Siamese Multi-layer Perceptrons for Dimensionality Reduction and Face Identification

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

Noname manuscript No.
(will be inserted by the editor)

Siamese Multi-layer Perceptrons for Dimensionality Reduction and Face Identification

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

Received: date / Accepted: date

Abstract This paper presents a framework using siamese Multi-layer Perceptrons (MLP) for supervised dimensionality reduction and face identification. Compared with the classical MLP that trains on fully labeled data, the siamese MLP learns on side information only, i.e., how similar of data examples are to each other. In this study, we compare it with the classical MLP on the problem of face identification. Experimental results on the Extended Yale B database demonstrate that the siamese MLP training with side information achieves comparable classification performance with the classical MLP training on fully labeled data. Besides, while the classical MLP fixes the dimension of the output space, the siamese MLP allows flexible output dimension, hence we also apply the siamese MLP for visualization of the dimensionality reduction to the 2-d and 3-d spaces.

Keywords siamese neural networks · multi-layer perceptrons · metric learning · face identification · dimensionality reduction

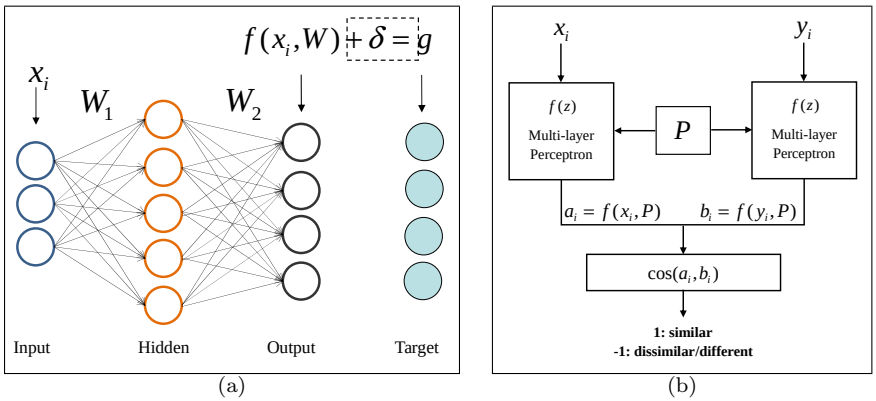
1 Introduction

With the capability of approximating non-linear mappings, Multi-layer Perceptrons (MLP) has been a popular solution to object classification problems since the 1980s, finding applications in diverse fields such as image recognition [28] and speech recognition [20, 5].

A classical MLP consists of an input layer, one or more hidden layer(s) and an output layer of perceptrons. Generally, in a multi-class classification problem, the size of the output layer (i.e., the output dimension), is fixed to

Lilei Zheng
E-mail: lilei.zheng@insa-lyon.fr

All the authors are with the Université de Lyon, CNRS, INSA-Lyon, LIRIS, UMR5205, F-69621, France



(a) (b)

Fig. 1 (a)Traditional single multi-layer perceptrons. (b) Siamese multi-layer perceptrons .

the number of classes in this problem. Figure 1 (a) illustrates the structure of an MLP. The objective of such an MLP is to make the network outputs approximating predefined target values (or ground truth) for different classes. In practice, the error δ between the output and the target is used to update the network parameters via the Back-propagation algorithm [25]. Moreover, these predefined target values are typically binary for classification problems. For example, for a 3-class classification problem, we usually set unit vectors $[1,0,0]^T, [0,1,0]^T, [0,0,1]^T$ as the target vectors for the 3 classes.

In this work, we propose a siamese MLP framework to relax the constraint on the output dimension, making flexible dimensionality reduction to the input data. A siamese MLP is a symmetric architecture consisting of two MLPs, where they actually share the same set of parameters P (Figure 1 (b)). Compared with the single MLP (Figure 1 (a)), instead of constraining the outputs approaching some predefined target values, the siamese MLP defines a specific objective: (1) for an input pair from the same class, making the pairwise similarity between their outputs larger; (2) for an input pair from different classes, making the pairwise similarity between their outputs smaller. With such an objective, the dimension of the target space can be arbitrarily specified.

Another advantage of the siamese MLP over the classical MLP is that the siamese MLP is able to learn on data pairs instead of fully labeled data. In other words, the siamese MLP is applicable for weakly supervised cases where we have no access to the labels of training instances: only some side information of pairwise relationship is available. This is a meaningful setting in various applications where labeled data are more costly than the side information [3]. Examples include users' implicit feedback on the internet (e.g., clicks on search engine results), citations among articles or links in a social network, kinship relationship between individuals [22].

More interestingly, the siamese MLP has these two advantages WITHOUT losing its superior ability of accurate classification. In the experiments, we compare the siamese MLP with the classical MLP for face identification on the

Extended Yale B database [13]. In addition, we employ a statistical significance testing method called Bootstrap Resampling [18] to evaluate the comparison between the siamese MLP and the classical MLP. The testing results show that the siamese MLP achieves comparable performance with the classical MLP on the problem of face identification.

Overall, the main contributions of this paper are summarized as below:

- we have presented the siamese MLP as a semi-supervised learning method for classification. It can perform learning from side-information only, instead of fully labeled training data.
- we have shown the capability of the siamese MLP for dimensionality reduction and data visualization in 2-d and 3-d spaces. We find that the siamese MLP projects the original input data to the vertexes of a regular polyhedron (see Figure 7).
- we have demonstrated that the siamese MLP has the above two advantages WITHOUT losing its superior ability of accurate classification. It achieves comparable performance with the standard MLP on face identification.

The remainder of this paper is organized as follows: Section 2 briefly summarizes the related work on siamese neural networks and metric learning. Section 3 presents the proposed siamese MLP method. Section 4 depicts the datasets and experiments on face identification. Finally, we draw the conclusions in Section 5.

2 Related Work

Using MLP for dimensionality reduction is an old idea which has its origins in the late 1980s and early 1990s. The first work may be the Auto-Associate Neural Networks (AANN) [8,14], a special type of MLP where the input and output layers have the same number of neurons, and the middle hidden layer has fewer neurons than the input and output layers. The objective of AANN is to reproduce the input pattern at its output. Thus it actually learns a mapping on the input patterns into a lower-dimensional space and then an inverse mapping to reconstruct the input patterns. Since it does not need the input data to be labeled, the middle hidden layer learns a compact representation of the input data in an unsupervised manner [11]. However, researchers have found that the dimensionality reduction by the AANN is quite similar with the well-known Principal Components Analysis (PCA) technique [12].

