Stacked PCA Network (SPCANet): An Effective Deep Learning for Face Recognition

Lei Tian[1], Chunxiao Fan[1], Yue Ming[1], Yi Jin[2]
[1]Beijing Key Laboratory of Work Safety Intelligent Monitoring,
School of Electronic Engineering, Beijing University of Posts and Telecommunications,
Beijing, 100876, P.R.China
[2]School of Computer & Information Technology,
Beijing Jiaotong University,
Beijing 100044, P.R.China
Email: tianlei189@sina.com, fcxg100@163.com, myname35875235@126.com, yjin@bjtu.edu.cn

***Abstract*—High-level features can represent the semantics of the original data and it is a plausible way to avoid the problem of hand-crafted features for face recognition. This paper proposes an effective deep learning framework by stacking multiple output features that learned through each stage of the Convolutional Neural Network (CNN). Different from the traditional deep learning network, we use Principal Component Analysis (PCA) to get the filter kernels of convolutional layer, which is name as Stacked PCA Network (SPCANet). Our SPCANet model follows the basic architecture of the CNN, which comprises three layers in each stage: convolutional filter layer, nonlinear processing layer and feature pooling layer. Firstly, in the convolutional filter layer of our model, PCA instead of stochastic gradient descent (SGD) is employed to learn filter kernels, and the output of all cascaded convolutional filter layers is used as the input of nonlinear processing layer. Secondly, the following nonlinear processing layer is also simplified. We use hashing method for nonlinear processing. Thirdly, the block based histograms instead of max-pooling technique are employed in the feature pooling layer. In the last output layer, the output of each stage is stacked together as one final feature output of our model. Extensive experiments conducted on many different face recognition scenarios demonstrate the effectiveness of our proposed approach.**

***Index Terms*—Convolutional Neural Network, Stacked PCA Network, Face Recognition, Malicious Occlusion.**

I. Introduction

During the past decades, face recognition has been successfully applied in many areas, such as access control, ID authentication, medical nursing and watch-list surveillance. In recent years, numerous efforts are made to manually designed low-level features for 2D face recognition [1]–[5] and 3D face recognition [6]–[9]. Because of their effectiveness and efficient computation, the low level features achieve great success for face recognition tasks. However, to achieve the performance as accurate as human on those uncontrolled scenarios, it is still a far way to go [10] [11]. Their successes mainly depend on the effectiveness of manually designed features for specific data and tasks. Designing effective features for new tasks usually requires new domain knowledge, therefore, most hand-crafted features can't simply be adopted to new conditions [12].

With the development of Deep Learning (DL), it is possible for the face recognition algorithm to realize the robustness and accuracy simultaneously. Deep Neural Network (DNN) learns feature from data instead of manually designed feature and it is considered as an alternative approach to remedy the problems of hand-crafted features. The representative example is convolutional neural network (CNN) [13]–[21] . The CNN consists of multiple trainable stages which generally comprise three layers: a convolutional layer, a nonlinear layer and pooling layer. In general, the filter kernels are learned by stochastic gradient descent (SGD) in each stage of CNN [13]–[19]. The Restricted Boltzmann Machine (RBM) [20] is also used as the kernels of the filter learning. In order to avoid parameter tuning, [21] proposed an approach that learns filter kernels by PCA or LDA.

However, the limitations of the conventional CNN are twofold. **On the one hand**, its success on visual classification partly depends on parameter tuning which is a time-consuming process, even need some *ad hoc* tricks. **On the other hand**, CNN achieves excellent performance only when their architecture is deep enough, or their performances are not as good as the hand-crafted feature. To overcome the problems mentioned above, our proposed SPCANet model whose the number of deep network layers is fewer than traditional DL framework, without tuning parameter. For each stage, we extract the leading eigenvectors from facial image's patches, which are used as filter kernels instead of randomly initializing kernels and update them by SGD. In order to extract more discriminative features, there is no nonlinear processing layer until the end of convolutional filter layer. It means that the feature maps are extracted from all cascaded convolutional layers and they are used as the input of nonlinear processing layer. The binary hashing, instead of the *Sigmoid* or *ReLU* function [13], is employed to nonlinearly process feature maps. Then, the block histograms method is chosen to feature pooling. Since the input of pooling layer is a hashed decimal-valued image, the conventional pooling method are unfit for our model, such as Max-Pooling and Average-Pooling. At last, we stack the output of each stage as the final feature of our model. Figure 1 illustrates how our SPCANet model extracts multiple features from face images.

