Hybrid Deep Learning for Face Verification

Yi Sun, Xiaogang Wang, *Member, IEEE*, and Xiaoou Tang, *Fellow, IEEE*

Abstract—This paper proposes a hybrid convolutional network (ConvNet)-Restricted Boltzmann Machine (RBM) model for face verification. A key contribution of this work is to learn high-level relational visual features with rich identity similarity information. The deep ConvNets in our model start by extracting local relational visual features from two face images in comparison, which are further processed through multiple layers to extract high-level and global relational features. To keep enough discriminative information, we use the last hidden layer neuron activations of the ConvNet as features for face verification instead of those of the output layer. To characterize face similarities from different aspects, we concatenate the features extracted from different face region pairs by different deep ConvNets. The resulting high-dimensional relational features are classified by an RBM for face verification. After pre-training each ConvNet and the RBM separately, the entire hybrid network is jointly optimized to further improve the accuracy. Various aspects of the ConvNet structures, relational features, and face verification classifiers are investigated. Our model achieves the state-of-the-art face verification performance on the challenging LFW dataset under both the unrestricted protocol and the setting when outside data is allowed to be used for training.

Index Terms—Convolutional networks, deep learning, face recognition

1 Introduction

Face recognition has gained great progress in recent years due to better design and learning of features [1], [4], [10], [13], [17], [33], [37], [47], [54], [55], [56], [58], [61] and face recognition models [9], [12], [14], [25], [28], [49], [52], [53], [57], [60]. This paper focuses on the task of face verification, which aims to determine whether two face images belong to the same identity. This problem is challenging when faces are acquired in unconstrained conditions, given their large intra-personal variations in poses, illuminations, expressions, ages, makeups, and occlusions. Existing methods generally address the problem in two steps: feature extraction and recognition. In feature extraction, existing approaches design or learn features from each individual face image separately to acquire a better representation, while recognition is to calculate the similarity score between two compared faces by using the feature representation of each face. Since face recognition involves comparing two faces, it is interesting to explore the usefulness of features jointly, instead of separately, extracted from two faces in comparison. These features would reflect the relation between the two compared faces and may be easier for the following face recognition model to calculate similarity scores.

Deep models have been popular in computer vision in recent years [6], [15], [19], [21], [22], [32], [35], [45]. With large learning capacity, they can learn feature representations from large-scale data and model complex data variations. Therefore, they are suitable to address the large face variations in unconstrained conditions. We propose to learn relational features with a hybrid deep network. A high-level illustration of our model is shown in Fig. 1. The lower part of our model contains multiple deep convolutional networks (deep ConvNets) [36], each of which takes two face regions in comparison. By taking the two face regions as two input maps of the deep ConvNet, its first convolutional layer compares the corresponded local areas between the two faces to extract the initial low-level relational features with learned filter pairs. These low-level features are further processed through multiple feature extraction stages to extract the high-level relational features, which more explicitly reflect the identity relations of the two compared faces. These high-level relational features are readily to be used for the final face recognition model.

Recent high-performing face recognition algorithms typically extract their features densely in locations and scales [9], [13], [25], [47], which indicates that face regions in different locations and scales contain rich complementary information. For this reason, we extract the high-level relational features from various face region pairs. Each region pair covers a particular region of the two faces, from which a particular set of ConvNets are learned for feature extraction. Unlike [49] which used the ConvNet output vector as features, we propose to use the last hidden layer neuron activations as features instead. This is because features in the last hidden layer are pooled to a single output element of the output vector. In this process feature dimensions are reduced significantly and the output feature may lose much discriminative information. We further concatenate the last hidden layer features of multiple ConvNets to form the high-dimensional relational features, which would contain much richer information related to the identity relations compared to [49] while still being high-level.

The extracted high-level relational features already reflect the identity similarities to a certain degree, which makes it easier for the face recognition model to compute

- *Y. Sun and X. Tang are with the Department of Information Engineering, The Chinese University of Hong Kong, Hong Kong, China. E-mail: {sy011, xtang}@ie.cuhk.edu.hk.*
- *X. Wang is with the Department of Electronic Engineering, The Chinese University of Hong Kong, Hong Kong, China. E-mail: xgwang@ee.cuhk.edu.hk.*

Manuscript received 6 Sept. 2014; revised 9 July 2015; accepted 29 Nov. 2015. Date of publication 2 Dec. 2015; date of current version 12 Sept. 2016. Recommended for acceptance by A. Vedaldi.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TPAMI.2015.2505293

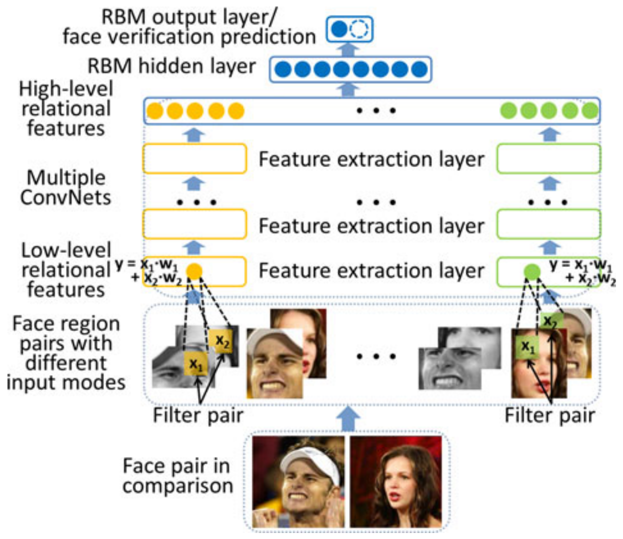


Fig. 1. The hybrid ConvNet-RBM model. The blue arrows show the forward propagation directions. Best viewed in color.

the final face similarity scores. So we choose a shallow classification restricted Boltzmann machine (classification RBM) [34] model for the final face verification. In previous methods, feature extraction and recognition are generally two separate stages, which cannot be jointly optimized. Once useful information is lost in feature extraction, it cannot be recovered in recognition. On the other hand, without the guidance of recognition, the best way to design feature descriptors to capture identity information is not clear. In contrast, our multiple ConvNets and RBM can be unified into a single hybrid deep network, which conducts feature extraction and face verification simultaneously. The parameters of the entire network can be jointly optimized for the task of face verification. Compared with the conference version [49] of this work, the new contributions of this submission are summarized below.

- High-dimensional high-level relational features for face verification, which significantly out-perform the low-dimensional version in [49].
- Detailed validation of the use of the ConvNet structures, high-dimensional features, and face verification classifiers.
- State-of-the-art performance on LFW under both the unrestricted protocol and the setting when outside data is allowed to be used for training.

2 Related Work

Existing methods for face verification extract features from two faces in comparison separately, and then computes the similarity scores between the features. The low-level hand-crafted features are commonly used in face recognition as the first step of feature extraction [2], [4], [5], [7], [8], [9], [12], [13], [20], [25], [33], [37], [38], [42], [47], [56], like LBP and its variants, SIFT, and Gabor. Some methods learned features in an unsupervised way using models like decision trees [10], [59], convolutional DBN [26], or sparse coding [16]. Both the hand-crafted features and features learned in this way encode large intra-personal variations. While they could be used for those face verification models that explicitly models the intra- and inter-personal variations [9], [12], [13], [38], it is inadequate and generally need to be further processed for other models like SVMs or those directly computing the Euclidean or cosine distances.

Ideally, features extracted from individual faces should be discriminative between different identities while consistent for the same identity. For this purpose, a variety of metric learning methods may be applied to further process the initially extracted features [2], [7], [8], [16], [20], [25], [26], [42], [47], or learn the metric directly from pixels by deep learning [14], [24], [41]. Another way is to learn identity-related features [4], [5], [30], [33], [50], [51], [56], [59], [61]. These identity-related features are outputs of classifiers which distinguish different identities or predict the identity-related attributes. Among these methods, Zhu et al. [61] and Kan et al. [30] learned to predict faces of frontal poses for each person. In this way it effectively distinguished multiple identities. Taigman et al. [51] and Sun et al. [50] learned identity features with the face identification task as the supervisory task, while our approach learns the identity features with the verification task as the supervisory signals. Both supervisory signals are complementary and could be combined. The other methods learned binary classifiers to distinguish a person with the background [33], [56], [59], every two people [4], [5], or the binary attributes [33]. In contrast to all the previous methods which learned features from each face separately to reflect the identity of each individual face, our relational features are jointly extracted from two compared faces to reflect the identity relations of the two faces.

Our work is derived from [49] but with significant improvement. In [49], ConvNet outputs are used as features. The output of each ConvNet is a single similarity score of the two input face regions it compares. It suffers from severe information loss since from the last hidden layer to the output, the feature dimension is reduced for 80 times. [49] further pooled the features with high correlations, resulting in a short 12 dimensional feature vector, before passing them to the final face verification model. Though pooling improves prediction accuracies when looking at each individual feature, it further loses the discriminative information and the face verification model learned from the combination of pooled features may be even worse than that learned from the original features. With the above considerations, we take the last hidden layer, instead of the output layer, neuron activations as features without pooling. After concatenating features from multiple ConvNets with various face region pairs under multiple input modes as input (explained in Section 3.1), we form a $38{,}400$-dimensional high-level relational feature vector, which significantly outperforms the low-dimensional version in [49]. Concatenating low-level hand-crafted features extracted from dense locations and multiple scales have been the starting point of many high-performance face recognition algorithms [9], [13], [25], [47], while there are relatively few studies of concatenating high-level features in a similar way. We give the first attempt based on the high-level relational features.

