP) at the Advanced Digital Sciences Center (ADSC) from the Agency for Science, Technology and Research (A*STAR) of Singapore.

References

[1] T. Ahonen, S. Member, A. Hadid, M. Pietikanen, and S. Member. Face description with local binary patterns: Application to face recognition. *PAMI*, 28:2037–2041, 2006.

[2] Y. Bengio. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.

[3] X. Cai, C. Wang, B. Xiao, X. Chen, and J. Zhou. Deep nonlinear metric learning with independent subspace analysis for face verification. In *ACM MM*, pages 749–752, 2012.

[4] Z. Cao, Q. Yin, X. Tang, and J. Sun. Face recognition with learning-based descriptor. In *CVPR*, page 27072714, 2010.

[5] H. Cevikalp and B. Triggs. Face recognition based on image sets. In *CVPR*, pages 2567–2573, 2010.

[6] 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 *CVPR*, pages 3554–3561, 2013.

[7] J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon. Information-theoretic metric learning. In *ICML*, pages 209–216, 2007.

[8] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, pages 249–256, 2010.

[9] M. Guillaumin, J. Verbeek, and C. Schmid. Is that you? metric learning approaches for face identification. In *ICCV*, pages 498–505, 2009.

[10] G. E. Hinton, S. Osindero, and Y.-W. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, 2006.

[11] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[12] Y. Hu, A. S. Mian, and R. A. Owens. Sparse approximated nearest points for image set classification. In *CVPR*, pages 121–128, 2011.

[13] G. B. Huang, H. Lee, and E. G. Learned-Miller. Learning hierarchical representations for face verification with convolutional deep belief networks. In *CVPR*, pages 2518–2525, 2012.

[14] 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. Technical Report 07-49, University of Massachusetts, Amherst, 2007.

[15] S. Ji, W. Xu, M. Yang, and K. Yu. 3d convolutional neural networks for human action recognition. *PAMI*, 35(1):221–231, 2013.

[16] M. Kan, D. Xu, S. Shan, W. Li, and X. Chen. Learning prototype hyperplanes for face verification in the wild. *TIP*, 22(8):3310–3316, 2013.

[17] T.-K. Kim, J. Kittler, and R. Cipolla. Discriminative learning and recognition of image set classes using canonical correlations. *PAMI*, 29(6):1005–1018, 2007.

[18] 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 *CVPR*, pages 3361–3368, 2011.

[19] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[20] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *ICML*, pages 609–616, 2009.

[21] H. Li, G. Hua, Z. Lin, J. Brandt, and J. Yang. Probabilistic elastic matching for pose variant face verification. In *CVPR*, pages 3499–3506, 2013.

[22] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

[23] 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.*, 35(1):39–51, 2013.

[24] H. Mendez-Vazquez, Y. Martinez-Diaz, and Z. Chai. Volume structured ordinal features with background similarity measure for video face recognition. In *ICB*, pages 1–6, 2013.

[25] A. Mignon and F. Jurie. Pcca: A new approach for distance learning from sparse pairwise constraints. In *CVPR*, pages 2666–2672, 2012.

[26] H. V. Nguyen and L. Bai. Cosine similarity metric learning for face verification. In *ACCV*, pages 709–720, 2010.

[27] M. Ranzato, F. J. Huang, Y.-L. Boureau, and Y. Lecun. Unsupervised learning of invariant feature hierarchies with applications to object recognition. In *CVPR*, pages 1–8, 2007.

[28] K. Simonyan, O. M. Parkhi, A. Vedaldi, and A. Zisserman. Fisher vector faces in the wild. In *BMVC*, 2013.

[29] M. K. stinger, M. Hirzer, P. Wohlhart, P. M. Roth, and H. Bischof. Large scale metric learning from equivalence constraints. In *CVPR*, pages 2288–2295, 2012.

[30] G. W. Taylor, R. Fergus, Y. LeCun, and C. Bregler. Convolutional learning of spatio-temporal features. In *ECCV*, pages 140–153. 2010.

[31] I. W. Tsang, J. T. Kwok, C. Bay, and H. Kong. Distance metric learning with kernels. In *ICANN*, pages 126–129, 2003.

[32] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol. Extracting and composing robust features with denoising autoencoders. In *ICML*, pages 1096–1103, 2008.

[33] R. Wang, S. Shan, X. Chen, and W. Gao. Manifold-manifold distance with application to face recognition based on image set. In *CVPR*, pages 1–8, 2008.

[34] L. Wolf, T. Hassner, and I. Maoz. Face recognition in unconstrained videos with matched background similarity. In *CVPR*, pages 529–534, 2011.

[35] L. Wolf, T. Hassner, and Y. Taigman. Descriptor based methods in the wild. In *ECCVW*, 2008.

[36] L. Wolf, T. Hassner, and Y. Taigman. Similarity scores based on background samples. In *ACCV*, pages 88–97, 2009.

[37] L. Wolf and N. Levy. The svm-minus similarity score for video face recognition. In *CVPR*, pages 3523–3530, 2013.

[38] D.-Y. Yeung and H. Chang. A kernel approach for semisupervised metric learning. *TNN*, 18(1):141–149, 2007.

[39] D. Yi, Z. Lei, and S. Z. Li. Towards pose robust face recognition. In *CVPR*, pages 3539–3545, 2013.

[40] Y. Ying and P. Li. Distance metric learning with eigenvalue optimization. *JMLR*, 13:1–26, 2012.

[41] W.-Y. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld. Face recognition: A literature survey. *ACM Computer Surveys*, 35(4):399–458, 2003.

[6] We use the SSIFT (square root) and LBP feature for the LFW and YTF datasets to compute the computational time of our method, respectively.