The rest of this paper is organized as follows: In Section 2, we detail the architecture of our proposed SPCANet model.

978-1-4799-8058-1/15/$31.00 ©2015 IEEE.

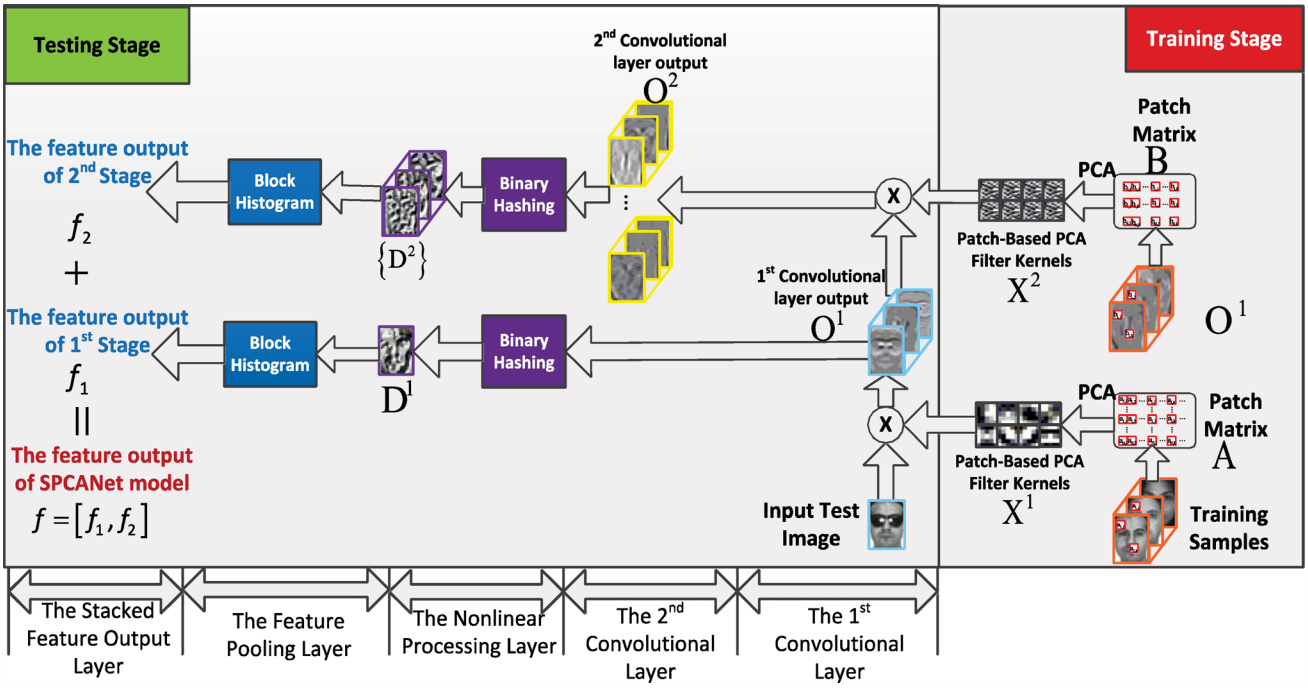

Fig. 1. Illustration of how the proposed SPCANet train filter kernels and extracts feature.

Then we present the experimental results of our algorithm in Section 3. Finally, we conclude our proposed work in Section 4.

II. Stacked PCA Network (SPCANet)

In this section, we first introduce the convolutional layer of our SPCANet model, then describe the nonlinear processing layer and feature pooling layer in detail, respectively.