Deep learning has been extremely successful in computer vision since the large learning capacities of deep models meet the abundant data and computing resources gradually available in recent years. It has significantly improved previous state-of-the-art in a variety of challenging tasks [6], [15], [31],

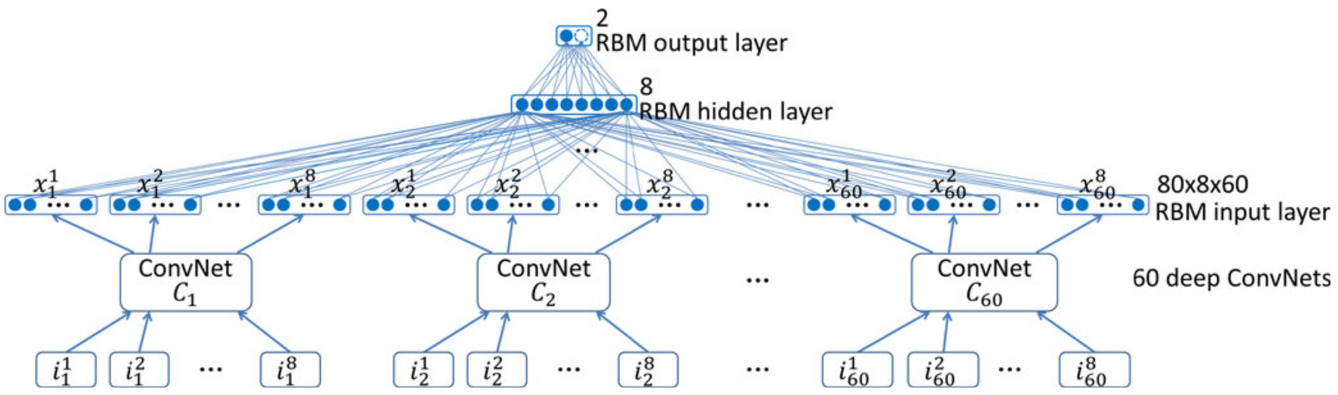

2
RBM output layer
8
RBM hidden layer
x_1^1 x_1^2 x_1^8 x_2^1 x_2^2 x_2^8 x_{60}^1 x_{60}^2 x_{60}^8
80x8x60
RBM input layer
ConvNet C_1
ConvNet C_2
ConvNet C_{60}
60 deep ConvNets
i_1^1 i_1^2 i_1^8 i_2^1 i_2^2 i_2^8 i_{60}^1 i_{60}^2 i_{60}^8

Fig. 2. Architecture of the hybrid ConvNet-RBM model. For the convenience of illustration, we show the input and output layers of the RBM as two separate layers instead of a single visible layer.

[32], [35], [43], [46]. It is also becoming popular in face recognition [14], [26], [30], [41], [44], [49], [50], [51], [61] and other face related tasks [39], [40], [48], where deep models have demonstrated strong abilities in modeling large face variations due to poses, illuminations, occlusions, and other factors in unconstrained conditions [30], [39], [40], [48], [49], [61].

The multi-column deep neural network proposed by Ciresan et al. [15] is one of the earliest study of using deep ConvNet ensembles to improve the performance over a single deep ConvNet. In their work the classification scores of multiple deep ConvNets are averaged to make the final decision. Deep ConvNets in an ensemble could take either the same or different input images. The deep ConvNet assenbles were also successfully applied to ImageNet classification by Krizhevsky et al. [32]. The assembling of 60 deep ConvNets in our proposed deep model is partially inspired by these early works. One important improvement is that, instead of simply averaging the ConvNet output scores, we merge the last hidden layer features by further learning a classifier such as the RBM.

Most existing face recognition pipelines include two or more processing steps, each of which is either hard-coded or optimized independently. For example, Cao et al. [9], [12], [13] fist extracted hand-crafted features, then did PCA, and finally learned the joint Bayesian model for face verification. Simonyan et al. [47] first extracted hand-crafted features, then encoded them into fisher vectors, and finally learned a linear metric for face verification. In [4], [5], [33] hand-crafted features were first extracted, and then identity-related features were learned as the outputs of multiple identity/attribute classifiers. Finally face verification was performed with an additional SVM classifier based on these identity-related features. The common problem in the existing pipelines is that their feature extraction process lacks the guidance of the face verification target. In contrast, our hybrid ConvNet-RBM model can be jointly optimized after pre-training each part separately. Therefore, the feature extraction and face verification modules can best cooperate with each other, which further improves its performance.

3 Hybrid ConvNet-RBM Model

The following sections explain the structures of the hybrid ConvNet-RBM model and its learning procedure. Section 3.1 gives the big picture of the whole system. The detailed explanations of the ConvNets and RBM are given in Sections 3.2 and 3.3, respectively. Section 3.4 discusses their interactions.

3.1 Architecture Overview

Fig. 2 is an overview of our hybrid ConvNet-RBM model, which contains two processing steps, i.e., extracting relational features with multiple deep ConvNets and face verification with the classification RBM [34]. In the first step we use 60 deep ConvNets, each extracting relational features hierarchically from a particular pair of aligned face regions from two images in comparison. The input region pairs for different ConvNets differ in locations, scales, and color channels to make their features complementary. Part of the 60 regions taken from a particular face is shown in Fig. 4. Each region pair further generates eight different input modes by exchanging the two regions and horizontally flipping each region as shown in Fig. 5. When the eight input modes i_n^k for $k = 1, 2, \ldots, 8$ of the nth region pair are input to the nth deep ConvNet C_n, respectively, eight 80-dimensional high-level relational feature vectors x_n^k for $k = 1, 2, \ldots, 8$ are generated. When concatenating the feature vectors extracted from all the 60 region pairs with all the eight input modes for each pair, we get a long $80 \times 8 \times 60 = 38{,}400$-dimensional feature vector. To make full use of the discriminative information it contains, we do not pool the features as did in [49]. Instead, the high-dimensional feature vector is directly used for face verification.

Face verification is conducted by a classification RBM, which takes the 38,400-dimensional relational feature vector as its input layer, followed by a short eight-dimensional hidden layer and a two-class probability output layer indicating whether the two compared faces are from the same person. The deep and wide network structure means that our model has a high capacity. Directly optimizing the whole network would lead to over-fitting. Therefore, we first train each ConvNet separately under the supervision of whether two faces in comparison belong to the same person. Then, by fixing all the ConvNets, the RBM is trained. These two steps initialize the model to a good point. Finally, the whole network is fine-tuned by back-propagating errors from the top-layer RBM to all the lower-layer ConvNets.

3.2 Deep ConvNets

Our deep ConvNets contain four convolutional layers (the first three followed by max-pooling) and two fully-connected

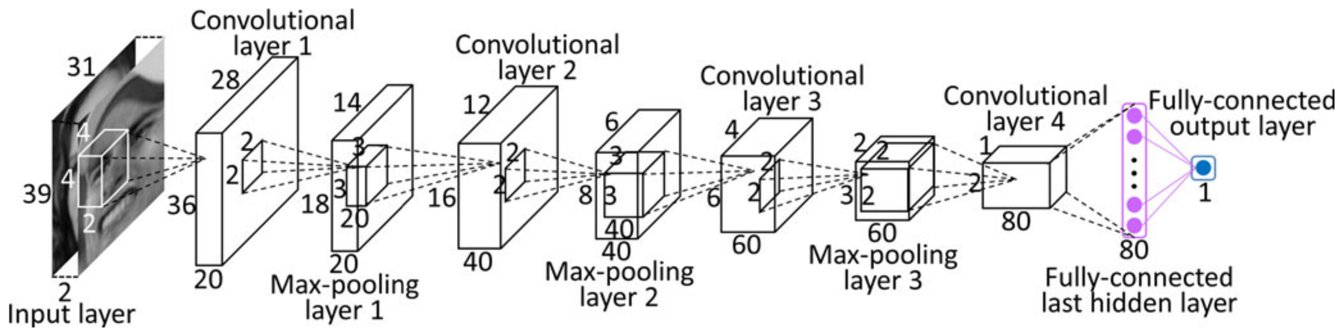


Fig. 3. Structure of one ConvNet. The map numbers and sizes are illustrated as the length, width, and height of cuboids for the input layer and all the convolutional and max-pooling layers, respectively. The 3D convolution kernel sizes of the convolutional layers and the 2D pooling region sizes of the max-pooling layers are shown as the small cuboids and squares inside the large cuboids of maps, respectively. Neuron numbers of the fully-connected layers are marked beside each layer.

layers. Fig. 3 shows the structure of one particular ConvNet with two gray face regions as input. When the size of the input region changes, the map sizes in the following layers of the ConvNets will change accordingly. The last hidden layer contains the high-level relational features to be learned for face verification. The single output neuron is used to add the binary supervision signals for face verification when pre-training each ConvNet separately, and is discarded later when incorporating the ConvNet into the hybrid network.