More recently, a more mature and powerful AANN, the deep autoencoder networks [16] have presented an effective way of initializing the network parameters that leads the low-dimensional coding much better than PCA. For all the layers in the deep networks, the authors proposed a restricted Boltzmann machine to pretrain the network parameters layer-by-layer, followed by a fine-tuning procedure for optimal reconstruction via the Back-propagation algorithm [25].

Different from the unsupervised dimensionality reduction by the above AANNs, we propose to employ the MLP to perform dimensionality reduction

in a supervised manner using siamese neural networks. Siamese neural networks have first been presented by Bromley et al. [6] using Time Delay Neural Networks (TDNN) on the problem of signature verification. This idea was then adopted by Chopra et al. [7] who used siamese Convolutional Neural Networks (CNN) for face verification, i.e., to decide if two given face images belong to the same person or not. Recently, Berlemont et al. [4] also successfully employed the siamese neural networks for inertial gesture recognition and rejection.

Concretely, the siamese neural networks minimize a loss function that drives the similarity metric to be small for data pairs from the same class, and large for pairs from different classes [7]. This technique of specifying a metric from data pairs (or triplets) is also called *Metric Learning* [3, 27, 10, 29]. In this paper, the proposed siamese MLP employs the Triangular Similarity Metric Learning (TSML) objective function [29] as the loss function, and shows its effectiveness on dimensionality reduction and object classification.

3 Siamese Multi-Layer Perceptron

In this section, we present the classical MLP model and the proposed siamese MLP model. Since the siamese MLP takes the MLP as a basic component, we first introduce the classical MLP model in detail. After that, we develop the siamese variant. Concretely, we use a 3-layer MLP consisting of an input layer, an output layer and only one hidden layer.

3.1 Three-layer MLP

An MLP is a *feed-forward* neural network, i.e., the activation of the neurons is propagated layer-wise from the input to the output layer [11]. Moreover, the activation function of the neurons has to be differentiable in order to update the network parameters via the Back-propagation algorithm. Commonly used non-linear activation functions include the sigmoid function and the *tanh* function (i.e., the hyperbolic tangent function). In contrast with that the sigmoid function allows only positive output values, the *tanh* function produces both negative and positive output values. Since negative values are necessary in the proposed siamese MLP (Section 3.2), we choose the *tanh* function in our experiments. The *tanh* function and its derivative are:

$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}, \tag{1}$$

$$tanh'(x) = 1 - tanh^2(x). \tag{2}$$

3.1.1 Feed-forward

First, we introduce the feed-forward procedure of the 3-layer MLP. For any given input sample $\mathbf{x}_i$, assuming its output on the MLP is $\mathbf{a}_i$. At the first

step, from the the input layer to the hidden layer, with the parameter matrix $W^{(1)}$ and the bias vector $b^{(1)}$, the values in the hidden layer are computed as $\mathbf{h}_i = tanh(W^{(1)}\mathbf{x}_i + b^{(1)})$. At the second step, from the hidden layer to the output layer, with the parameter matrix $W^{(2)}$ and the bias vector $b^{(2)}$, the output values are calculated as $\mathbf{a}_i = tanh(W^{(2)}\mathbf{h}_i + b^{(2)})$. Finally, the objective function of an MLP classifier is simply the Mean Squared Error (MSE) between the computed outputs and their desired targets for all training samples:

$$J = \frac{1}{2N}\sum_{i=1}^{N}(\mathbf{a}_i - \mathbf{g}_i)^2, \tag{3}$$

where N is the number of all possible training samples, $\mathbf{g}_i$ is the target vector for the output sample $\mathbf{a}_i$. Remind that $\mathbf{g}_i$ is usually hand-crafted unit vectors. For example, for a 3-class classification problem, we usually set unit vectors $[1,0,0]^T, [0,1,0]^T, [0,0,1]^T$ as the target vectors for the 3 classes.

3.1.2 Back-propagation

Now we use the Back-propagation algorithm [25] to update the set of parameters $P : \{W^{(2)}, b^{(2)}, W^{(1)}, b^{(1)}\}$. Taking derivative of Equation (3), the gradient for the i_{th} sample is:

$$\frac{\partial J_i}{\partial P} = (\mathbf{a}_i - \mathbf{g}_i)^T \frac{\partial \mathbf{a}_i}{\partial P}, \tag{4}$$

and the differential on the output layer, with respect to $\mathbf{z}_i^{(2)} = W^{(2)}\mathbf{h}_i + b^{(2)}$, is:

$$\delta_i^{(2)} = (1 - \mathbf{a}_i \odot \mathbf{a}_i) \odot (\mathbf{a}_i - \mathbf{g}_i), \tag{5}$$

where the notation $\odot$ means element-wise multiplication. Subsequently, the differential on the hidden layer, with respect to $\mathbf{z}_i^{(1)} = W^{(1)}\mathbf{x}_i + b^{(1)}$, is:

$$\delta_i^{(1)} = (1 - \mathbf{h}_i \odot \mathbf{h}_i) \odot [(W^{(2)})^T \delta_i^{(2)}], \tag{6}$$

and the differentials of the network parameters are computed as:

$$\Delta_i W^{(2)} = \delta_i^{(2)} \mathbf{h}_i^T, \tag{7}$$

$$\Delta_i b^{(2)} = \delta_i^{(2)}, \tag{8}$$

$$\Delta_i W^{(1)} = \delta_i^{(1)} \mathbf{x}_i^T, \tag{9}$$

$$\Delta_i b^{(1)} = \delta_i^{(1)}. \tag{10}$$

After that, the parameters $P : \{W^{(2)}, b^{(2)}, W^{(1)}, b^{(1)}\}$ can be updated by using the following gradient descent function:

$$P = P - \mu \sum_{i=1}^{N} \Delta_i P, \tag{11}$$

where μ is the learning rate. The default learning rate is set to 10^{-4} in our experiments.