A. The Convolutional Filter Layer

In order to take full advantage of convolution, we reserve the basic convolutional processing of CNN and just replace SGD with PCA to learn the filter kernels. We extract the leading eigenvectors from the image's patches whose sizes are equal to the size of CNN's kernels. We assume that N face images are used as training samples of size $m_1 \times m_2$. The patch size is $p_1 \times p_2$ and the step of overlapping patch is $s_1 = s_2 = 1$. We choose K_i (the number of filter kernels) leading eigenvectors in the *ith* stage. For each training sample, we collect all overlapping patches step by step and vectorize them. Then, we subtract patch mean from each patch and get mean-removed matrix $A_i = [a_{i,1}, a_{i,2}, \cdots, a_{i,m_1 m_2}]$ from the *ith* image. By repeating the same process for all training samples, we get

$$A = [A_1, A_2, \cdots, A_N] \in \mathbb{R}^{p_1 p_2 \times m_1 m_2 N} \tag{1}$$

Then, we get the transformation vectors V^i from data matrix A. The PCA aims to find a set of orthogonal basis functions that has maximum variance from the data. Its objective function is that:

$$V_{opt} = \arg\max_{V} V^T C V \tag{2}$$

where C is the data covariance matrix. In order to clearly denote, $[F(a)]_{k=1}^{K_1}$ means data a through PCA algorithm and get the leading K_1 eigenvectors. Then we map each eigenvector to matrix of size $p_1 \times p_2$. So the filter kernels of our SPCANet model in first stage is can be expressed as

$$X^1 = vec2mat\left([F(A)]_{k=1}^{K_1}\right) \in \mathbb{R}^{p_1 \times p_2 \times K_1} \tag{3}$$

For the *ith* image, we convolute it with PCA-learned kernels and get

$$O_i^1 = \left\{I_i * X_{k_1}^1\right\}_{k_1=1}^{K_1} \in \mathbb{R}^{m_1 \times m_2 \times K_1} \tag{4}$$

where * denotes convolution, and the boundary of I_i is zero-padded before convolution, so that O_i^1 has the same size of I_i. The filter output of the first convolutional layer are K_1 convolutional images.

We only insert nonlinear processing layer when all convolutional layers have been processed, which is different from the architecture of conventional CNN. Since the operation that we insert nonlinear processing layer after each convolutional layer shows no chance of improving the performance of our SPCANet model through extensive experiments. We simplified our network's architecture by considering the efficiency of training. The output of the first convolutional layer is used as the input of second convolutional layer instead of nonlinear processing layer. Almost repeating the same process as the first stage, we collect all patches of O_i^1 and obtain $B = [B_1, B_2, \cdots, B_N] \in \mathbb{R}^{p_1 p_2 \times m_1 m_2 K_1 N}$. Then we compute filter kernels of the second stage X^2 from matrix B by PCA. For each input $O_i^1(k_1)$ of the second stage, we obtain K_2 outputs $O_i^2(k_1)$:

$$O_i^2(k_1) = \left\{O_i^1(k_1) * X_{k_2}^2\right\}_{k_2=1}^{K_2} \tag{5}$$

where $k_1 = 1, \cdots, K_1$ and $i = 1, 2, \cdots, N$.

B. The Nonlinear Processing Layer

We choose both O_i^1 and O_i^2 as the input of next nonlinear processing layer. The method of learning filter in our model is different from conventional CNN, it is obvious that the *Sigmoid* and *ReLU* function [13] aren't suitable for our SPCANet

model, although both of them are efficient nonlinear functions on CNN. For a input image, we get K_1 outputs O_i^1 from the first stage, while the Heaviside step function is employed to binarize these outputs. Then we sum K_1 binary outputs with their bits weighted, and obtain one decimal-valued image D_i^1:

$$D_i^1 = \sum_{k_1=1}^{K_1} 2^{k_1-1}\varphi\left(O_i^1\right) \tag{6}$$

Similar to the output of the first stage, we get K_1K_2 outputs O_i^2 from the second stage. We binarize all images and sum each K_2 binary outputs with their weights, then we obtain K_1 decimal-valued images $\left\{D_i^2\right\}$:

$$D_i^2(k_1) = \sum_{k_2=1}^{K_2} 2^{k_2-1}\varphi\left(O_i^2(k_1)\right), k_1 = 1, \cdots, K_1 \tag{7}$$