Input maps. We detect five facial landmarks, i.e., the two eye centers, the nose tip, and the two mouth corners, with the facial point detection method in [48]. Faces are first aligned by similarity transformation according to the two eye centers and the mid-point of the two mouth corners. We then crop two kinds of regions from the aligned faces. One is called the global regions, which are cropped from fixed regions on faces as did in [49]. The other is the local regions, which are centered around one of the five facial landmarks similar to [13]. For either the global or local regions, there are three scales, two different types of colors (RGB or gray), and five different regions for each scale and type of color, making a total of 60 different regions. The color regions are illustrated in Fig. 4. A pair of gray regions forms two input maps of the ConvNet, while a pair of color regions forms six input maps, replacing each gray map with three maps from RGB channels. The region pairs are resized to $39 \times 31 \times k$, $31 \times 39 \times k$, or $31 \times 31 \times k$, depending on their original shapes, before input to the ConvNets, where $k = 2, 6$ for gray and color region pairs, respectively. To precisely describe the ConvNet structure, we take symbols similar to [48]. In particular, the input layer is denoted by $I(h, w, t)$, where h and w are the height and width of the input region, and t is the number of input maps. The input regions are stacked into multiple maps instead of being concatenated to form one map, which enables the ConvNet to model the relations between the two regions from the first convolutional layer.

Convolution. The operation in each convolutional layer can be expressed as

$$y^{j(r)} = f\left(b^{j(r)} + \sum_i k^{ij(r)} * x^{i(r)}\right), \tag{1}$$

where $*$ denotes convolution, x^i and y^j are the ith input map and the jth output map respectively, k^{ij} is the convolution kernel (or filter) connecting the ith input map and the jth output map, and b^j is the bias for the jth output map. $f(\cdot)$ is a non-linear activation function, which is operated element-wise for each neuron. We take $\max(0, \cdot)$ as the non-linear function, which has been shown to have better fitting abilities than the traditionally used $tanh(\cdot)$ [32]. Neurons with such non-linearities are called rectified linear units (ReLU) [32]. Moreover, weights of neurons (including convolution kernels and biases) in the same map may be locally shared, as suggested by Huang et al. [26]. The superscript r in Equation (1) indicates a local region where weights are shared. Since faces are structured objects, locally sharing weights in higher layers allows the network to learn different high-level features in different locations. A convolutional layer is denoted by $C(s, n, p, q)$. s is the convolution kernel size (the side length of the square kernels k^{ij}). n is the

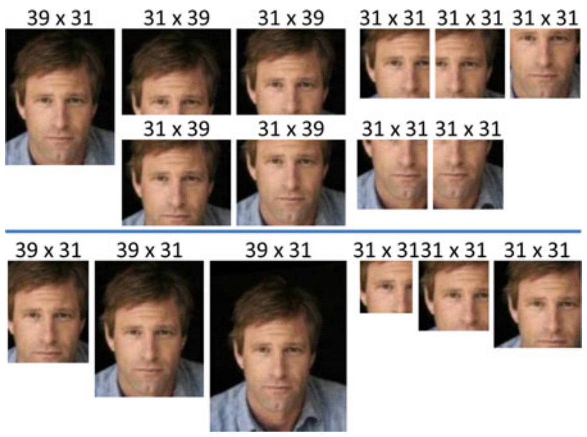


Fig. 4. Top: 10 color face regions of medium scales. The five regions in the top left and top right are the so-called global and local regions, respectively. Bottom: three scales of two particular regions. The region sizes, after being re-scaled to fit the ConvNet input dimensions, are marked above each region.

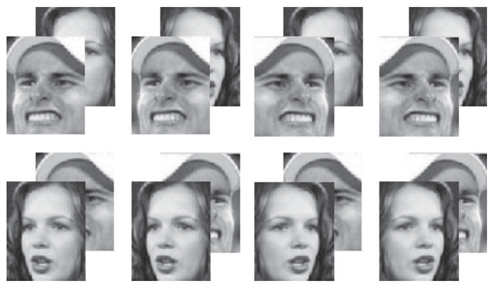


Fig. 5. Eight possible input modes for a pair of face regions.

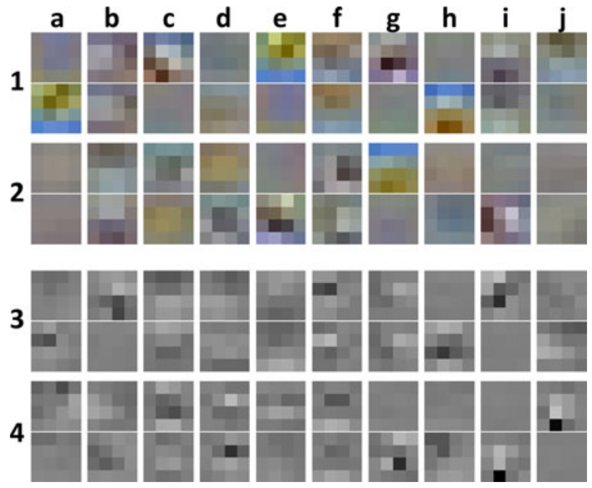


Fig. 6. Examples of the learned $20\ 4 \times 4$ filter pairs of the first convolutional layer of ConvNets taking color (line 1,2) and gray (line 3,4) input region pairs, respectively. The upper and lower filters in each pair convolve with the two face regions in comparison, respectively, and the results are added. We use line 1-4 and column a-j to indicate each filter pair. For filter pairs in which one filter varies greatly while the other remains near uniform (1c, 1g, 1h, 2e, 2i, 3h, 3i, 4g, 4i, 4j), features are extracted from the two input regions separately. For those filter pairs in which both filters vary, some kind of relations between the two input regions are extracted. Among the latter, many filter pairs extract simple relations such as addition (1b, 1f, 3g, 4f) or subtraction (1i, 2b, 2c, 2d, 2f, 3c, 3e, 3f, 4b, 4c, 4d), while others extract more complex relations. Interestingly, we find that filters in some filter pairs are similar to those in some others, except that the order of the two filters are inversed (1a versus 1e, 1h versus 2g, 2c versus 2d, 1i versus 2b, 3j versus 4b, 4i versus 4j). This makes sense since face similarities should be invariant with the order of the two face regions in comparison. Best viewed in color.

number of maps. p and q are weight sharing parameters. Each map in the convolutional layer is evenly divided into p by q regions, and weights are locally shared in each region. We find that locally sharing weights in higher convolutional layers can significantly improve the fitting and generalization abilities of ConvNets.

Convolution is the feature extraction process, through which relations between the two face regions are modeled hierarchically. For example, operations in the first convolutional layer (omitting the superscript r) can be reformulated as

$$y^j = f\big(b^j + k^{1j} * x^1 + k^{2j} * x^2\big), \tag{2}$$

where x^1 and x^2 denote the two face regions compared, which are convolved by the two kernels k^{1j} and k^{2j}, respectively, and the results are added. So y^j reflects a kind of relation between the two face regions x^1 and x^2. The relation type is decided by the two kernels k^{1j} and k^{2j}. See Fig. 6 for examples. As the network goes deeper, more global and higher-level relations between the two regions are modeled. These high-level relational features make it possible for the top layer output neuron to predict the high-level concept of whether the two input regions come from the same person.

Pooling. The first three convolutional layers are followed by max-pooling for feature reduction and increasing their robustness to distortions of face images. Max-pooling is formulated as

$$y^i_{j,k} = \max_{1 \le m,n \le s} \left\{ x^i_{(j-1)\cdot s+m,\,(k-1)\cdot s+n} \right\}, \tag{3}$$

where each neuron in the ith output map y^i pools over an $s \times s$ non-overlapping local region in the ith input map x^i. Pooling layer is denoted by $P(s)$.

Full connection. The fourth convolutional layer is followed by two successive fully-connected layers. A fully-connected layer with n neurons is denoted by $F(n)$ with function

$$y_j = f\left(\sum_{i=1}^{m} x_i \cdot w_{i,j} + b_j \right), \tag{4}$$

for $j = 1, \ldots, n$, where n and m are neuron numbers of the current and previous layers, respectively. $f(\cdot)$ is an element-wise nonlinear function. We use ReLU nonlinearity for the first fully-connected layer (of the relational features) as did for the previous convolutional layers, and tanh nonlinearity for the single output neuron to obtain a probability output of face similarities. The ConvNet parameters are initialized by small random numbers and learned by minimizing the squared loss $\frac{1}{2}(o-t)^2$, where o is the ConvNet output and $t = \pm 1$ is the binary face verification target. The loss is minimized by stochastic gradient descent, where the gradient is calculated by back-propagation [36].