3.2 Siamese MLP

As we have illustrated in Figure 1 (b), a siamese MLP consists of two MLPs which actually share the same set of parameters $P : \{W^{(2)}, b^{(2)}, W^{(1)}, b^{(1)}\}$. Let $\mathbf{a}_i = f(\mathbf{x}_i, P)$ denotes the output of an input $\mathbf{x}_i$, and $\mathbf{b}_i = f(\mathbf{y}_i, P)$ denotes the output of the other input $\mathbf{y}_i$. Compared with the traditional MLP that makes the output $\mathbf{a}_i$ close to its hand-crafted target $\mathbf{g}_i$, the siamese MLP aims to make $\{\mathbf{a}_i, \mathbf{b}_i\}$ close if $\{\mathbf{x}_i, \mathbf{y}_i\}$ are of the same class and to separate $\{\mathbf{a}_i, \mathbf{b}_i\}$ if $\{\mathbf{x}_i, \mathbf{y}_i\}$ are of two different classes [29]. Consequently, the siamese MLP needs no hand-crafted targets.

To achieve this goal, we employ a modified Triangular Similarity Metric Learning (TSML) objective function [29]:

$$J_i = K(\|\mathbf{a}_i\| + \|\mathbf{b}_i\| - \|\mathbf{c}_i\|) + \frac{1}{2}(\|\mathbf{a}_i\| - K)^2 + \frac{1}{2}(\|\mathbf{b}_i\| - K)^2, \tag{12}$$

where K is a constant that constrains the length (i.e., the L_2 norm) of $\mathbf{a}_i$ and $\mathbf{b}_i$; $\mathbf{c}_i = \mathbf{a}_i + s_i\mathbf{b}_i$ and $s_i = 1$ (resp. $s_i = -1$) means that the two vectors $\mathbf{a}_i$ and $\mathbf{b}_i$ are a within-class pair (resp. a between-class pair). Generally, we can set the constant K with the average length of all the input training vectors.

The first part of Equation (12), $\|\mathbf{a}_i\| + \|\mathbf{b}_i\| - \|\mathbf{c}_i\|$, includes three sides of an triangle (Figure 2 (a)). According to 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, the first part should be always larger than 0. Moreover, minimizing this part is equivalent to minimizing the angle θ inside a within-class pair ($s_i = 1$) or maximizing the angle θ inside a between-class 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. Besides, the second part of Equation (12), $\frac{1}{2}(\|\mathbf{a}_i\| - K)^2 + \frac{1}{2}(\|\mathbf{b}_i\| - K)^2$, aims to prevent $\|\mathbf{a}_i\|$ and $\|\mathbf{b}_i\|$ from degenerating to 0.

Further, Equation (12) can be rewritten as:

$$J_i = \frac{1}{2}\|\mathbf{a}_i\|^2 + \frac{1}{2}\|\mathbf{b}_i\|^2 - K\|\mathbf{c}_i\| + K^2, \tag{13}$$

with gradient over the parameters P:

$$\frac{\partial J_i}{\partial P} = (\mathbf{a}_i - K\frac{\mathbf{c}_i}{\|\mathbf{c}_i\|})^T \frac{\partial \mathbf{a}_i}{\partial P} + (\mathbf{b}_i - K\frac{s_i\mathbf{c}_i}{\|\mathbf{c}_i\|})^T \frac{\partial \mathbf{b}_i}{\partial P}. \tag{14}$$

Now, we can obtain the optimal cost $J_i = 0$ at the zero gradient: $\mathbf{a}_i - K\frac{\mathbf{c}_i}{\|\mathbf{c}_i\|} = 0$ and $\mathbf{b}_i - K\frac{s_i\mathbf{c}_i}{\|\mathbf{c}_i\|} = 0$. In other words, the gradient function has set $K\frac{\mathbf{c}_i}{\|\mathbf{c}_i\|}$ and $K\frac{s_i\mathbf{c}_i}{\|\mathbf{c}_i\|}$ as targets for $\mathbf{a}_i$ and $\mathbf{b}_i$, respectively. See Figure 2(b): for a within-class pair, $\mathbf{a}_i$ and $\mathbf{b}_i$ are mapped to the same vector along the diagonal (the red solid line); for a between-class pair, $\mathbf{a}_i$ and $\mathbf{b}_i$ are mapped to opposite vectors along the other diagonal (the blue solid line).

More interestingly, substituting the hand-crafted target $\mathbf{g}_i$ with the two automatically computed targets $K\frac{\mathbf{c}_i}{\|\mathbf{c}_i\|}$ and $K\frac{s_i\mathbf{c}_i}{\|\mathbf{c}_i\|}$, the siamese MLP gradient

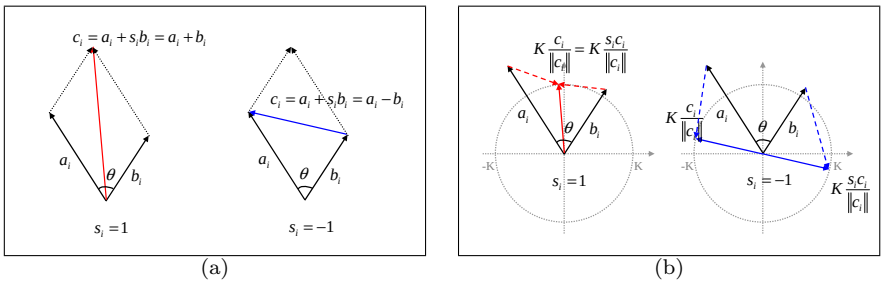


Fig. 2 Geometrical interpretation of the cost and gradient. (a) Minimizing the cost means to make a within-class pair parallel and make a between-class pair opposite. (b) Taking zero gradient means to set diagonal vectors as targets for a_i and b_i. ($s_i = 1$ for a within-class pair and $s_i = -1$ for a between-class pair)

function (Equation (14)) is exactly a double copy of the traditional MLP gradient function (Equation (4)). And this fact allows us to use the same Back-propagation algorithm to update the network parameters (Section 3.1.2).

3.3 Difference between MLP and Siamese MLP

In the last two subsections, we have shown that the classical MLP and the siamese MLP have similar gradient formulations that allows us to employ the same Back-propagation algorithm for training. However, there are also apparent differences between them on both the input and output layers.

For each input vector x_i, the classical MLP needs to know which class x_i belongs to. In contrast, the siamese MLP takes a more flexible constraint: it only needs the side information – whether two input vectors x_i and y_i are of the same class or not. The relationship between the two constraints can be summarized as:

- when we know the classes of x_i and y_i, we know whether x_i and y_i are of the same class or not;
- however, even we know whether x_i and y_i are of the same class or not, we may have no idea about the class labels of x_i and y_i.

