# Triangular Similarity Metric Learning for Face Verification

Lilei Zheng, Khalid Idrissi, Christophe Garcia, Stefan Duffner, Atilla Baskurt

# Triangular Similarity Metric Learning for Face Verification

Lilei Zheng, Khalid Idrissi, Christophe Garcia, Stefan Duffner and Atilla Baskurt
Université de Lyon, CNRS, INSA-Lyon, LIRIS, UMR5205, F-69621, France

***Abstract*— We propose an efficient linear similarity metric learning method for face verification called Triangular Similarity Metric Learning (TSML). Compared with relevant state-of-the-art work, this method improves the efficiency of learning the cosine similarity while keeping effectiveness. Concretely, we present a geometrical interpretation based on the triangle inequality for developing a cost function and its efficient gradient function. We formulate the cost function as an optimization problem and solve it with the advanced L-BFGS optimization algorithm. We perform extensive experiments on the LFW data set using four descriptors: LBP, OCLBP, SIFT and Gabor wavelets. Moreover, for the optimization problem, we test two kinds of initialization: the identity matrix and the WCCN matrix. Experimental results demonstrate that both of the two initializations are efficient and that our method achieves the state-of-the-art performance on the problem of face verification.**

## I. Introduction

The objective of face verification is to determine whether two face images are of the same person or not. Formally, two face images of the same person are called a similar pair; otherwise, two face images of different persons are called a dissimilar pair or a different pair.

Face verification is a challenging problem in computer vision due to the possible variations in face pose, facial expression, illumination and partial occlusions, etc. Most of these variations are present in the recently collected data set 'Labeled Faces in the Wild' (LFW) [1] containing numerous annotated images from Yahoo News. Hence LFW has been the most popular benchmark for face verification.

Tremendous efforts have been put on developing robust face descriptors [2], [3], [4], [5], [6], [7], [8], [9], [10] and facial image comparison methods [11], [12], [13], [14], [15], [16], [17], [18], [9]. Popular face descriptors include eigenfaces [2], Gabor wavelets [5], SIFT [4], Local Binary Patterns (LBP) [3], etc. Especially, LBP and its variants, such as center-symmetric LBP (CSLBP) [19], multi-block LBP (MLLBP) [20], three patch LBP (TPLBP) [6] and over-complete LBP (OCLBP) [9], have been proven to be effective at describing facial texture.

Since face verification needs an appropriate way to measure the difference or similarity between two images, many researchers have been studying metric learning which aims at automatically specifying a metric from data pairs [21]. Besides metric learning, discriminative subspace methods have also attracted much attention. For example, whitened Principle Component Analysis (WPCA) [14] and Within Class Covariance Normalization (WCCN) [18], [9] have been used for face verification. In this work, we focus on metric learning methods.

This work was supported by the China Scholarship Council (CSC).

According to the learnt metrics in different methods, one can divide metric learning methods into two main families: distance metric learning and similarity metric learning. Most of the work in distance metric learning has been performed on learning the Mahalanobis distance [11], [12], [13]: $d_M(x,y) = \sqrt{(x-y)^T M(x-y)}$, where $x$ and $y$ are two vectors and $M$ is the matrix that needs to be learnt. Note that when $M$ is the identity matrix, $d_M(x,y)$ is the Euclidean distance. In contrast, similarity metric learning aims to learn similarity of the following form: $s_M(x,y) = x^T My/N(x,y)$, where $N(x,y)$ is a normalization term [22]. Specifically, when $N(x,y) = 1$, $s_M(x,y)$ is the bilinear similarity function [23]; when $N(x,y) = \sqrt{x^T Mx}\sqrt{y^T My}$, $s_M(x,y)$ is the generalized cosine similarity function [15], which has shown its effectiveness on face verification.

In this paper, we propose a novel similarity metric learning method. Compared with the above method of Cosine Similarity Metric Learning (CSML) [15], we develop a geometrically motivated approach based on the triangle inequality which leads to efficient cost and gradient functions. We call our approach TSML, for Triangular Similarity Metric Learning. As a result, besides keeping good performance on face verification, our TSML method is about $20\%$ faster than CSML for calculating the cost and gradient. We formulate the cost and gradient functions into an optimization problem and solve it using the L-BFGS optimization algorithm [24].

To represent face images, besides the common used face descriptors Gabor wavelets [5], SIFT [4] and LBP [3], we introduce OCLBP [9] as another descriptor to improve the overall performance on face verification. We compare our approach with state-of-the-art methods, CSML [15] and WCCN [18], [9] on the LFW data set. All the experiments are performed under the LFW restricted configuration with label-free outside data (LFW-a) [1]. Experimental results show that our method achieves comparable performance with CSML, and slightly surpasses WCCN. Moreover, fusion on three different descriptors brings a significant performance improvement to the proposed TSML: TSML-fusion achieves an accuracy of $89.80\%$ which is competitive with the state-of-the-art results.

The remaining sections are organized as follows. Section II develops the cost and gradient functions motivated by a geometrical interpretation. Section III presents in detail the scheme of our face verification system. Experimental results are reported in section IV. Finally, we draw our conclusions and present some perspectives in section V.

## II. Triangular Similarity Metric Learning

The notion of pairwise metric plays an important role in the area of machine learning [21]. Euclidean distance and cosine similarity are two major metrics that have been used to measure numerical difference or similarity between two vectors. For the task of face verification, Hieu et al.[15] showed that cosine similarity based metric learning achieved better performance than the distance based methods in the literature. In this section, we introduce a similar but more efficient method for similarity metric learning.