3.3 Classification RBM

Classification RBM models the joint distribution between its output neurons y (one out of C classes), input neurons x (binary), and hidden neurons h (binary), as $p(y,x,h) \propto e^{-E(y,x,h)}$, where $E(y,x,h) = -h^\top Wx - h^\top Uy - b^\top x - c^\top h - d^\top y$. Given input x, the conditional probability of its output y can be explicitly expressed as

$$p(y_i \,|\, x) = \frac{e^{d_i} \prod_j \Big(1 + e^{c_j + U_{ji} + \sum_k W_{jk} x_k}\Big)}{\sum_i e^{d_i} \prod_j \Big(1 + e^{c_j + U_{ji} + \sum_k W_{jk} x_k}\Big)}, \tag{5}$$

where $i = 1, \ldots, C$ are class indices. We discriminatively train the Classification RBM by minimizing $-\log p(y_t \,|\, x)$, where t is the target class, which is also optimized by stochastic gradient descent as the ConvNets. Due to the closed form expression of the likelihood, the gradient $-\frac{\partial \log p(y_t \,|\, x)}{\partial \theta}$ can be computed exactly, where $\theta \in \{W, U, c, d\}$ are parameters to be learned.

3.4 Fine-Tuning the Entire Network

The single ConvNet output is discarded when learning the RBM and fine-tuning the entire network. Therefore, each ConvNet C_n maps the nth pair of face regions under the kth input mode i^k_n to an 80-dimensional relational feature vector x^k_n, i.e.,

$$x^k_n = C_n\big(i^k_n\big), \tag{6}$$

for $n = 1, \ldots, 60$ and $k = 1, \ldots, 8$. The feature vectors x^k_n for all n and k are concatenated into a long feature vector x. The RBM takes the feature vector x as input and predicts the class label as

$$\mathrm{argmax}_{i \in \{1,2\}} p(y_i \,|\, x), \tag{7}$$

with $p(y_i \,|\, x)$ defined in Equation (5). The RBM is first trained based on the feature vector x with the ConvNet parameters fixed. Then error is back-propagated from the RBM to all the ConvNets and the whole model is fine-tuned. Let $L = -\log p(y_t \,|\, x)$ be the RBM loss function, where t is the target class, and α_n be the parameters of the nth ConvNet. The gradient of the loss w.r.t. α_n is

$$\frac{\partial L}{\partial \alpha_n} = \sum_{k=1}^{8} \frac{\partial L}{\partial x_n^k} \frac{\partial x_n^k}{\partial \alpha_n} = \sum_{k=1}^{8} \frac{\partial L}{\partial x_n^k} \frac{\partial C_n\left(i_n^k\right)}{\partial \alpha_n}. \tag{8}$$

$\frac{\partial L}{\partial x_n^k}$ is calculated by the closed form expression of L, and $\frac{\partial C_n\left(i_n^k\right)}{\partial \alpha_n}$ is calculated by back-propagation in the ConvNet.

4 Experiments

We evaluate our algorithm on LFW [27], which has been used extensively to evaluate algorithms of face verification in the wild. We conduct evaluation under two different settings: (1) 10-fold cross validation under the unrestricted protocol of LFW without using data outside LFW to train the model, and (2) cross-dataset validation in which a typically larger, external dataset exclusive to LFW is used for training. The identities of people in the external dataset are mutually exclusive to those in LFW. The former shows the performance with a limited amount of training data, while the latter shows the ability to learn from extra data and also the generalization across different datasets. Section 4.1 explains the experimental settings in detail, Sections 4.2 and 4.3 validate various aspects of model design, and Section 4.4 derives our final results and compares them with the previous state-of-the-art results in literature.

4.1 Experiment Settings

LFW is divided into 10 folds of mutually exclusive people sets. For the unrestricted setting, performance is evaluated by using the 10-fold cross-validation. Each time one fold is used for testing and the other nine for training. Results averaged over the 10 folds are reported. The 600 testing pairs in each fold are predefined by LFW and fixed, whereas training pairs can be generated using the identity information in the other nine folds and the number is not limited. We refer to this as the LFW training setting.

For the cross-dataset setting, we use outside data exclusive to LFW for training. PubFig [33] and WDRef [12] are two large datasets other than LFW with faces in the wild. However, PubFig only contains 200 people with a large number of images for each person. Therefore the inter-personal variations are quite limited. WDRef contains both a large number of identities and a modest number of images for each identity, but the images or image URLs are not publicly available. Accordingly, we created a new dataset, called the Celebrity Faces dataset (CelebFaces) [49]. It contains 87,628 face images of 5,436 celebrities from the web, and was assembled by first collecting the celebrity names that do not exist in LFW, then searching for the face images for each name on the web. To avoid any overlap between the identities in LFW and CelebFaces, we manually checked people in these two datasets with similar face images and removed the identity in CelebFaces which exists in LFW but taking a different name in CelebFaces. The similarity score was calculated by a preliminary version of our deep model. To conduct cross-dataset evaluation, the model is trained on CelebFaces and tested on the predefined 6,000 test pairs in LFW. We refer to this setting as the CelebFaces training setting.

For both settings, we randomly choose 80 percent people from the training data to train the deep ConvNets, and use the remaining 20 percent people to train the top-layer RBM and fine-tune the entire model. Since stochastic gradient descent is used, we keep randomly generating the positive and negative face pairs for training. The positive training pairs are generated such that, for each person, the sample number is linearly proportional to the number of images available for that person. Generating in this way prevents sample pairs generated from people with much more images than average dominating the data, since the total number of positive pairs that could be generated by one person is the square of the number of images. We generated an equal number of negative pairs as the positive ones. The ConvNets are first trained independently with eight times the available sample pairs due to the eight possible input modes formed by each pair. Then the RBM is trained and the whole network is fine-tuned with two times the available sample pairs by exchanging the two images in a pair. In test, the two similarity scores calculated by the two original and exchanged face images are averaged to get the final score. Evaluation on the two exchanged faces is achieved by reordering the feature vectors x_n^k (Fig. 2) extracted by each ConvNet without recalculating these features. For example, (x_n^1, x_n^2, x_n^3, x_n^4, x_n^5, x_n^6, x_n^7, x_n^8) would be reordered as (x_n^5, x_n^6, x_n^7, x_n^8, x_n^1, x_n^2, x_n^3, x_n^4), for $n = 1, 2, \ldots, 60$. Then reevaluating the RBM based on the reordered feature vectors.

A separate validation dataset is used during training to avoid over-fitting. After each training epoch, we observe the errors on the validation dataset and select the model that provides the lowest validation error. We randomly select 100 people from the training people to generate the validation data. The free parameters in training (the learning rate and its decreasing rate) are selected using view 1 of LFW[1] and are fixed in all the experiments. We report both the accuracies and the ROC curves. Each face pair is assigned to the class with higher probabilities without further learning a threshold for the final classification. We provide the detailed free parameters in training as following.

First, each of the 60 deep ConvNets are trained independently with an initial learning rate of 0.001. The learning rate is decreased exponentially by multiplying a factor of 0.9 after each iteration. We generated approximately 40 thousand and 240 thousand pairs of training samples per iteration for the LFW and CelebFaces training settings, respectively, using the 80 percent training identities. These sample pairs are then augmented with the eight input modes and fed into the deep ConvNets in a random order. Training takes approximately 10 and 20 iterations for the LFW and CelebFaces training settings, respectively. Weights in the deep ConvNets are randomly initialized with a uniform distribution between $[-0.05, 0.05]$.

1. View 1 is provided by LFW for algorithm development and parameter selection without over-fitting the test data [27].

TABLE 1
Summary of Network Structures

	L0	L1	L2	L3	L4	L5	L6	L7	L8	L9
S0	I(39,31,k)	C^r(4,20,1,1)	P(2)	C^r(3,40,2,1)	P(2)	C^r(3,60,3,2)	P(2)	C^r(2,80,2,1)	F^r(80)	F^t(1)
	I(31,39,k)	C^r(4,20,1,1)	P(2)	C^r(3,40,1,2)	P(2)	C^r(3,60,2,3)	P(2)	C^r(2,80,1,2)	F^r(80)	F^t(1)
	I(31,31,k)	C^r(4,20,1,1)	P(2)	C^r(3,40,1,1)	P(2)	C^r(3,60,2,2)	P(2)	C^r(2,80,1,1)	F^r(80)	F^t(1)
S1	I(39,31,k)	C^r(4,20,1,1)	P(2)	C^r(3,40,2,1)	P(2)	C^r(3,60,3,2)	P(2)	F^r(80)	F^t(1)	
S2	I(39,31,k)	C^r(4,20,1,1)	P(2)	C^r(3,40,2,1)	P(2)	F^r(80)	F^t(1)			
S3	I(39,31,k)	C^r(4,20,1,1)	P(2)	F^r(80)	F^t(1)					
S4	I(39,31,k)	C^r(4,20,1,1)	P(2)	C^r(3,40,1,1)	P(2)	C^r(3,60,1,1)	P(2)	C^r(2,80,1,1)	F^r(80)	F^t(1)
S5	I(39,31,k)	C^t(4,20,1,1)	P(2)	C^t(3,40,2,1)	P(2)	C^t(3,60,3,2)	P(2)	C^t(2,80,2,1)	F^t(80)	F^t(1)
S6	I(39,31,k)	C^a(4,20,1,1)	P(2)	C^a(3,40,2,1)	P(2)	C^a(3,60,3,2)	P(2)	C^a(2,80,2,1)	F^t(80)	F^t(1)