As a result, the siamese MLP is applicable with the second constraint while the classical MLP is not, i.e., the siamese MLP can learn on side information only (Section 1). More important, we will demonstrate that the relaxation of constraint would not cause classification accuracy loss to the experiments (Section 4).

On the output layer, the classical MLP fixes the output dimension equal to the number of classes. However, the siamese MLP has no constraint on the output dimension. Therefore, for a problem with more than 3 classes, the siamese MLP is applicable for data visualization, i.e., projecting the input data into 2-d or 3-d spaces; but the classical MLP can only make a projection into a space with dimension more than 3. In Section 4.4, we will illustrate the effect of the siamese MLP on dimensionality reduction and data visualization.

3.4 Batch Gradient Descent or Stochastic Gradient Descent

Once we have defined an error function and its gradient, the Back-propagation algorithm [25] applies the gradient descent technique to minimize the overall error for all training data iteratively.

There are mainly three modes to perform gradient descent: stochastic gradient descent, batch gradient descent, or the trade-off between them, mini-batch gradient descent. Concretely, stochastic gradient descent uses only one training sample in each iteration while batch gradient descent uses all training samples in each iteration. Mini-batch gradient descent, as the name suggests, takes several training samples in each iteration. Usually, the mini-batch gradient descent is the fastest choice among the three for many optimization problems[1].

Particularly, batch gradient descent can be involved in some advanced optimization algorithms to accelerate the learning speed, such as the Conjugate Gradient Descent (CGD) algorithm [23] and the Limited-memory Broyden Fletcher Goldfarb Shanno (L-BFGS) algorithm [21]. Compared with the standard gradient descent technique, these advanced algorithms have no need to manually pick a learning rate and are usually much faster for small and medium scale problems. However, for a large scale problem with an overlarge training dataset, it may be impossible to load all the training data into memory in a single iteration. In this case, the mini-batch gradient descent may be more applicable as it takes only a few training samples in each iteration.

For the proposed siamese MLP, the advanced algorithms using batch gradient descent maybe only suitable for small scale problems, because the siamese MLP takes *data pairs* in the learning procedure, and the total number of all training sample pairs is exponentially larger than the total number of training samples. Specifically, for a problem of N training samples, the number of all possible sample pairs is $N(N-1)/2$. Therefore, for medium and large scale problems, we have to use stochastic gradient descent or mini-batch gradient descent.

Commonly, a probable mini-batch contains equivalent number of within-class pairs and between-class pairs [7, 29]. However, the actual ratio of within-class pairs and between-class pairs is not equivalent. For example, for m classes each with n training samples, the number of within-class pairs is $mn(n-1)/2$ and the number of all between-class pairs is $mn(mn-n)/2$. Thus the ratio between within-class pairs and between-class pairs is $\frac{n-1}{n(m-1)}$, i.e., one within-class pair is accompanied by $\frac{n(m-1)}{n-1}$ between-class pairs. Consequently, instead of taking equivalent number of within-class pairs and between-class pairs in a mini-batch, we propose the following strategy to choose data pairs for a mini-batch:

- Count the training samples and denote the number as N, hence there are totally $N(N-1)/2$ sample pairs.

[1] http://en.wikipedia.org/wiki/Stochastic_gradient_descent

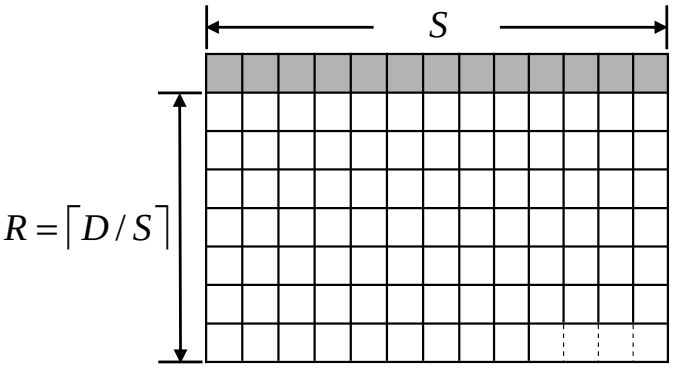


Fig. 3 Index matrix for mini-batch gradient descent of the siamese MLP. The first row stores the S within-class pairs, followed by all the between-class pairs. The empty positions in the end of the matrix can be optionally filled with some between-class pairs.

- Count the within-class pairs and denote the number as S, then the number of between-class pairs is $D = N(N-1)/2 - S$.
- Let $R = \lceil D/S \rceil$, i.e., the smallest integer not less than D/S.
- Make an index matrix with $R+1$ rows and S columns (Figure 3), put the indexes of the S within-class pairs in the first row and put the indexes of all the between-class pairs in the following rows.
- (Optional) Randomly pick some between-class pairs to fill the remain empty position in the end of the matrix.
- Take the indexes in each column as a mini-batch, which contains a single within-class pair and R between-class pairs.

In general, we summarize the optimization procedure for the proposed siamese MLP in Algorithm 1. For a large scale problem, the mini-batch gradient descent is used in optimization; for a small scale problem, the batch gradient descent is adopted. Particularly, the scale of a problem is small or large depends on the machine capacity we used. In our case, we usually consider a problem with more than $1,000$ training samples as a large scale problem, since the number of all possible similar and dissimilar pairs is at least $499,500$.

4 Experiment and Analysis

4.1 Extended Yale B Database

We perform experiments on the Extended Yale B database [13]. It contains 2,414 frontal-face images of 38 individuals. These images were captured under various lighting conditions. All the images have been cropped and normalized, with the same size 192×168. Figure 4 provides some example images of an individual in the database. We can see that the lighting directions in different

Algorithm 1: Optimization of the siamese MLP

input : Training set; Number of training data N;
output: Parameters P

% *initialization*
Random initialization to the set of parameters P;
% *optimization by back propagation*
if N *is large* **then**
    % *this is a large scale problem* ($N > 1000$)
    Set learning rate $\mu = 10^{-4}$;
    Generate mini batches that each contains 1 similar pair and R dissimilar pairs (Figure 3);
    Employ mini-batch gradient descent to update P;
else
    % *this is a small scale problem*
    Generate a whole batch which contains all similar and dissimilar pairs;
    Employ batch gradient descent (the advanced L-BFGS algorithm) to update P;
% *output the final set of parameters*
return P.