The cosine similarity (CS) between two feature vectors $x$ and $y$ is defined as:

$$CS(x,y) = \frac{x^T y}{\|x\|\|y\|}. \tag{1}$$

Applying a linear transformation $f(x,A) = Ax$ on the inputs $x$ and $y$, we obtain two new vectors $Ax$ and $Ay$ in a new space (the target space). Therefore, the cosine similarity in the target space is:

$$CS(x,y,A) = \frac{(Ax)^T Ay}{\|Ax\|\|Ay\|}. \tag{2}$$

The objective of this transformation is to make similar vectors closer and separate dissimilar vectors: an optimal matrix $A$ makes $CS(x,y,A) \geq CS(x,y)$ for a similar pair $(x,y)$ and $CS(x,y,A) \leq CS(x,y)$ for a dissimilar pair. In this work, instead of directly employing the cosine function (Equation 2) in the optimization problem, we develop a novel function to achieve the same goal. At first, we start from the geometrical interpretation of the objective.

### A. Geometrical motivation

We use a triplet $(x_i, y_i, s_i)$ to represent a pair of instances, where $x_i$ and $y_i$ are any two vectors, and $s_i = 1$ (resp. $s_i = -1$) means that the two vectors are similar (resp. dissimilar). Using the linear transformation function $f(x,A)$, we obtain another triplet $(a_i, b_i, s_i)$, where $a_i = f(x_i, A) = Ax_i$ and $b_i = f(y_i, A) = Ay_i$. Let $c_i = a_i + s_i b_i$: $c_i$ is a diagonal of the parallelogram formed by $a_i$ and $b_i$ (Fig. 1(a)). With the well-known *triangle inequality theorem*: the sum of the lengths of two sides of a triangle must always be greater than the length of the third side, we get:

$$\|a_i\| + \|b_i\| - \|c_i\| > 0. \tag{3}$$

We therefore propose to minimize the cost $J_i = \|a_i\| + \|b_i\| - \|c_i\|$. In addition, to prevent $\|a_i\|$ and $\|b_i\|$ from degenerating to 0, we assume that we can keep the norms $\|a_i\|$ and $\|b_i\|$ unchanged during the minimization of $J_i$. Under this assumption, minimizing the cost $J_i$ is equivalent to maximizing $\|c_i\|$. Furthermore, (see Fig. 1(a)) it is also equivalent to minimizing the angle $\theta$ inside a similar pair ($s_i = 1$) or maximizing the angle $\theta$ inside a dissimilar pair ($s_i = -1$), in other words, *minimizing the cosine similarity* between $a_i$ and $s_i b_i$. Note that $\|a_i\| + \|b_i\| = \|c_i\|$ when the cost $J_i$ arrives the minimum 0.

However, it is difficult to leave the norms $\|a_i\|$ and $\|b_i\|$ unchanged during the optimization. Therefore, to prevent $\|a_i\|$ and $\|b_i\|$ from degenerating to 0, we constrain the norms to be approaching 1, i.e., we add constraint factors $(\|a_i\| - 1)^2$ and $(\|b_i\| - 1)^2$ to the cost:

$$\begin{aligned} J_i &= \frac{1}{2}(\|a_i\| - 1)^2 + \frac{1}{2}(\|b_i\| - 1)^2 + \|a_i\| + \|b_i\| - \|c_i\| \\ &= \frac{1}{2}\|a_i\|^2 + \frac{1}{2}\|b_i\|^2 - \|c_i\| + 1. \end{aligned} \tag{4}$$

Substituting the equations $a_i = Ax_i$ and $b_i = Ay_i$ to the above cost function, we get:

$$J_i = \frac{1}{2}x_i^T A^T A x_i + \frac{1}{2}y_i^T A^T A y_i - \|Ax_i + s_i Ay_i\| + 1. \tag{5}$$

The gradient function over the matrix $A$ is:

$$\begin{aligned} \frac{\partial J_i}{\partial A} &= (Ax_i - \frac{Ax_i + s_i Ay_i}{\|Ax_i + s_i Ay_i\|})x_i^T + (Ay_i - s_i\frac{Ax_i + s_i Ay_i}{\|Ax_i + s_i Ay_i\|})y_i^T \\ &= (a_i - \frac{c_i}{\|c_i\|})x_i^T + (b_i - \frac{s_i c_i}{\|c_i\|})y_i^T. \end{aligned} \tag{6}$$

Now, we can obtain the optimal cost at the zero gradient: $a_i - \frac{c_i}{\|c_i\|} = 0$ and $b_i - \frac{s_i c_i}{\|c_i\|} = 0$. In other words, the gradient function has set $\frac{c_i}{\|c_i\|}$ and $\frac{s_i c_i}{\|c_i\|}$ as targets for $a_i$ and $b_i$, respectively. See Fig. 1(b): for a similar pair, $a_i$ and $b_i$ are mapped to the same unit vector along the diagonal (the red solid line); for a dissimilar pair, $a_i$ and $b_i$ are mapped to opposite unit vectors along the other diagonal (the blue solid line).