After the deep ConvNets are trained, we continue to train the classification RBM with an initial learning rate of 0.01, which is exponentially decreased by multiplying a factor of 0.7 after each iteration. We generated approximately eight thousand and 50 thousand training sample pairs per iteration for the LFW and CelebFaces training settings, respectively, using the remaining 20 percent training identities. Then we extract the $38{,}400$ dimensional relational features using the 60 pre-trained deep ConvNets. The $38{,}400$ dimensional feature vectors are augmented by reordering the features, which corresponds to exchanging the order of the two face images in a pair, as has been discussed. The high-dimensional relational feature vectors are fed into the classification RBM in a random order. The weight matrix W and U in RBM (see Equation (5)) are initialized randomly with a uniform distribution between $[-\frac{1}{\sqrt{m}}, \frac{1}{\sqrt{m}}]$, where m is the maximum between the number of rows and columns of the matrix [34]. The bias c and d are initialized with zeros. Training takes approximately five iterations for both settings.

Finally the deep ConvNets and the RBM are jointly fine-tuned. During the fine-tuning, the RBM takes the same initial learning rate and its decreasing rate as in the pre-training. For the deep ConvNets, we take an initial learning rate of 0.0006 and decreases by a factor of 0.7 after each iteration. The training sample pairs are generated in the same way as in pre-training the RBM for every iteration. The hybrid ConvNet-RBM model is initialized with the pre-trained weights and training takes approximately five iterations for both training settings.

We use stochastic gradient descent in all training stages. The hybrid ConvNet-RBM model is implemented by C++ code written from scratch by us and is run on a computer cluster of 60 CPUs with MPI parallelism. It takes approximately ten days to train the hybrid model. Testing takes approximately 0.12 second for comparing one pair of face images.

4.2 Deep ConvNet structure

We investigate the impact of network depth, weight sharing schemes, and neuron activation functions to performance. Table 1 summarizes the compared network structures. The three structures in S0 are adopted for our ConvNets, respectively, according to their input region shapes, where $k = 6$ for color pairs and $k = 2$ for gray pairs. We compare different structures S1-S6 to S0 in experiments. For S1-S6, only the network structure with input sizes of $39 \times 31 \times k$ is given to save space. Structures for other input sizes are similar. The superscripts r, t, and a denote the neuron activation functions of ReLU, tanh, and abstanh, respectively, which are used by the corresponding layers.

Each time only one factor is changed and the 60 ConvNets trained on different face region pairs are evaluated separately for face verification. The ConvNet is trained on 80 percent of the CelebFaces images and tested on LFW. The test score of each face pair is derived by averaging the eight scores (given by the output neuron) calculated from the eight different input modes. Since the output neuron predicts directly from the last hidden layer features, better face verification accuracy would mean better hidden features learned.

The eight modes of the input face pair are complementary since our ConvNet is not designed to be commutative to the input face pair, neither is it symmetric with respect to the horizontal flipping of each of the face images in the input face pair. Given the structures in S0, the mean face verification accuracy of the 60 ConvNets taking a single input mode is 86.63 percent. When averaging the scores of four input modes by allowing the image flipping but not exchange, the mean accuracy increases to 88.01 percent. It is further increased to 88.46 percent when averaging all the eight ConvNet output scores from the eight input modes by allowing both image flipping and exchange. The complementarity of ConvNet output scores also implies the complementarity of the hidden feature representations.

Network depth. We compare ConvNets with increasing depth from containing one to four convolutional layers. The resulting network structures are S3, S2, S1, and S0 as shown in Table 1. S0 is the structure used by our system. The averaged accuracies are shown in Table 2. Adding the kth convolutional layer increases 1.79, 0.57, and 0.35 percent

TABLE 2
Average Test Accuracies for ConvNets Taking Structures S0-S6 Specified in Table 1

structure	average accuracy (%)
S0	**88.46**
S1	88.11
S2	87.54
S3	85.75
S4	86.95
S5	87.41
S6	83.77

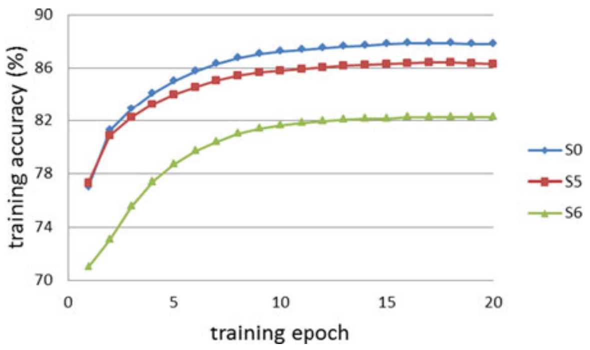


Fig. 7. Training accuracies w.r.t. the number of training epoches, averaged over the 60 ConvNets trained on the 60 different region pairs, with ReLU (S0), tanh (S5), and abstanh (S6) neuron activation functions, respectively. Best viewed in color.

accuracies on average, respectively, for $k = 2, 3, 4$, which verifies the deep structures used. It also shows that larger improvement is achieved in earlier convolutional stages.

Weight sharing. Our ConvNets locally share weights in higher convolutional layers. S0 specifies the weight sharing scheme taken by our ConvNets. In general, the higher the layers, the fewer the neurons which share weights, which helps to learn diverse high-level features. In the first two convolutional layers of S0, weights are either globally shared or shared by halves of the feature maps. In the third convolutional layer, weights are shared among neurons in every 2×2 local region. In the last convolutional layer (more appropriately called the locally-connected layer), weights are completely unshared. We compare S0 to S4, which globally share weights in all convolutional layers. Table 1 shows the performance gap between S0 and S4. Local weight sharing increases 1.51 percent face verification accuracy on average.

Activation function. We evaluate three activation functions proposed in literature to replace the function $f(\cdot)$ in Equations (1), (2), and (4), that is, the ReLU $\max(0, \cdot)$ [32], the hyperbolic tangent $tanh(\cdot)$ [36], and the absolute value rectified hyperbolic tangent $abs(tanh(\cdot))$ [29]. See S0, S5, and S6 in Table 1 for the corresponding structures. Table 1 shows their differences in performance. ConvNets with ReLU activation functions perform 1.05 and 4.69 percent better on average than those with tanh and abstanh activation functions, respectively.

The performance differences between the three nonlinear functions are due to their differences in fitting abilities. ReLU has the strongest fitting ability because its function value is unbounded while the other two functions are bounded. abstanh has the weakest fitting ability, even weaker than tanh, due to the absolute value rectification. To illustrate this, we plot the average verification accuracies on the training data w.r.t. the training epochs (one training epoch is a single pass of a given set of training samples) in Fig. 7, where higher accuracies mean better fitting abilities. The figure confirms what we have argued above. Since the number of face pairs formed for training could be potentially large (proportional to the squared image number), over-fitting is unlikely to occur given a moderately large training set, e.g., the CelebFaces. Therefore, the fitting ability and test accuracy are strongly correlated.

Regions versus accuracy. The face verification accuracies of individual ConvNet vary for different input face regions.

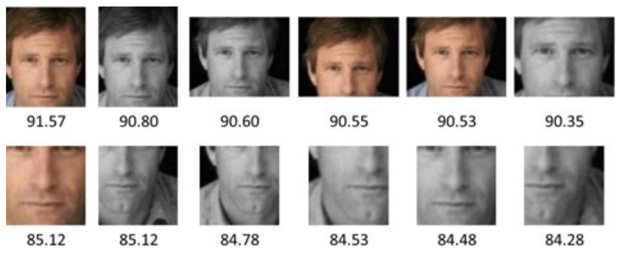


Fig. 8. Face regions with the highest and lowest face verification accuracies, respectively.

Fig. 8 shows the six input face regions from which the ConvNet gives the highest face verification accuracies (the first row) as well as the six face regions with the lowest face verification accuracies (the second row). The accuracy is labeled below each face region, which is calculated from the averaged scores of the eight input modes. The ConvNets take structures in S0. It is shown that the eye regions are more critical in face recognition than the nose and mouth regions. Regions covering the two eyes have over 90 percent face verification accuracies, while those excluded the eye regions only have approximately 85 percent verification accuracies.

4.3 Learning From the Relational Features

We investigate the discriminative power and complementarity of the relational features, as well as the performance of a few commonly used classifiers learned on these features. The ConvNets extracting these features are pre-trained and their weights are fixed without jointly fine-tuning the entire network. Classifiers in this section are trained with the remaining 20 percent CelebFaces images and tested on LFW. In each experiment the classifier training process is repeated for five times. For each time we randomly choose the validation people from the training people, and randomly generate the training/validation sample pairs. The averaged test results over the five trials are reported.