Fig. 4 Example images of an individual in the Extended Yale B database. These frontal-face images were captured under various lighting conditions.

images are significantly varied. For instance, it is difficult to recognize the face in the middle of Figure 4 since it hides in deeply dark.

We divide the whole database into three non-overlapping subsets: training, validation and testing. We learn a model on the training set, choose the best set of parameters that achieves the highest performance on the validation set, and report the performance on the testing set using the best parameters. Especially, we take a small scale training set in the experiments: for each individual, only one out of ten images are used for training, i.e., there are 263 face images in the training set. And the size ratio of the training, validation and testing sets is 1:3:6. All the experiments are repeated 10 times with randomly shuffled data, and the mean accuracy (± standard error of the mean) are reported.

4.2 Face Descriptors

Popular face descriptors for face detection and face recognition include eigenfaces [26], Gabor wavelets [9], haar-like features [19], SIFT [17], Local Binary Pattern(LBP) [1], etc. Recently, Barkan *et al.* [2] proposed Over-complete Local Binary Patterns (OCLBP), a new variant of LBP that significantly improved the face verification performance. Thus we adopt the OCLBP feature as the major face descriptor in our experiments. Besides, we also use Gabor wavelets and the standard LBP to represent the face images as a comparison. Following [2,29], both the original face descriptors and their square roots are evaluated in the experiments.

Gabor wavelets: we extract Gabor wavelets with 5 scales and 8 orientations on each downsampled image. The downsampling rate is 10×10 for all the 192×168 images, thus the dimension of an extracted Gabor vector is 12160 (= 5×8×19×16).

Local Binary Patterns: we use the uniform LBP [24] to represent face images. 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. Concretely, each 192×168 image is divided into non-overlapping 16×16 blocks and uniform LBP patterns $LBP^{u2}_{8,1}$ are extracted from all the blocks. We catenate all the LBP patterns into a feature vector, whose dimension is 7788 (= 12×11×59).

Over-complete Local Binary Patterns: besides LBP, we also use its new variant, OCLBP, to improve the overall performance on face identification [29]. 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, the OCLBP is composed of several different configurations: $S_1 : (16, 16, \frac{1}{2}, \frac{1}{2}, 8, 1)$, $S_2 : (24, 24, \frac{1}{2}, \frac{1}{2}, 8, 2)$, $S_3 : (32, 32, \frac{1}{2}, \frac{1}{2}, 8, 3)$. The three configurations consider three block sizes: 16×16, 24×24, 32×32, and adopt half overlap rates along the vertical and horizontal directions. We shift the block window to produce overlaps. Taking the 16×16 block window for example,

with the shifting step $16 \times \frac{1}{2} = 8$ to the left and downwards, the total number of 16×16 blocks is $(\frac{192}{8} - 1) \times (\frac{168}{8} - 1) = 460$. Similarly, shifting the 24×24 window produces 195 blocks and shifting the 32×32 window produces 110 blocks. The dimension of our OCLBP vectors is $45,135$ $((460+195+110)\times 59)$. Apparently, the OCLBP contains the LBP as a subpart, hence using OCLBP always achieves better classification performance than using LBP.

Usually, directly taking the original face descriptors for learning causes computational problem. For example, the time required for multiplications between $45,135$-d OCLBP vectors would be unacceptable. Therefore, before learning, we apply whitened PCA to reduce the vector dimension. Since the size of the training set is small (only 263 samples), we keep all the variance during dimensionality reduction. Thus the reduced dimension is 262, and these 262-d feature vectors are taken as inputs to the classical MLP or the siamese MLP.

4.3 Dimensionality Reduction in Face Identification

We evaluate three different methods in our experiments: K-Nearest Neighbors (KNN), MLP and the proposed siamese MLP. Since the siamese MLP is designed for nonlinear mapping rather than classification, it is hard to directly make class predictions on its output. Hence we apply KNN on its output to perform class identification. This is also the reason why we evaluate the KNN method as a comparison. Specifically, KNN in our experiments uses the cosine function to measure the pairwise distance and the number of nearest neighbors K is set to 1.

4.3.1 Output dimension of the siamese MLP

Empirically, the size of the hidden layer is set to 100 for both the classical MLP and the siamese MLP. As the number of different classes in the Extended Yale B database is 38, the output dimension of the classical MLP is fixed to 38. In contrast, the siamese MLP allows flexible output dimension, thus we shift the output dimension from 2 to 250 and record the influence on the identification accuracy. Note that the input dimension is 262, so we keep the output dimension less than 262 in order to perform dimensionality reduction. Figure 5 shows the identification accuracy curve of the siamese MLP method on the square-rooted OCLBP feature. We can see that the curve rises rapidly when the output dimension increases from 2 to 10, but then climbs much more slowly. The optimal solution is with the output dimension of more than 80.

4.3.2 Comparison to the classical MLP

Table 1 summarizes the results of different methods on different face descriptors on the extended Yale B database. The output dimension of the siamese

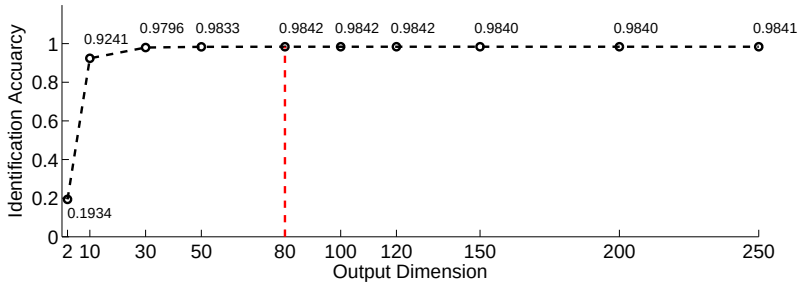


Fig. 5 Identification accuracy curve of the siamese MLP method on the square-rooted OCLBP feature, with respect to the increasing output dimension.

Table 1 Face identification performance on the extended Yale B database. Generally, Siamese MLP = MLP > KNN. The output dimension of the siamese MLP is set to 80.