### B. Cost function

For all possible similar and dissimilar pairs, the overall cost function is:

$$J = \frac{1}{n}\sum_{i=1}^{n} J_i = \frac{1}{n}\sum_{i=1}^{n}[\frac{1}{2}\|a_i\|^2 + \frac{1}{2}\|b_i\|^2 - \|c_i\| + 1], \tag{7}$$

and the corresponding gradient function is:

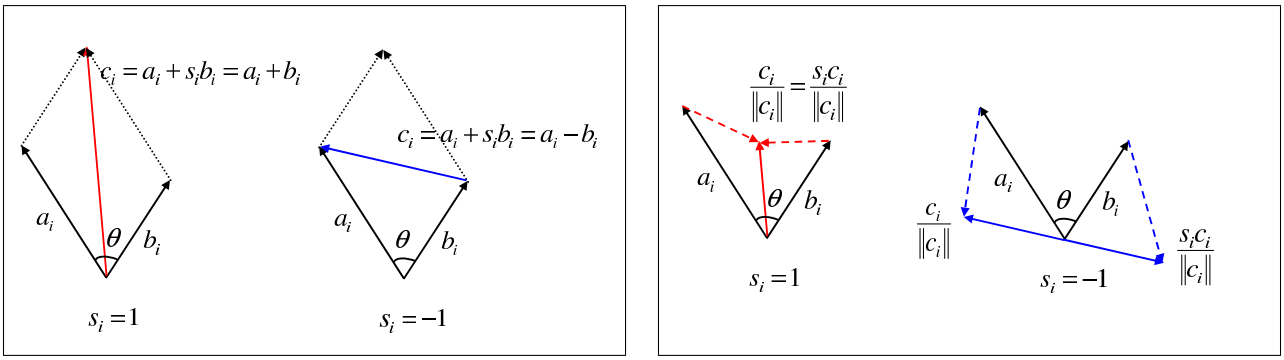
$$\frac{\partial J}{\partial A} = \frac{1}{n}\sum_{i=1}^{n}[(a_i - \frac{c_i}{\|c_i\|})x_i^T + (b_i - \frac{s_i c_i}{\|c_i\|})y_i^T]. \tag{8}$$

However, in this overall case, we can not obtain the optimal solution by computing the zero gradient. Indeed, a specific vector $a_i$ appears in more than one pairs (with different $b_i$), that means taking zero gradient would set many different targets for $a_i$. However, it is impossible for $a_i$ to satisfy all the targets at the same time. Consequently, in the optimal solution, $a_i$ will be mapped to a balanced point of all the targets rather than any one of them.

To prevent the above cost function from over-fitting, we add a regularization term to it:

$$J = \frac{1}{n}\sum_{i=1}^{n}[\frac{1}{2}\|a_i\|^2 + \frac{1}{2}\|b_i\|^2 - \|c_i\| + 1] + \frac{\lambda}{2}\|A - A_0\|^2, \tag{9}$$

where $A_0$ is a pre-defined constant matrix. Additionally, $A_0$ is also the initialization for $A$, i.e., $A$ is set to be $A_0$ before optimizing the cost. The positive parameter $\lambda$ adjusts the effects of the regularization term: the larger the parameter $\lambda$ is, the closer $A$ is to $A_0$. And the best $A$, denoted by $A_\star$, which performs the best on face verification on a validation set, is selected by tuning the parameter




(a) (b)

Fig. 1. Geometrical interpretation of the cost and gradient. (a) Minimizing the cost means to make similar vectors parallel and make dissimilar vectors opposite. (b) Taking zero gradient means to set unit diagonal vectors as targets for $a_i$ and $b_i$. ($s_i = 1$ for similar pair and $s_i = -1$ for dissimilar pair)

$\lambda$. Thus, this regularization guarantees the learnt optimal transformation function $f(x, A_\star)$ to perform better than the initial function $f(x, A_0)$ [25]. The corresponding gradient function is:

$$\frac{\partial J}{\partial A} = \frac{1}{n}\sum_{i=1}^{n}[(a_i - \frac{c_i}{\|c_i\|})x_i^T + (b_i - \frac{s_i c_i}{\|c_i\|})y_i^T] + \lambda(A - A_0). \tag{10}$$

Compared with the gradient of CSML [15]:

$$\frac{\partial J_{CSML}}{\partial A} = \frac{1}{n}\sum_{i=1}^{n}\frac{s_i}{\|a_i\|\|b_i\|}[(\frac{a_i^T b_i}{\|a_i\|^2}a_i - b_i)x_i^T + (\frac{a_i^T b_i}{\|b_i\|^2}b_i - a_i)y_i^T] + \lambda(A - A_0), \tag{11}$$

our gradient (Equation 10) needs fewer matrix multiplications, this makes TSML more efficient than CSML during gradient descent. In the experiment section IV-C, we present a runtime comparison of CSML and TSML to prove the efficiency of our method. Concretely, we use the L-BFGS [24] optimization algorithm to compute the optimal solution. Compared with the standard gradient descent algorithm, the L-BFGS algorithm has no need to manually pick a learning rate and it's usually much faster. In this work, we use a MATLAB implementation of L-BFGS provided by Mark Schmidt [1].

## III. Face Verification

We apply the linear TSML method for face verification on the LFW data set [1]. In this section, we introduce the scheme of our verification system in detail. First, we consider several data pre-processing techniques: subset partition, feature extraction, dimensionality reduction, and parameter initialization. Then, we explain the procedure of learning and evaluation. All the experiments are performed under the LFW restricted configuration with label-free outside data: only the provided 6000 pairs of data are used in evaluation.

### A. Subsets partition

As suggested in [1], we only use View 2 subset of LFW for performance evaluation. In View 2, all the 5749 people in the data set are divided into 10 folds where the identities are mutually exclusive. The total number of images for all the people is 13,233, however, the number of images for each person varies from 1 to 530.

We experiment only on the restricted configuration of LFW, i.e., training and evaluating only on the specified 3000 similar pairs and 3000 dissimilar pairs. Concretely, all data of the restricted configuration are separated into 3 partitions: a training set, a validation set and a test set. We learn a model on the training set, choose the best model that achieves the highest performance on the validation set, and report the performance on the testing set using the best model.