Feature complementarity. We validate that the relational features extracted from different face region pairs are complementary, and evaluate how much performance gain could be obtained by combining those features. We discriminatively train the classification RBM for face verification, based on the relational features extracted from n region pairs, for $n = 1$, 5, 15, 30, and 60, respectively. For each region pair, given its eight different input modes, eight 80-dimensional feature vectors are extracted from the last hidden layer of ConvNets. All the $8n$ feature vectors from the n region pairs are concatenated to a long feature vector for face verification. When n increases from 1 to 5, 15, 30, and 60, different variations are added in sequence. For $n = 1$, we experiment with features from each region pair, respectively. For $n = 5$, we concatenate features from region pairs in different areas, but with the same color type (RGB or gray), scale (small, medium, or large), and region type (global or local regions). For example, the five regions shown in the top lest or top right in Fig. 4. For $n = 15$, we further include region pairs with all the three different scales, but still with the same color type and region type. For $n = 30$, we use all the color or gray region pairs. For $n = 60$, all the 60 region pairs are used. Fig. 9 shows that the performance is significantly improved whenever combining features from a richer variety of region pairs. So all different

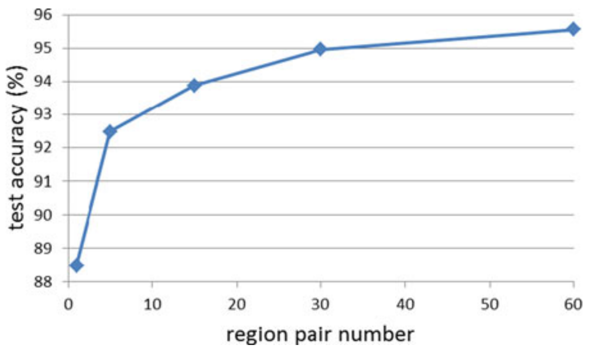


Fig. 9. Average RBM prediction accuracies based on features extracted from 1, 5, 15, 30, and 60 face region pairs. The accuracy is consistently improved when concatenating more features.

region pairs contain additional information. The total growth from $n = 1$ to $n = 60$ is approximately 7 percent. The large improvement is in consistency with that reported for low-level features in [13].

RBM structure. Our classification RBM takes the 38,400-dimensional feature vector as input and outputs a two-dimensional probability distribution over the two classes (being the same person or not), while the number of hidden neurons is undetermined. We choose it by maximizing the face verification accuracy on the validation data. In fact, the RBM is very robust on this parameter and the accuracy keeps almost unchanged in a large range, as shown in Fig. 10. The relational features are already near linearly separable since a linear SVM also gives comparable performance as shown in the next section. Therefore, increasing the number of hidden neurons in RBM, which means using functions which can fit more complex distributions, does not help much. The RBM could be very efficient by using only a few, eight in our case, hidden neurons, since the efficiency is proportional to the product of the number of input and hidden neurons. The input features to RBM are from the rectified linear units, which could be approximated by the sum of an infinite number of binary units with shared weights. Therefore the mathematics derived in Section 3.3 remain unchanged [41].

Feature discriminative power and classifier comparison. We compare a few commonly used classifiers to the classification RBM, including PCA + LDA [3], linear SVM [18], and SVM with RBF kernels [11]. The feature dimension for PCA and the hyper-parameters for SVMs are optimized on the validation data. Moreover, we compare features from the

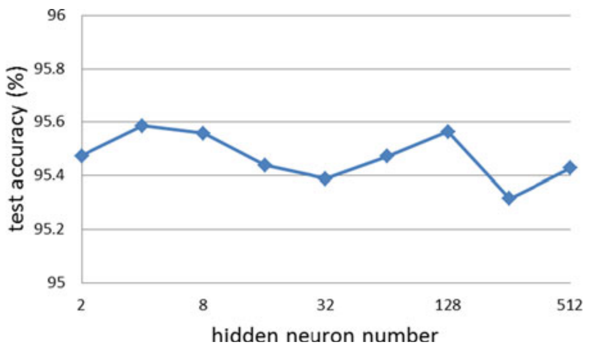


Fig. 10. RBM prediction accuracies on LFW test data w.r.t. the number of hidden neurons. The accuracy keeps almost unchanged.

TABLE 3
Test Accuracies of Various Features and Classifiers

	hid	hid+out	out	out+p1	out+p2
dimension	38,400	38,880	480	60	20
per dim (%)	60.25	60.58	86.63	88.46	89.70
PCA+LDA (%)	94.55	94.42	93.41	93.20	92.60
SVM linear (%)	95.12	95.04	93.45	93.53	92.74
SVM rbf (%)	94.95	94.89	94.00	93.83	92.81
classRBM (%)	**95.56**	95.32	93.79	93.69	93.09

last hidden layer of ConvNets (hid), the output of ConvNets (out), and both of them (hid + out) based on the above classifiers. To make a comparison with [49], we further evaluate the pooled output features, including those pooled from the outputs of the same ConvNet with different input modes, or level one pooling (out + p1), and those pooled from different ConvNets, or level two pooling (out + p2). To acquire more complementary features, we do not train multiple ConvNets with the same input region pairs as did in [49]. Therefore, we pooled the outputs of ConvNets taking similar input region pairs, i.e., those with the three different scales but being the same otherwise. For example, the three regions in the bottom left or bottom right in Fig. 4.

The comparison results are shown in Table 3. It also shows the feature dimensions (dimension) and the average verification accuracy of directly comparing each feature dimension to a threshold (per dim). Based on these results, the following conclusions can be drawn. First, classifiers learned from the output features perform significantly worse than those learned from the high-dimensional last hidden layer features. The performance would decrease further when the output features are pooled. Given the last hidden layer features, there is no additional benefit by adding the output features. This verifies the use of the last hidden layer features, which contain much more discriminative information than the output ones. Pooling the output features further loses the information, even though it increases the accuracy when looking at the individual features. Second, the classification RBM performs better than the other classifiers in most cases, while the other classifiers also perform reasonably well. For example, the simple linear SVM model achieves 95.12 percent accuracy when learned on the last hidden layer features. The relational features extracted by our deep models more explicitly reflect the high-level identity similarities. These features are near linearly separable, which makes the following face verification step easier. We choose the RBM as our face verification model for its superior performance as well as that it can be jointly optimized with the lower ConvNet layers.

4.4 Final Results and Comparison

We evaluate the performance of our final system after fine-tuning and compare it with the state-of-the-art results. The evaluation is conducted on LFW and CelebFaces training settings, respectively.

Fine-tuning. After separately learning each of the 60 ConvNets and the RBM, the whole model is fine-tuned to jointly optimize all the parts. Fine-tuning is conducted on the same 20 percent training data as used to train the RBM, which aims to let the 60 ConvNets to coordinate with each other for better face verification. We find that the LFW

TABLE 4
Accuracy Comparison on Each Learning Step of Our Hybrid ConvNet-RBM Model

	LFW	CelebFaces
Single ConvNet (%)	85.05	88.46
RBM (%)	93.45	95.56
Fine-tuning (%)	93.58	96.60
Model averaging (%)	**93.83**	**97.08**

training data is easily to be over-fitted by our deep model due to the limited face pairs that could be formed. To alleviate over-fitting, we use dropout learning [23] to pre-train the RBM and fine-tune the entire network for the LFW training setting, where 50 percent of the high-dimensional relational features are randomly dropped each time during the online learning process. There is no obvious over-fitting for the CelebFaces training setting and dropout is not used.

Moreover, we find that the performance can be further enhanced by averaging five different hybrid ConvNet-RBM models. This is achieved by first training five RBMs (each with a different set of randomly generated training data) with the weights of the ConvNets pre-trained and fixed, and then fine-tuning each of the whole ConvNet-RBM model separately. Table 4 shows the accuracies of each learning step of our model for the LFW and CelebFaces training settings, respectively, including the (average) accuracies of each single ConvNet by averaging its outputs from the eight different input modes of each compared face pair (single ConvNet), training a classification RBM based on the last hidden layer features of the 60 ConvNets (RBM), fine-tuning the whole hybrid ConvNet-RBM model (fine-tuning), and averaging the predictions of five hybrid ConvNet-RBM models (model averaging). We achieved our best results of **93.83** and **97.08** percent for the LFW and CelebFaces training settings, respectively, with the averaging of five hybrid ConvNet-RBM model predictions (model averaging). The improvement of fine-tuning the entire model on LFW is not obvious due to fast over-fitting of the training data even though dropout learning is used.

To further justify the use of the classification RBM as the top classification layer of our hybrid deep model, we replace the RBM with two perceptron networks and compare their face verification accuracies with that of the original hybrid deep model, respectively. The first is a single-layer perceptron which linearly combines the 38,400-dimensional features of our deep ConvNet assemble followed by a single sigmoid output neuron for face verification. The second is a multi-layer perceptron with a single hidden layer. The dimension of the hidden layer is chosen to be the same as the hidden layer of our classification RBM, i.e., eight neurons, with ReLU nonlinearity, followed by a single sigmoid output neuron.