Method		KNN	MLP	Siamese MLP
Gabor	original	0.6937(±0.0432)	**0.7972(±0.0349)**	0.7970(±0.0344)
	square-rooted	0.8032(±0.0043)	0.9248(±0.0027)	**0.9262(±0.0028)**
LBP	original	0.7906(±0.0042)	0.9215(±0.0041)	**0.9227(±0.0039)**
	square-rooted	0.8478(±0.0051)	0.9628(±0.0030)	**0.9634(±0.0031)**
OCLBP	original	0.8250(±0.0054)	0.9641(±0.0028)	**0.9659(±0.0031)**
	square-rooted	0.8611(±0.0055)	0.9833(±0.0017)	**0.9842(±0.0016)**

Table 2 Significance testing between MLP and siamese MLP. A p-value smaller than 0.05 or 0.01 indicates a significant difference. Results confirm no significant difference between MLP and siamese MLP.

Method		MLP	Siamese MLP	p-value
Gabor	original	**0.7972(±0.0349)**	0.7970(±0.0344)	0.4982
	square-rooted	0.9248(±0.0027)	**0.9262(±0.0028)**	0.3559
LBP	original	0.9215(±0.0041)	**0.9227(±0.0039)**	0.4150
	square-rooted	0.9628(±0.0030)	**0.9634(±0.0031)**	0.4486
OCLBP	original	0.9641(±0.0028)	**0.9659(±0.0031)**	0.3364
	square-rooted	0.9833(±0.0017)	**0.9842(±0.0016)**	0.3341

MLP is set to 80. Compared with KNN, the siamese MLP has brought significant improvement on face identification. Compared with the classical MLP, the siamese MLP achieves comparable results. For example, on the square-rooted LBP features, the siamese MLP obtains an accuracy of 0.9634, seems slightly better than the result of the classical MLP, 0.9628. Besides, methods using square-rooted features always obtain better performance than those using the original features. This phenomenon is consistent with that on the problem of face verification [29].

To confirm the comparison, we employ the Bootstrap Resampling approach [18] to evaluate the pairwise statistical significance between the two methods. Note that the smaller the p-value, the larger the significance. Usually, we consider a p-value smaller than 0.05 or 0.01 to indicate a significant difference. The significance testing results in Table 2 are all in the range [0.3,

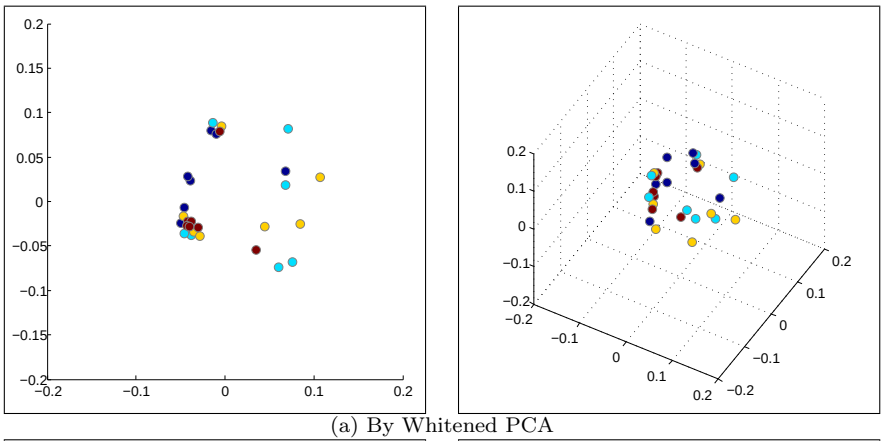
Fig. 6 Face images that the siamese MLP using square-rooted OCLBP failed to recognize.

0.5], showing that there is no significant performance difference between the classical MLP and the siamese MLP. We also test the significance between siamese MLP and KNN, the p-value is always 0 on all the difference features, demonstrating that the siamese MLP has significantly improve the performance over the KNN method.

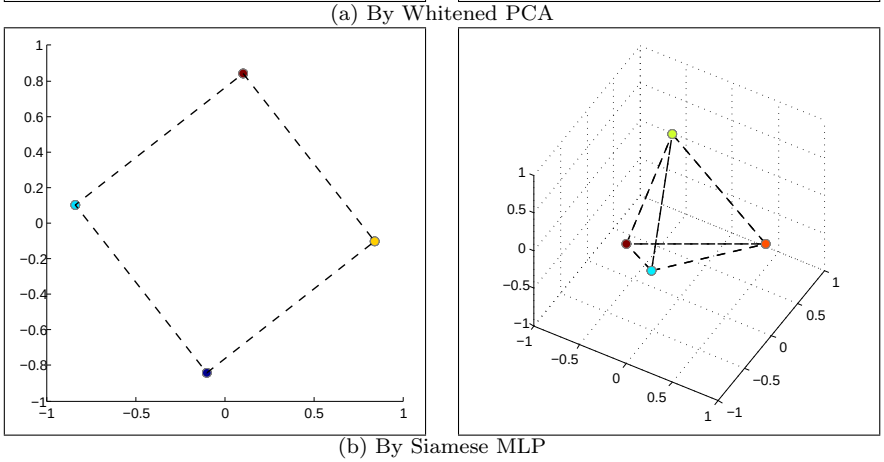
Comparing the three different face descriptors, the results on OCLBP are significantly better than those on Gabor wavelets and those on LBP. For example, the siamese MLP using square-rooted OCBLP achieves an average accuracy of 0.9842 on the 10 repeated experiments. Figure 6 shows the face images that the siamese MLP failed to recognize. Most of the failure examples are rather dark so that it is difficult to extract effective facial texture features from them. However, there are also some failure examples in good lighting condition. This is probably because we apply KNN as the classifier and the final decision relies on the test sample's nearest neighbor in the training set. Since the training data are randomly selected, a good nearest neighbor for each test sample is not guaranteed.