We perform cross-validation on the 10 folds: there are overall 10 experiments, in each repetition, 4800 pairs from 8 folds are used for training, 600 pairs from another fold are used for validation and 600 pairs from the last fold are used for testing. For example, the first experiment uses subsets (1,2,3,4,5,6,7,8) for training, subset 9 for validation and subset 10 for testing; the second experiment uses (2,3,4,5,6,7,8,9) for training, subset 10 for validation and subset 1 for testing. At last, we report the mean accuracy ($\pm$standard error of the mean) of the 10 experiments.

### B. Feature extraction

Following [15], [9], [18], we use aligned images of the LFW-a version [26]: the original $250 \times 250$ images are aligned and cropped to subimages of $150 \times 80$ pixels, from which we extract four face descriptors: Gabor wavelets [5], LBP [3], SIFT [4] and OCLBP [9]. For Gabor and LBP, we used the same setting as in [15], dimension of Gabor and LBP is 4800 and 7080, respectively. For SIFT, we directly use the feature data provided by [27]. For OCLBP, we used similar setting as in [9], dimension of our OCLBP vectors is 46,846. Readers are referred to the appendix for the details of extracting OCLBP. Additionally, square roots of all the descriptors are also evaluated.

### C. Dimensionality reduction

Usually, automatic learning is performed on thousands of feature vectors. Directly taking the original facial vectors for learning causes computational problem. For example, the time required for multiplications between 46846-d OCLBP vectors would be unacceptable. Therefore, before learning,

[1] http://www.di.ens.fr/ mschmidt/Software/minFunc.html

we apply whitened Principle Component Analysis (WPCA) for dimensionality reduction [15], [9]. On the one hand, performing PCA reduces the original dimension; on the other hand, whitening makes new feature vectors more discriminative for face verification: the PCA reduced feature is normalized by the eigenvalues over all the dimensions, thus the negative influences of the large eigenvectors are reduced while the discriminating details of the smaller eigenvectors are enhanced [28].

In our experiments, following [18], all the original feature vectors are transformed to new vectors with dimension 300. The transformation matrix of WPCA is computed only using similar pairs from the 8-fold training set, i.e., 4800 feature vectors. Usually, data selection for WPCA does not significantly affect the performance after transformation. The only point worth noting is that we should select enough data, for example, under this experiment setting, an appropriate number of feature vectors for whitening is more than 1000 [15], [18].

### D. Parameter initialization

As we have mentioned when commenting Equations 9 and 10, we need to specify the initialization matrix $A_0$ before learning. In our experiments, we use two kinds of initialization: the first one is the identity matrix $I$, the second one is the WCCN matrix $W$ [9], [18], i.e., we directly set $A = A_0 = I$ or $A = A_0 = W$ before learning.

WCCN was first used for speaker recognition [29]. Recently, it was introduced to improve the discrimination for face verification [9]. At first, a within class covariance matrix $C$ is defined as:

$$C = \frac{1}{t}\sum_{i=1}^{t}\frac{1}{m_i}\sum_{j=1}^{m_i}(x_{ij}-\mu_i)(x_{ij}-\mu_i)^T, \qquad (12)$$

where $t$ is the number of different classes, $m_i$ is the number of instances in the $i$th class, $x_{ij}$ is the $j$th instance in the $i$th class and $\mu_i$ is the mean of the $i$th class. Decomposition on the matrix $C$ produces eigenvalues $\lambda_1, \ldots, \lambda_k$ and eigenvectors $V = \{v_1, \ldots, v_k\}$. Thus, the WCCN matrix is:

$$W = diag(\lambda_1, \ldots, \lambda_k)V^T, \qquad (13)$$

Under the restricted configuration of LFW, we regard each similar pair as a mini-class of its own for doing WCCN transformation [9]. It is worth noting that all the eigenvalues and eigenvectors are retained, i.e., we do not perform dimensionality reduction in this step.

### E. Learning and evaluation

Once we set up the initialization, we use the advanced optimization algorithm L-BFGS [24] to compute the optimal solution.

Inspired by WCCN that only considers the within class covariance, we *perform learning on similar pairs only*. In fact, using similar pairs only achieved better performance than using both similar and dissimilar pairs. Indeed, since in LFW, the identities in the validation and testing set are different from those in the training set, the discriminative information learnt between the individuals in the training set is not a good prediction of that between the individuals in the validation and testing set. So it is better not to use the dissimilar pairs.

Let $x$ and $y$ denote whitened feature vectors after performing dimensionality reduction. Once we learn the optimal solution $A_\star$, we compute the cosine similarity score $CS(x, y, A_\star)$ using Equation 2. The final decision is made by comparing to a threshold $\gamma$: if $CS(x, y, A_\star) \geq \gamma$, $x$ and $y$ are similar; otherwise, they are dissimilar. We record the percentage of right decisions on the validation set, i.e.,

$$accuracy = \frac{number\ of\ right\ decisions}{total\ number\ of\ pairs}. \qquad (14)$$

After that, we select the best optimal solution $A_\star$ and the best threshold $\gamma$ that give the best accuracy on the validation set. At last, we report accuracy on the testing set using the best $A_\star$ and $\gamma$.