TABLE 5
Accuracy Comparison between the Proposed Classification RBM and Single- and Multi-Layer Perceptrons for Face Verification in the Hybrid Deep Model

	RBM	single-layer perceptron	multi-layer perceptron
pre-training (%)	95.56	94.32	95.03
Fine-tuning (%)	96.60	95.07	96.02
Model averaging (%)	97.08	95.40	96.60

TABLE 6
The Estimated Mean Accuracy and the Standard Error of the Mean of Our Hybrid ConvNet-RBM Model and the State-of-the-Art Methods under the LFW Unrestricted Protocol

Method	Accuracy (%)
PLDA [38]	90.07 ± 0.51
Sub-SML [8]	90.75 ± 0.64
Joint Bayesian [12]	90.90 ± 1.48
ConvNet-RBM previous [49]	91.75 ± 0.48
VMRS [2]	92.05 ± 0.45
Fisher vector faces [47]	93.03 ± 1.05
High-dim LBP [13]	93.18 ± 1.07
ConvNet-RBM	$\mathbf{93.83 \pm 0.52}$

Table 5 compares the single- and multi-layer perceptron networks to our proposed classification RBM for face verification on LFW under the CelebFaces training settings. The two compared perceptron networks are learned in the same way as the classification RBM. They are first pre-trained by fixing the weights of the deep ConvNets (pre-training). Then the entire hybrid deep model is fine-tuned (fine-tuning). The averaging of five perceptron networks is also compared (model-averaging). It is shown that the non-linearity provided by the hidden layers of the RBM and the multi-layer perceptron improves the performance, compared to the linear single-layer perceptron model. In case of the non-linear models, the classification RBM is better than the multi-layer perceptron. The single-layer perceptron model is inferior to the linear SVM (94.32 versus 95.12 percent without fine-tunining) probably due to different optimization algorithms. However, the linear SVM is not readily to be incorporated into deep neural networks.

Method comparison. We compare our best results on LFW with the state-of-the-art methods in accuracies (Tables 6 and 7) and ROC curves (Figs. 11 and 12) respectively. ConvNet-RBM previous indicates our previous method in [49]. Table 6 and Fig. 11 are comparisons of methods that follow the LFW unrestricted protocol without using outside data to train the model. This protocol is difficult even for models with

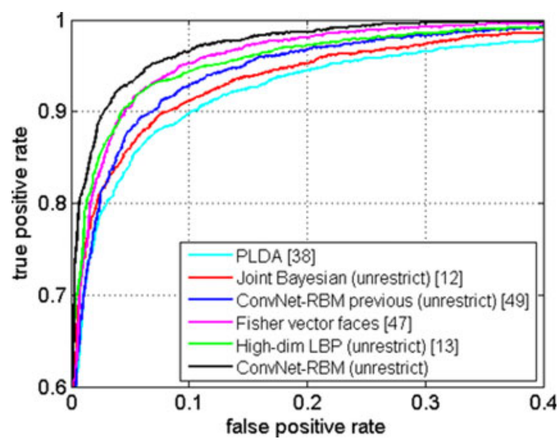


Fig. 11. ROC comparison of our hybrid ConvNet-RBM model and the state-of-the-art methods under the LFW unrestricted protocol. Best viewed in color.

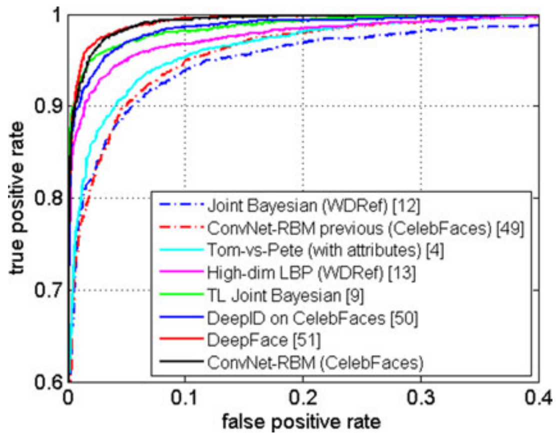


Fig. 12. ROC comparison of our hybrid ConvNet-RBM model and the state-of-the-art methods relying on outside training data. Best viewed in color.

TABLE 7
The Estimated Mean Accuracy and the Standard Error of the Mean of Our Hybrid ConvNet-RBM Model and the State-of-the-Art Methods that Rely on Outside Training Data

Method	Accuracy (%)
Joint Bayesian [12]	92.42 ± 1.08
ConvNet-RBM previous [49]	92.52 ± 0.38
Tom-vs-Pete (with attributes) [4]	93.30 ± 1.28
High-dim LBP [13]	95.17 ± 1.13
TL Joint Bayesian [9]	96.33 ± 1.08
DeepID on CelebFaces [50]	96.05 ± 0.21
DeepFace [51]	97.35 ± 0.25
ConvNet-RBM	**97.08 ± 0.28**

learning capacities due to the limited data in LFW. Therefore, the accuracies of previous methods under this protocol increase much slower than those using outside data to learn. We are the best under this protocol and improve the previous state-of-the-art significantly. Also, all the previous best methods compared, except our previous method, used hand-crafted low-level features. We are the only one using deep models to learn such features.

Table 7 and Fig. 12 report the results when the training data outside LFW is allowed to use. We compare with DeepID [50] by using the same training data (CelebFaces) and the same 60 cropped face regions, and surpass it by a large margin. Although DeepFace [51] is marginally better than our result, it used 4.4 million training data, two orders of magnitude larger than ours. It also used an accurate 67-point 3D face alignment, while we only use five points and do not correct the out-of-plane rotations in the alignment step. Instead, we let our deep networks to learn such pose variations. Most previous best methods compared in Table 7 used hand-crafted features as their base features [4], [9], [12], [13], while deep learning has only shown recent success on this problem [50], [51]. We believe our unified deep network approach will promote more face recognition solutions using deep learning.

5 Conclusion

This paper has proposed a new hybrid ConvNet-RBM model for face verification. The deep ConvNets in this model learn directly and jointly extracts relational visual features from face pairs under the supervision of face identities. Taking the last hidden layer features instead of the output and concatenating features extracted from various face region pairs are essential to form discriminative and complementary features. It it also important to select the appropriate ConvNet structures and face verification classifiers. Both feature extraction and face verification stages are unified under a single deep network architecture and all the components are jointly optimized for the target of face verification. Joint optimization further improves the performance compared to learning each part separately, especially when there is enough training samples. Compared with other models, ours has achieved the best face verification performance on LFW under both the unrestricted protocol and the setting when outside data is allowed to be used for training.

Acknowledgments

This work is partially supported by CUHK Computer Vision Cooperation grant from Huawei, the General Research Fund sponsored by the Research Grants Council of Hong Kong (Project Nos. CUHK 416510, 416312, and 14207814), National Natural Science Foundation of China (91320101), Guangdong Innovative Research Team Program (No. 201001D0104648280), and Shenzhen Basic Research Program (JCYJ20130402113127496).

References

[1] T. Ahonen, A. Hadid, and M. Pietikainen, “Face description with local binary patterns: Application to face recognition,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 12, pp. 2037–2041, Dec. 2006.
[2] O. Barkan, J. Weill, L. Wolf, and H. Aronowitz, “Fast high dimensional vector multiplication face recognition,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2013, pp. 1960–1967.
[3] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman, “Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 7, pp. 711–720, Jul. 1997.
[4] T. Berg and P. Belhumeur, “Tom-vs-Pete classifiers and identity-preserving alignment for face verification,” in *Proc. Brit. Mach. Vis. Conf.*, 2012, pp. 129.1–129.11.
[5] T. Berg and P. Belhumeur, “Poof: Part-based one-vs-one features for fine-grained categorization, face verification, and attribute estimation,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2013, pp. 955–962.
[6] A. Bissacco, M. Cummins, Y. Netzer, and H. Neven, “Photoocr: Reading text in uncontrolled conditions,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2013, pp. 785–792.
[7] X. Cai, C. Wang, B. Xiao, X. Chen, and J. Zhou, “Deep nonlinear metric learning with independent subspace analysis for face verification,” in *Proc. ACM Multimedia*, 2012, pp. 749–752.
[8] Q. Cao, Y. Ying, and P. Li, “Similarity metric learning for face recognition,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2013, pp. 2408–2415.
[9] X. Cao, D. Wipf, F. Wen, G. Duan, and J. Sun, “A practical transfer learning algorithm for face verification,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2013, pp. 3208–3215.
[10] Z. Cao, Q. Yin, X. Tang, and J. Sun, “Face recognition with learning-based descriptor,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2010, pp. 2707–2714.
[11] C.-C. Chang and C.-J. Lin, “LIBSVM: A library for support vector machines,” *ACM Trans. Intell. Syst. Technol.*, vol. 2, pp. 27:1–27:27, 2011.
[12] D. Chen, X. Cao, L. Wang, F. Wen, and J. Sun, “Bayesian face revisited: A joint formulation,” in *Proc. Eur. Conf. Comput. Vis.*, 2012.
[13] D. Chen, X. Cao, F. Wen, and J. Sun, “Blessing of dimensionality: High-dimensional feature and its efficient compression for face verification,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2013, pp. 3025–3032.