4.4 Dimensionality Reduction in Data Visualization

In this subsection, we apply the siamese MLP to illustrate data visualization on a few data from the Extended Yale B database. We select the first 4 classes each with 7 face images, totally 28 face images. These images are represented



(a) By Whitened PCA

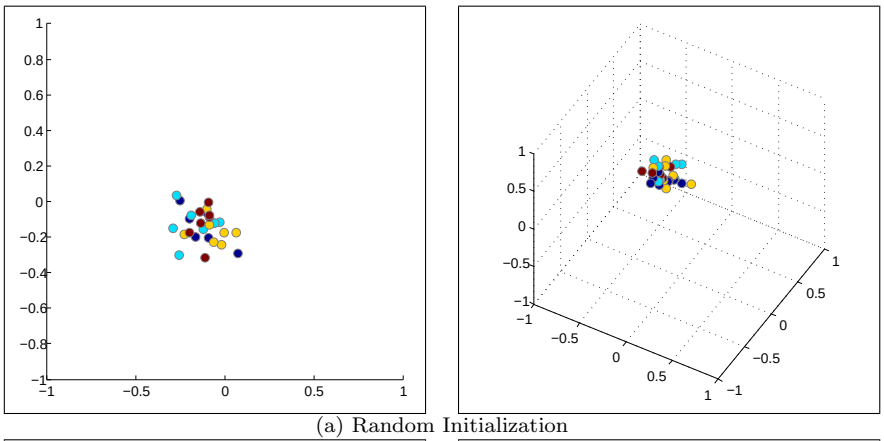


(b) By Siamese MLP

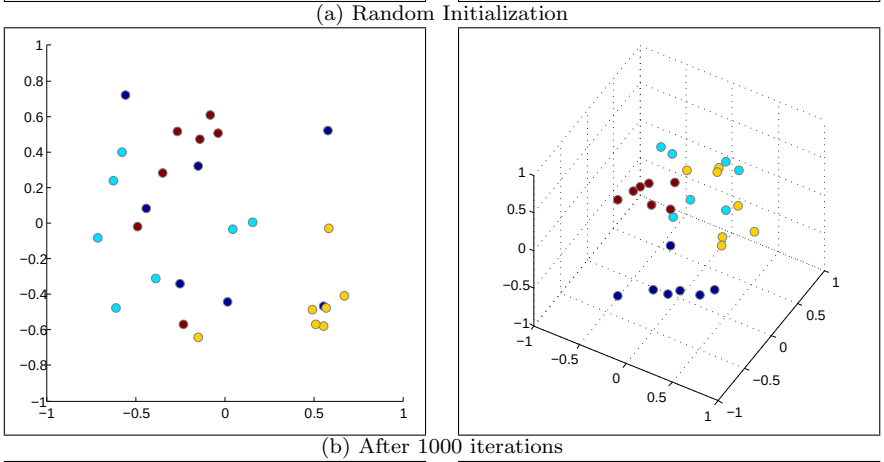
Fig. 7 Visualization of dimensionality reduction into the 2-d or 3-d spaces using (a) Whitened PCA and (b) Siamese MLP.

by 262-d OCLBP feature vectors. For data visualization, all the input vectors are projected into the 2-d and 3-d spaces, respectively. In addition, we also visualize the projection of whitened PCA as a comparison in Figure 7.

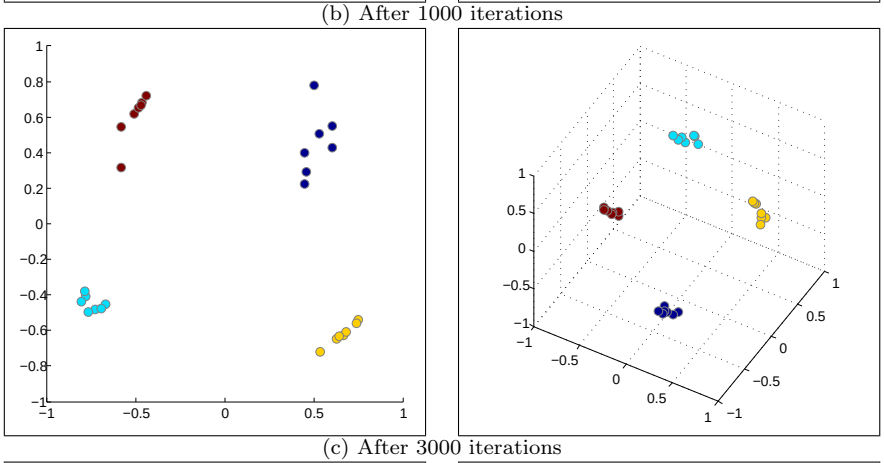
Figure 7 (a) shows the data distribution in the 2-d and 3-d target spaces using whitened PCA, points with different colors are from 4 different classes. We can see that points of different classes are mixing in both the 2-d and 3-d spaces. In contrast, the siamese MLP successfully separates the points of different classes (Figure 7 (b)). More interestingly, points of the same class concentrate tightly at a certain position, standing as a vertex of a square in the 2-d space or a regular tetrahedron in the 3-d space. Note that both the square and the regular tetrahedron take the origin point as the center. Thus all the between-class pairs share exactly the same angle: (1) in the 2-d space, the angle between two points from different classes is 90°; (2) in the 3-d space, the between-class angle is about 109.47°. In summary, the objective of our



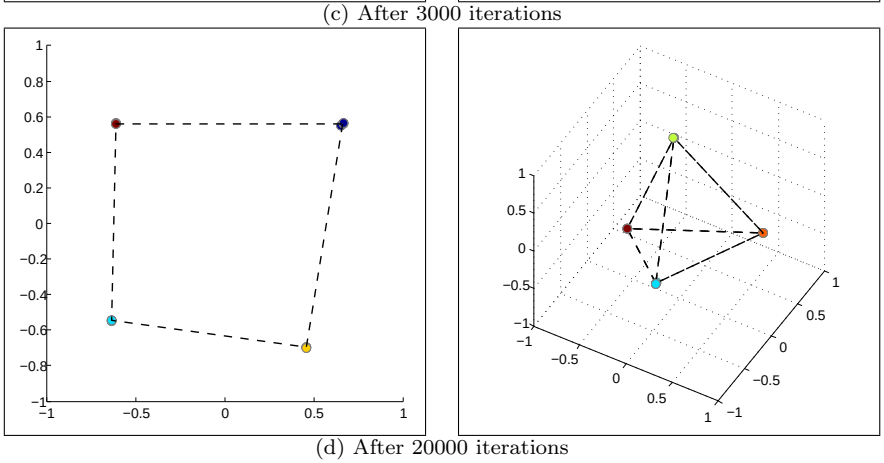
(a) Random Initialization



(b) After 1000 iterations



(c) After 3000 iterations



(d) After 20000 iterations

Fig. 8 Illustration of dimensionality reduction into the 2-d or 3-d spaces using Siamese MLP.

siamese MLP has been satisfied perfectly: separating the between-class pairs and concentrating the within-class pairs.