## IV. Experiments and Analysis

We experiment with five different methods for face verification on LFW:

- WCCN [18], [9]: performing WCCN on the whitened feature vectors $x$ and $y$ and evaluating on the outputs, i.e., computing cosine similarity score using $CS(x, y, W)$, where $W$ is the WCCN matrix;
- CSML-I: performing CSML on the whitened feature vectors $x$ and $y$ with initialization matrix $A_0 = I$, and evaluating on the outputs, i.e., computing cosine similarity score using $CS(x, y, A_\star)$, where $A_\star$ is the learnt transformation matrix;
- CSML-WCCN: performing CSML on the whitened feature vectors $x$ and $y$ with initialization matrix $A_0 = W$, and evaluating on the outputs, i.e., computing cosine similarity score using $CS(x, y, A_\star)$, where $A_\star$ is the learnt transformation matrix;
- TSML-I: performing TSML on the whitened feature vectors $x$ and $y$ with initialization matrix $A_0 = I$, and evaluating on the outputs, i.e., computing cosine similarity score using $CS(x, y, A_\star)$, where $A_\star$ is the learnt transformation matrix;
- TSML-WCCN: performing TSML on the whitened feature vectors $x$ and $y$ with initialization matrix $A_0 = W$, and evaluating on the outputs, i.e., computing cosine similarity score using $CS(x, y, A_\star)$, where $A_\star$ is the learnt transformation matrix;

We have only two parameters to tune in the experiments: the regularization term $\lambda$ and the decision threshold $\gamma$. The tuning range of $\lambda$ was from $10^{-4}$ to $10^{-3}$ with a step size of $2 \times 10^{-5}$; the tuning range of $\gamma$ was from $-1$ to 1 with a step size of $0.001$.

### A. Experimental results

At the beginning, we directly perform evaluation on the 300-d whitened feature vectors, i.e., computing cosine similarity score using $CS(x, y, I)$, where $I$ is the identity matrix.

TABLE I

FACE VERIFICATION ACCURACY (±STANDARD ERROR OF THE MEAN) ON LFW-A UNDER THE RESTRICTED CONFIGURATION USING FIVE DIFFERENT METHODS: WCCN, CSML-I, CSML-WCCN, TSML-I, TSML-WCCN. DIMENSION OF THE WHITENED FEATURE VECTORS IS 300.

| Method | LBP | | OCLBP | | SIFT | | Gabor | |
|---|---|---|---|---|---|---|---|---|
| | original | square root | original | square root | original | square root | original | square root |
| Baseline | 77.17±0.49 | 79.73±0.38 | 80.43±0.25 | 81.55±0.44 | 76.88±0.42 | 77.52±0.49 | 75.28±0.45 | 77.25±0.32 |
| WCCN | 80.40±0.39 | 84.23±0.33 | 83.75±0.51 | 86.83±0.37 | 82.72±0.39 | 84.17±0.25 | 78.68±0.62 | 81.52±0.65 |
| CSML-I | **83.18±0.71** | 85.17±0.60 | 85.27±0.60 | 87.03±0.50 | 84.73±0.51 | 85.95±0.37 | 81.45±0.54 | 83.53±0.42 |
| CSML-WCCN | 82.88±0.74 | 85.17±0.61 | 85.15±0.76 | ***87.18±0.46*** | **85.00±0.53** | 85.82±0.32 | **81.62±0.61** | **83.58±0.52** |
| TSML-I | 82.63±0.68 | 85.40±0.52 | **85.27±0.73** | 87.02±0.40 | 84.50±0.67 | **86.08±0.44** | 80.63±0.54 | 83.00±0.47 |
| TSML-WCCN | 82.40±0.80 | **85.58±0.58** | 84.98±0.75 | 87.10±0.43 | 84.83±0.58 | 85.70±0.43 | 80.82±0.52 | 83.35±0.57 |

TABLE II

TIME COST (±STANDARD ERROR OF THE MEAN) IN **MILLISECONDS** OF CSML-I AND TSML-I ON LFW-A UNDER THE RESTRICTED CONFIGURATION.

| Method | LBP | | OCLBP | | SIFT | | Gabor | |
|---|---|---|---|---|---|---|---|---|
| | original | square root | original | square root | original | square root | original | square root |
| CSML-I | 91.38±1.23 | 92.46±1.40 | 92.26±0.76 | 93.71±1.11 | 91.42±0.62 | 93.30±1.03 | 92.13±0.70 | 90.96±0.78 |
| TSML-I | 73.41±0.71 | 73.79±0.62 | 74.74±1.03 | 74.77±0.79 | 74.69±0.44 | 74.75±0.64 | 74.64±0.86 | 74.45±0.72 |
| Relative Improvement | 19.67% | 20.19% | 18.99% | 20.21% | 18.30% | 19.88% | 18.98% | 18.15% |

We consider this evaluation as the baseline. Tables I summarizes the experimental results of the baseline and the five methods. In general, the proposed linear methods, TSML-I and TSML-WCCN, achieve competitive performance on all the different descriptors. For example, TSML-WCCN obtains the highest accuracy of $85.58\%$ on whitened LBP.

Especially, we implemented state-of-the-art methods WCCN [18], [9] and CSML [15] in our experiments (Table I). Readers are referred to [15], [18], [9] to confirm that results of our implementation are consistent with the original versions.

### B. Effectiveness of TSML

From Table I, comparing the first two rows, we can see that using WCCN significantly improves the performance on face verification. For instance, WCCN allows an improvement from $79.73\%$ to $84.23\%$ over the baseline on the square-rooted LBP. This phenomenon has also been reported in both [9] and [18]. Moreover, compared with WCCN, metric learning methods CSML-I, CSML-WCCN, TSML-I and TSML-WCCN further improve the decision accuracy. For example, TSML-WCCN gets $85.58\%$, which outperforms the $84.23\%$ of WCCN on the square-rooted LBP.