[14] S. Chopra, R. Hadsell, and Y. LeCun, “Learning a similarity metric discriminatively, with application to face verification,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2005, pp. 539–546.

[15] D. Ciresan, U. Meier, and J. Schmidhuber, “Multi-column deep neural networks for image classification,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2012, pp. 3642–3649.

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

[17] O. Déniz, G. Bueno, J. Salido, and F. De la Torre, “Face recognition using histograms of oriented gradients,” *Pattern Recog. Lett.*, vol. 32, no. 12, pp. 1598–1603, Sep. 2011.

[18] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, “LIBLINEAR: A library for large linear classification,” *J. Mach. Learn. Res.*, vol. 9, pp. 1871–1874, 2008.

[19] C. Farabet, C. Couprie, L. Najman, and Y. LeCun, “Learning hierarchical features for scene labeling,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1915–1929, Aug. 2013.

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

[21] G. Hinton and S. Osindero, “A fast learning algorithm for deep belief nets,” *Neural Comput.*, vol. 18, pp. 1527–1554, 2006.

[22] G. E. Hinton and R. R. Salakhutdinov, “Reducing the dimensionality of data with neural networks,” *Science*, vol. 313, pp. 504–507, 2006.

[23] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Improving neural networks by preventing co-adaptation of feature detectors,” *CoRR*, abs/1207.0580, 2012.

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

[25] C. Huang, S. Zhu, and K. Yu, “Large scale strongly supervised ensemble metric learning, with applications to face verification and retrieval,” NEC Tech. Rep. TR115, 2011.

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

[27] 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,” Univ. Massachusetts, Amherst, Tech. Rep. 07-49, 2007.

[28] Z. Huang, X. Zhao, S. Shan, R. Wang, and X. Chen, “Coupling alignments with recognition for still-to-video face recognition,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2013, pp. 3296–3303.

[29] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun, “What is the best multi-stage architecture for object recognition?” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2009, pp. 2146–2153.

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

[31] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and F.-F. Li, “Large-scale video classification with convolutional neural networks,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2014, pp. 1725–1732.

[32] A. Krizhevsky, I. Sutskever, and G. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Proc. NIPS*, 2012, pp. 1097–1105.

[33] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar, “Attribute and simile classifiers for face verification,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2009, pp. 365–372.

[34] H. Larochelle, M. Mandel, R. Pascanu, and Y. Bengio, “Learning algorithms for the classification restricted Boltzmann machine,” *J. Mach. Learn. Res.*, vol 13, pp. 643–669, 2012.

[35] Q. Le, M. Ranzato, R. Monga, M. Devin, K. Chen, G. Corrado, J. Dean, and A. Ng, “Building high-level features using large scale unsupervised learning,” in *Proc. 29th Int. Conf. Mach. Learn.*, 2012, pp. 81–88.

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

[37] H. Li, G. Hua, Z. Lin, J. Brandt, and J. Yang, “Probabilistic elastic matching for pose variant face verification,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2013, pp. 3499–3506.

[38] P. Li, S. Prince, Y. Fu, U. Mohammed, and J. Elder, “Probabilistic models for inference about identity,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 1, pp. 144–157, Jan. 2012.

[39] P. Luo, X. Wang, and X. Tang, “Hierarchical face parsing via deep learning,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2012, pp. 2480–2487.

[40] P. Luo, X. Wang, and X. Tang, “A deep sum-product architecture for robust facial attributes analysis,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2013, pp. 2864–2871.

[41] V. Nair and G. E. Hinton, “Rectified linear units improve restricted Boltzmann machines,” in *Proc. Int. Conf. Mach. Learn.*, 2010, pp. 807–814.

[42] H. V. Nguyen and L. Bai, “Cosine similarity metric learning for face verification,” in *Proc. 10th Asian Conf. Comput. Vis.*, 2010, pp. 709–720.

[43] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, “Learning and transferring mid-level image representations using convolutional neural networks,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2014, pp. 1717–1724.

[44] N. Pinto and D. D. Cox, “Beyond simple features: A large-scale feature search approach to unconstrained face recognition,” in *Proc. IEEE Int. Conf. Autom. Face Gesture Recog. Workshops*, 2011, pp. 8–15.

[45] H. Poon and P. Domingos, “Sum-product networks: A new deep architecture,” in *Proc. UAI*, 2011, pp. 337–346.

[46] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, “OverFeat: Integrated recognition, localization and detection using convolutional networks,” in *Proc. Int. Conf. Learn. Representations*, 2014.

[47] K. Simonyan, O. M. Parkhi, A. Vedaldi, and A. Zisserman, “Fisher vector faces in the wild,” in *Proc. Brit. Mach. Vis. Conf.*, 2013, pp. 8.1–8.12.

[48] Y. Sun, X. Wang, and X. Tang, “Deep convolutional network cascade for facial point detection,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2013, pp. 3476–3483.

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

[50] Y. Sun, X. Wang, and X. Tang, “Deep learning face representation from predicting 10,000 classes,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2014, pp. 1891–1898.

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

[52] A. Wagner, J. Wright, A. Ganesh, Z. Zhou, H. Mobahi, and Y. Ma, “Toward a practical face recognition system: Robust alignment and illumination by sparse representation,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, vol. 2, pp. 372–386, Feb. 2012.

[53] X. Wang and X. Tang, “A unified framework for subspace face recognition,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 9, pp. 1222–1228, Sep. 2004.

[54] L. Wiskott, J.-M. Fellous, N. Krger, and C. V. D. Malsburg, “Face recognition by elastic bunch graph matching,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 7, pp. 775–779, Jul. 1997.

[55] L. Wolf, T. Hassner, and Y. Taigman, “Descriptor based methods in the wild,” in *Proc. Workshop Faces Real-Life Images ECCV*, 2008.

[56] L. Wolf, T. Hassner, and Y. Taigman, “Effective unconstrained face recognition by combining multiple descriptors and learned background statistics,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 10, pp. 1978–1990, Oct. 2011.

[57] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, “Robust face recognition via sparse representation,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 2, pp. 210–227, Feb. 2009.

[58] M. Yang and L. Zhang, “Gabor feature based sparse representation for face recognition with Gabor occlusion dictionary,” in *Proc. Eur. Conf. Comput. Vis.*, 2010, pp. 448–461.

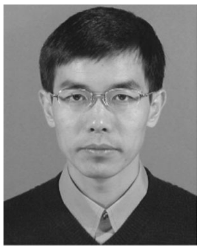
[59] Q. Yin, X. Tang, and J. Sun, “An associate-predict model for face recognition,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2011, pp. 497–504.

[60] L. Zhang, M. Yang, and X. Feng, “Sparse representation or collaborative representation: Which helps face recognition?” in *Proc. Int. Conf. Comput. Vis.*, 2011, pp. 471–478.

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

Yi Sun received the BEng degree from Tsinghua University in electronic engineering in 2011. Currently he is working toward the PhD degree in the Information Engineering Department at the Chinese University of Hong Kong. His research interests include computer vision and deep learning, especially in the areas of face recognition and face alignment.

Xiaogang Wang (S'03-M'10) received the BS degree from the special class for gifted young at the University of Science and Technology of China in electrical engineering and information science in 2001, and the MPhil degree from the Chinese University of Hong Kong in 2004. He received the PhD degree in computer science from the Massachusetts Institute of Technology. He is currently an assistant professor in the Department of Electronic Engineering at the Chinese University of Hong Kong. He received the Outstanding Young Researcher in Automatic Human Behaviour Analysis Award in 2011, Hong Kong RGC Early Career Award in 2012, and Young Researcher Award of the Chinese University of Hong Kong. He was the area chair of IEEE International Conference on Computer Vision (ICCV) 2011, European Conference on Computer Vision (ECCV) 2014, and Asian Conference on Computer Vision (ACCV) 2014. He is the associate editor of the *Image and Visual Computing Journal*, and *IEEE Transactions on Circuits and Systems for Video Technology*. His research interests include computer vision and machine learning. He is a member of the IEEE.



Xiaoou Tang (S'93-M'96-SM'02-F'09) received the BS degree from the University of Science and Technology of China, Hefei, in 1990, and the MS degree from the University of Rochester, Rochester, NY, in 1991. He received the PhD degree from the Massachusetts Institute of Technology, Cambridge, in 1996. He is a professor in the Department of Information Engineering and an associate dean (research) of the Faculty of Engineering of the Chinese University of Hong Kong. He worked as the group manager of the Visual Computing Group at the Microsoft Research Asia from 2005 to 2008. His research interests include computer vision, pattern recognition, and video processing. Dr. Tang received the Best Paper Award at the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2009. He is a program chair of the IEEE International Conference on Computer Vision (ICCV) 2009 and an associate editor of the *IEEE Transactions on Pattern Analysis and Machine Intelligence* (PAMI) and *International Journal of Computer Vision* (IJCV). He is a fellow of the IEEE.