Figure 8 pictures a more detailed procedure of data projection by the siamese MLP using the mini-batch gradient descent algorithm (Section 3.4). At the beginning, the siamese MLP is initialized with random parameters, so we observe mixed data classes around the origin point in Figure 8 (a). Towards the objective of closing the within-class pairs and separating between-class pairs, the points scatter away after 1000 iterations. Successively, after 3000 iterations, data from different classes have found their own optimal positions, and we can see clear blank boundaries between different classes. Finally, after 20000 iterations, data of the same class concentrate at each optimal position in Figure 8 (d).

5 Conclusion

In this work, we have presented the siamese MLP method for dimensionality reduction. One advantage of the siamese MLP is that it allows flexible output dimension, we have visualized the results of dimensionality reduction into the 2-d and 3-d spaces, showing interesting geometrical characteristic. Another advantage of the siamese MLP is that it learns on side information only. And we have compared it with the classical MLP on the problem of face identification, showing that the siamese MLP training with side information achieves comparable classification performance with the classical MLP training on fully labeled data. In the future, we are interested in changing the proposed optimal objective into a margin-based variant [27] and applying it for manifold learning [15].

References

1. Ahonen, T., Hadid, A., Pietikäinen, M.: Face recognition with local binary patterns. In: Proc. ECCV, pp. 469–481. Springer (2004)
2. Barkan, O., Weill, J., Wolf, L., Aronowitz, H.: Fast high dimensional vector multiplication face recognition. In: Proc. ICCV, pp. 1960–1967. IEEE (2013)
3. Bellet, A., Habrard, A., Sebban, M.: A survey on metric learning for feature vectors and structured data. arXiv preprint arXiv:1306.6709 (2013)
4. Berlemont, S., Lefebvre, G., Duffner, S., Garcia, C.: Siamese Neural Network based Similarity Metric for Inertial Gesture Classification and Rejection. In: 11th IEEE International Conference on Automatic Face and Gesture Recognition (2015)
5. Bourlard, H., Wellekens, C.J.: Links between markov models and multilayer perceptrons. Pattern Analysis and Machine Intelligence, IEEE Transactions on **12**(12), 1167–1178 (1990)
6. Bromley, J., Bentz, J.W., Bottou, L., Guyon, I., LeCun, Y., Moore, C., Säckinger, E., Shah, R.: Signature verification using a siamese time delay neural network. International Journal of Pattern Recognition and Artificial Intelligence **7**(04), 669–688 (1993)
7. Chopra, S., Hadsell, R., LeCun, Y.: Learning a similarity metric discriminatively, with application to face verification. In: Proc. CVPR, vol. 1, pp. 539–546. IEEE (2005)
8. Cottrell, G.W., Metcalfe, J.: Empath: face, emotion, and gender recognition using holons. In: Advances in Neural Information Processing Systems, pp. 564–571. Morgan Kaufmann Publishers Inc. (1990)

9. Daugman, J.G.: Complete discrete 2-d gabor transforms by neural networks for image analysis and compression. Acoustics, Speech and Signal Processing, IEEE Transactions on **36**(7), 1169–1179 (1988)
10. Davis, J.V., Kulis, B., Jain, P., Sra, S., Dhillon, I.S.: Information-theoretic metric learning. In: International Conference on Machine learning, pp. 209–216. ACM (2007)
11. Duffner, S.: Face image analysis with convolutional neural networks. Ph.D. thesis (2008)
12. Dunteman, G.H.: Principal components analysis. 69. Sage (1989)
13. Georghiades, A.S., Belhumeur, P.N., Kriegman, D.: From few to many: Illumination cone models for face recognition under variable lighting and pose. Pattern Analysis and Machine Intelligence, IEEE Transactions on **23**(6), 643–660 (2001)
14. Golomb, B.A., Lawrence, D.T., Sejnowski, T.J.: Sexnet: A neural network identifies sex from human faces. In: Advances in Neural Information Processing Systems, pp. 572–579 (1991)
15. Hadsell, R., Chopra, S., LeCun, Y.: Dimensionality reduction by learning an invariant mapping. In: Proc. CVPR, vol. 2, pp. 1735–1742. IEEE (2006)
16. Hinton, G.E., Salakhutdinov, R.R.: Reducing the dimensionality of data with neural networks. Science **313**(5786), 504–507 (2006)
17. Ke, Y., Sukthankar, R.: Pca-sift: A more distinctive representation for local image descriptors. In: Proc. CVPR, vol. 2, pp. II–506. IEEE (2004)
18. Koehn, P.: Statistical significance tests for machine translation evaluation. In: EMNLP, pp. 388–395. Citeseer (2004)
19. Lienhart, R., Maydt, J.: An extended set of haar-like features for rapid object detection. In: International Conference on Image Processing, vol. 1, pp. I–900. IEEE (2002)
20. Lippmann, R.P.: Review of neural networks for speech recognition. Neural Computation **1**(1), 1–38 (1989)
21. Liu, D.C., Nocedal, J.: On the limited memory bfgs method for large scale optimization. Mathematical Programming **45**(1-3), 503–528 (1989)
22. Lu, J., Zhou, X., Tan, Y.P., Shang, Y., Zhou, J.: Neighborhood repulsed metric learning for kinship verification. Pattern Analysis and Machine Intelligence, IEEE Transactions on **36**(2), 331–345 (2014)
23. Luenberger, D.G.: Introduction to linear and nonlinear programming, vol. 28. Addison-Wesley Reading, MA (1973)
24. Ojala, T., Pietikainen, M., Maenpaa, T.: Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. Pattern Analysis and Machine Intelligence, IEEE Transactions on **24**(7), 971–987 (2002)
25. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning internal representations by error propagation. Tech. rep., DTIC Document (1985)
26. Turk, M.A., Pentland, A.P.: Face recognition using eigenfaces. In: Proc. CVPR, pp. 586–591. IEEE (1991)
27. Weinberger, K.Q., Blitzer, J., Saul, L.K.: Distance metric learning for large margin nearest neighbor classification. In: Advances in Neural Information Processing Systems, pp. 1473–1480 (2005)
28. Zhang, Z., Lyons, M., Schuster, M., Akamatsu, S.: Comparison between geometry-based and gabor-wavelets-based facial expression recognition using multi-layer perceptron. In: IEEE International Conference on Automatic Face and Gesture Recognition, pp. 454–459. IEEE (1998)
29. Zheng, L., Idrissi, K., Garcia, C., Duffner, S., Baskurt, A.: Triangular Similarity Metric Learning for Face Verification. In: 11th IEEE International Conference on Automatic Face and Gesture Recognition (2015)