Interestingly, comparing the two different kinds of initialization (Table I: CSML-I vs. CSML-WCCN, TSML-I vs. TSML-WCCN), we observe that all of them obtain similar results. Remind that TSML-I takes the identity matrix as initialization and TSML-WCCN takes the WCCN matrix as initialization. We deduce that either of the two initialization matrices is acceptable for the linear metric learning methods. The difference is that *different initializations imply different lower bounds*: for example, TSML-I sets the results of the baseline as the lower bound, and TSML-WCCN sets the results of WCCN as the lower bound. However, though the two initializations set different lower bounds for learning, they may have comparable upper bounds, so we have observed similar results.

### C. Efficiency of TSML

Due to our aim of performing efficient similarity metric learning, we carry out a runtime comparison between CSML [15] and the proposed TSML. Comparing their performance on face verification, we find that the TSML methods achieve almost the same performance with the CSML methods (Table I). For example, on the square-rooted OCLBP descriptor, CSML-I, CSML-WCCN, TSML-I and TSML-WCCN obtain $87.03\%$, $87.18\%$, $87.02\%$ and $87.10\%$, respectively. However, our proposed linear TSML (Equation 10) is theoretically more efficient than CSML.

We perform the efficiency comparison on all the 300-d whitened features: with fixed regularization parameter $\lambda = 0$, we calculate the gradient of CSML-I and TSML-I on the training set for only once and record their time consumption, respectively. Like the reported accuracy on face verification, we report the average time (±standard error of the mean) spent for 10 repetitions. This comparison is performed on a machine with a 4-core CPU, 8 GB RAM and 64-bit operating system. Table II summarizes the time cost on each feature in milliseconds. Generally, TSML relatively improves the efficiency by about $20\%$ over CSML. For example, on the square-rooted OCLBP, calculating the gradient of CSML-I for once averagely costs 93.71 milliseconds, in contrast, calculating the gradient of TSML-I averagely costs only 74.77 milliseconds.

### D. Comparison with the state-of-the-art

Fusion on different descriptors generally leads to performance gain [15], [9], [18], [30], [31]. Therefore we perform fusion on the similarity scores of the proposed TSML methods and compare with the state-of-the-art methods (table III).

After learning a linear model, one can produce similarity scores for all the pairs of vectors on all possible descriptors. For each pair of vectors, all the corresponding similarity scores compose a new short vector which can be used to predict the final decision: a pair of vectors are similar or

TABLE III

FACE VERIFICATION ACCURACY (±STANDARD ERROR OF THE MEAN) ON LFW-A UNDER THE RESTRICTED CONFIGURATION USING DIFFERENT METHODS.

| Method | Accuracy |
|---|---|
| DML-eig-fusion [30] | 85.65±0.56 |
| CSML-fusion [15] | 88.00±0.37 |
| PAF [33] | 87.77±0.51 |
| WCCN-fusion [9] | **91.10±0.59** |
| Sub-SML-fusion [18] | 89.73±0.38 |
| DDML-fusion [31] | **90.68±1.41** |
| LM3L [34] | 89.57±1.53 |
| TSML-fusion (this work) | **89.80±0.47** |

not. We consider it as a two-class classification problem and employ a linear support vector machine (SVM) [32] to perform the classification.

Since LBP is a subpart of OCLBP [9], we abandon LBP and collect similarity scores on the other three descriptors and their square roots for fusion. Both of the two methods, TSML-I and TSML-WCCN, are used to produce the similarity scores. Thus we have totally 12 similarity scores for each pair. After that, we train an SVM model on the validation set and make prediction on the test set. The fusion result of the proposed methods is $89.80\%$, which occupies the third position among the state-of-the-art methods under the restricted configuration (Table III and Fig. 2). Available ROC curves of more approaches are present in the result page of the LFW data set [2].

Other methods that perform well on the face verification problem (Table III), for example WCCN, rely on a fusion of high-dimensional features (i.e., the 96520-d Scattering descriptor in WCCN-fusion), or they are naturally slower because they perform more complex optimization, e.g., DDML-fusion [31] integrates deep neural network with distance metric learning. Apart from fusion on multiple features, employing advanced processing steps specific to face verification can also produce promising results, such as the facial landmark extraction and 3D model fitting in PAF [33].

## V. CONCLUSION AND FUTURE WORK

In this paper, we have proposed an efficient linear TSML method for face verification. We illustrated a geometrical interpretation of our cost and gradient functions. By comparing TSML with CSML [15], we found that less matrix multiplication makes our method more efficient than CSML. In terms of effectiveness, our methods achieves comparable performance with the state-of-the-art methods on face verification.

Extensive experiments were performed on the LFW-a data set [1] under restricted configuration. We used WPCA to reduce dimension of the original feature vectors to a treatable scale. Besides the identity matrix, we also introduced the WCCN matrix for initialization of learning. We found that either of the identity matrix or the WCCN matrix is acceptable, the only difference is that different initializations imply different lower bounds of learning.

[2]http://vis-www.cs.umass.edu/lfw/results.html#ImageRestrictedLF

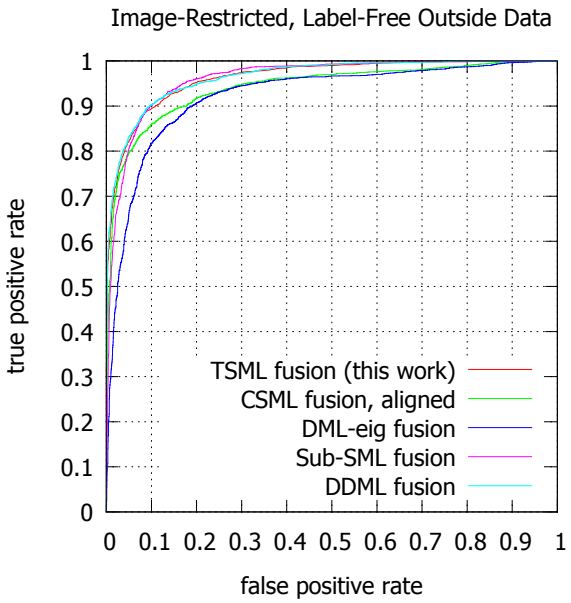



Fig. 2. ROC curves of the proposed TSML-fusion method (red line) and the other state-of-the-art methods on LFW under the restricted configuration with label-free outside data (LFW-a).

In addition, we have also confirmed the effectiveness of our TSML method on another problem of kinship verification in a recent evaluation [35]. In the future, we plan to extend the linear similarity metric learning method to non-linear case. We are interested in applying convolutional neural networks, that are able to automatically extract relevant features from the images while performing non-linear projection [36].

## REFERENCES


[1] 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," Tech. Rep., University of Massachusetts, Amherst, 2007.

[2] Ma. A. Turk and A. P. Pentland, "Face recognition using eigenfaces," in *Proc. CVPR*. IEEE, 1991, pp. 586–591.

[3] T. Ahonen, A. Hadid, and M. Pietikäinen, "Face recognition with local binary patterns," in *Proc. ECCV*. 2004, pp. 469–481, Springer.

[4] Y. Ke and R. Sukthankar, "Pca-sift: A more distinctive representation for local image descriptors," in *Proc. CVPR*. IEEE, 2004, vol. 2, pp. II–506.

[5] J. G. Daugman, "Complete discrete 2-d gabor transforms by neural networks for image analysis and compression," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 36, no. 7, pp. 1169–1179, 1988.

[6] L. Wolf, T. Hassner, and Y. Taigman, "Descriptor based methods in the wild," in *Workshop on Faces in'Real-Life'Images: Detection, Alignment, and Recognition*, 2008.

[7] S. U. Hussain, T. Napoléon, and F. Jurie, "Face recognition using local quantized patterns," in *Proc. BMVC*, 2012.

[8] K. Simonyan, O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Fisher vector faces in the wild," in *Proc. BMVC*, 2013, vol. 1, p. 7.

[9] O. Barkan, J. Weill, L. Wolf, and H. Aronowitz, "Fast high dimensional vector multiplication face recognition," in *Proc. ICCV*, 2013.

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

[11] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell, "Distance metric learning with application to clustering with side-information," *Advances in neural information processing systems*, pp. 521–528, 2003.
[12] K. Weinberger, J. Blitzer, and L. Saul, "Distance metric learning for large margin nearest neighbor classification," *Advances in neural information processing systems*, vol. 18, pp. 1473, 2006.
[13] J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon, "Information-theoretic metric learning," in *Proc. ICML*. ACM, 2007, pp. 209–216.
[14] N. V. Hieu, B. Li, and L. Shen, "Local gabor binary pattern whitened pca: A novel approach for face recognition from single image per person," in *Advances in Biometrics*, pp. 269–278. Springer, 2009.
[15] N. V. Hieu and B. Li, "Cosine similarity metric learning for face verification," in *Proc. ACCV*. 2011, pp. 709–720, Springer.
[16] T. Berg and P. N. Belhumeur, "Tom-vs-pete classifiers and identity-preserving alignment for face verification.," in *Proc. BMVC*, 2012, vol. 1, p. 5.
[17] P. Li, Y. Fu, U. Mohammed, J. H. Elder, and S. JD Prince, "Probabilistic models for inference about identity," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 34, no. 1, pp. 144–157, 2012.
[18] Q. Cao, Y. Ying, and P. Li, "Similarity metric learning for face recognition," in *Proc. ICCV*, 2013.
[19] M. Heikkilä, M. Pietikäinen, and C. Schmid, "Description of interest regions with center-symmetric local binary patterns," in *Computer Vision, Graphics and Image Processing*, pp. 58–69. Springer, 2006.
[20] L. Zhang, R. Chu, S. Xiang, S. Liao, and S. Z. Li, "Face detection based on multi-block lbp representation," in *Advances in biometrics*, pp. 11–18. Springer, 2007.
[21] A. Bellet, A. Habrard, and M. Sebban, "A survey on metric learning for feature vectors and structured data," *Computing Research Repository*, vol. abs/1306.6709, 2013.
[22] A. M. Qamar, E. Gaussier, J. P. Chevallet, and J. H. Lim, "Similarity learning for nearest neighbor classification," in *Proc. ICDM*. IEEE, 2008, pp. 983–988.
[23] G. Chechik, V. Sharma, U. Shalit, and S. Bengio, "Large scale online learning of image similarity through ranking," *The Journal of Machine Learning Research*, vol. 11, pp. 1109–1135, 2010.
[24] D. C. Liu and J. Nocedal, "On the limited memory bfgs method for large scale optimization," *Mathematical programming*, vol. 45, no. 1-3, pp. 503–528, 1989.
[25] N. V. Hieu, *Linear subspace methods in face recognition*, Ph.D. thesis, University of Nottingham, 2011.
[26] Y. Taigman, L. Wolf, and T. Hassner, "Multiple one-shots for utilizing class label information.," in *BMVC*, 2009, pp. 1–12.
[27] Matthieu Guillaumin, Jakob Verbeek, and Cordelia Schmid, "Is that you? metric learning approaches for face identification," in *Computer Vision, 2009 IEEE 12th International Conference on*. IEEE, 2009, pp. 498–505.
[28] W. Deng, J. Hu, and J. Guo, "Gabor-eigen-whiten-cosine: A robust scheme for face recognition," in *Analysis and Modelling of Faces and Gestures*, pp. 336–349. Springer, 2005.
[29] A. O. Hatch and A. Stolcke, "Generalized linear kernels for one-versus-all classification: application to speaker recognition," in *Proc. ICASSP*. IEEE, 2006, vol. 5.
[30] Yiming Ying and Peng Li, "Distance metric learning with eigenvalue optimization," *The Journal of Machine Learning Research*, vol. 13, no. 1, pp. 1–26, 2012.
[31] Junlin Hu, Jiwen Lu, and Yap-Peng Tan, "Discriminative deep metric learning for face verification in the wild," in *Proc. CVPR*, 2014, pp. 1875–1882.
[32] Chih-Chung Chang and Chih-Jen Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011, Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.
[33] D. Yi, Z. Lei, and S. Z. Li, "Towards pose robust face recognition," in *Proc. CVPR*. IEEE, 2013, pp. 3539–3545.
[34] J. Hu, J. Lu, J. Yuan, and Y.-P. Tan, "Large margin multimetric learning for face and kinship verification in the wild," in *Proc. ACCV*, 2014.
[35] J. Lu, J. Hu, V. E. Liong, X. Zhou, A. Bottino, I. U. Islam, T. F. Vieira, X. Qin, X. Tan, Y. Keller, L. Zheng, K. Idrissi, C. Garcia, S. Duffner, A. Baskurt, M. Castrillon-Santana, and J. Lorenzo-Navarro, "The FG 2015 Kinship Verification in the Wild Evaluation," in *Proc. FG*, 2015.
[36] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *Proc. CVPR*. IEEE, 2005, vol. 1, pp. 539–546.
[37] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 24, no. 7, pp. 971–987, 2002.

## APPENDIX

**Local Binary Patterns**: Ojala et al. [37] proposed uniform patterns of LBP for robust face recognition. The uniform LBP is denoted as $LBP^{u2}_{p,r}$, where $u2$ stands for 'uniform', $(p, r)$ means to sample $p$ points over a circle with a radius $r$. The dimension of an uniform pattern is $59$. In this work, we followed the setting as in [15]. We use aligned images of the LFW-a version [26]: the original $250 \times 250$ images are aligned and cropped to subimages with size $150 \times 80$. Each cropped image is divided into non-overlapping $10 \times 10$ blocks and uniform LBP patterns $LBP^{u2}_{8,1}$ are extracted from all the blocks. Finally, we catenate all the LBP patterns into a feature vector, whose dimension is $7080$ $(15 \times 8 \times 59)$.

**Over-complete Local Binary Patterns**: We also used a variant of LBP, OCLBP, to improve the overall performance on face verification [9]. Unlike LBP, OCLBP adopts overlapping to adjacent blocks. Formally, the configuration of OCLBP is denoted as $S : (a, b, v, h, p, r)$: an image is divided into $a \times b$ blocks with vertical overlap of $v$ and horizontal overlap of $h$, and then uniform pattern $LBP^{u2}_{p,r}$ are extracted from all the blocks. Moreover, OCLBP is composed of several different configurations. For example, Oren et al. [9] used three configurations: $S : (10, 10, \frac{1}{2}, \frac{1}{2}, 8, 1), (14, 14, \frac{1}{2}, \frac{1}{2}, 8, 2), (18, 18, \frac{1}{2}, \frac{1}{2}, 8, 3)$. The three configurations consider three block sizes: $10 \times 10, 14 \times 14, 18 \times 18$, and half overlap rates along the vertical and horizontal directions.

Concretely, *they shift the images to produce overlaps*. For instance, a cropped $150 \times 80$ image is divided into $15 \times 8 = 120$ blocks with the size $10 \times 10$. Shifting the image to the left by a step $10 \times \frac{1}{2} = 5$ also produces $120$ blocks; and shifting downwards produces another $120$ blocks. Hence there are totally $360$ blocks under the configuration $S : (10, 10, \frac{1}{2}, \frac{1}{2}, 8, 1)$. Similarly, there are $198$ blocks under the configuration $S : (14, 14, \frac{1}{2}, \frac{1}{2}, 8, 2)$ and $135$ blocks under the configuration $S : (18, 18, \frac{1}{2}, \frac{1}{2}, 8, 3)$. In summary, the dimension of their OCLBP vectors is $40,887$ $((360 + 198 + 135) \times 59)$.

In our work, *we shift the block window to produce overlaps*. Taking the $10 \times 10$ block window for example, with the shifting step $10 \times \frac{1}{2} = 5$ to the left and downwards, the total number of $10 \times 10$ blocks is $(\frac{150}{5} - 1) \times (\frac{80}{5} - 1) = 435$. Similarly, shifting the $14 \times 14$ window produces $231$ blocks and shifting the $18 \times 18$ window produces $128$ blocks. The dimension of our OCLBP vectors is $46,846$ $((435 + 231 + 128) \times 59)$.

Apparently, both the two forms of OCLBP contains LBP as a subpart. Comparing the experimental results using the two different OCLBP, we found no significant difference. For example, WCCN gets $86.83\%$ using our 46,846-d OCLBP, which is very close to $87.23\%$ in [9] using the 40,887-d OCLBP.