C. The Feature Pooling Layer

To our best knowledge, the max-pooling and average-pooling method are not suitable for processing the output of our model's nonlinear layer. So the statistics of block histogram are used as the output features of pooling layer. For each D_i^1, we partition it into J blocks (each block size is $h_1 \times h_2$) and compute the histogram of pixel value in each block, then we concatenate all J histograms into one vector and donate it as $f_1 = BlkHist\left(D_i^1\right)$, which is used as the final output of the first stage. Almost repeating the same process as mentioned above, for each $D_i^2(k_1)$, we also concatenate all the J histograms into one vector, donated as $BlkHist\left(D_i^2(k_1)\right)$. And put result vector together for each k_1, $k_1 = 1, \cdots, K_1$. We further define $f_2 = \left[BlkHist\left(D_i^2(1)\right), \cdots, BlkHist\left(D_i^2(K_1)\right)\right]$ as the final output of the second stage. At last, we stack the output of first stage and second stage, therefore the final feature of input image I_i is expressed as

$$f = [f_1, f_2] \in \mathbb{R}^{\left(2^{K_1}J + 2^{K_2}K_1J\right)} \tag{8}$$

III. Experiment Results

In this section, we conduct experiments on public databases for face recognition. We first investigate the impact of filter kernels that learned from different databases and the impact of the block size (for histogram computation) in the feature pooling layer on PIE database. We then examine the performance of our SPCANet algorithm when faced with various intra-class variabilities on AR database. Finally, we demonstrate the robustness of our algorithm when faced with artificial block occlusion on Yale-B database.

A. The Impact factors of SPCANet

We investigate the impact of filter kernels learned from different databases and the block size of histogram on PIE, respectively. The PIE face database, which consists of a total of 41368 face images for 68 individuals. The five near frontal poses(C05, C07, C09, C27, C29) are used in this test. Original images were normalized and cropped into 32×32 pixels. The chosen images are varied of different illumination, poses and expression, there are around 170 frontal face images for each individual. And we randomly choose 5 or 10 images per individual as training set, the rest of the database is used as testing set. For each number of training samples, we average the results over 10 random splits.

TABLE I
FACE RECOGNITION RATE (%) OF OUR PROPOSED SPCANET ON PIE FOR FILTER KERNELS LEARNED FROM DIFFERENT DATASET

	5 Train Samples	10 Train Samples
SPCANet (*V from PIE-itself*)	81.27	93.68
SPCANet (*V from MultiPIE*)	**86.17**	**96.08**

1) Filter Kernels Learned From Different Databases: We study how filter kernels in our SPCANet model that learned from different data sources impact the accuracy of face recognition. The preprocessed and down-sampled images in MultiPIE dataset [22] are used to learn the filter kernels of SPCANet. The training images include 129 subjects under all illuminations and all expressions at pose $-30°$ to $+30°$ and step is $15°$. And we also learn filter kernels from testing set itself. To compare fairly, we both set the patch size is $p_1 = p_2 = 5$, the number of filters is $K_1 = K_2 = 8$ in the convolutional filter layer, and the size of non-overlapping block in pooling layer is $h_1 = h_2 = 7$. The SVM classifier is used to classify output feature.

The results are shown in Table I. One can see that filter kernels learned from MultiPIE outperforms those learned from itself. As far as we know, there are more intra-class variabilities in MultiPIE. The results suggest that the more intra-class variabilities of training samples, the more discriminative features learned from SPCANet framework.

2) Impact of the block size of histogram computation: We next examine the impact of the block size on histogram computation by using PIE database. The filter kernels learned from MultiPIE are applied to our SPCANet framework. The kernel size is $p_1 = p_2 = 5$ and the number of filters is $K_1 = K_2 = 8$. We vary the $[h_1, h_2]$ from 3 to 15 for histogram computation in the feature pooling layer. And the step of block is equal to block size, which means that the blocks are non-overlapping. The results are shown in Fig 2.

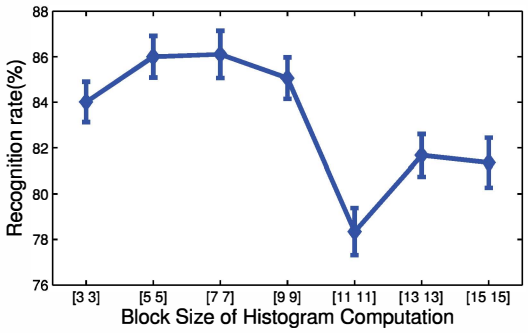

Fig. 2. Recognition rate of SPCANet on PIE for different histogram-block size

We can see that block size $h_1 = h_2 = 7$ achieves the best performance and the size of input image is 32×32. So a prominent message is drawn from extensive experiments, not only the mentioned above experiments. It is that our SPCANet

TABLE II
FACE RECOGNITION RATE (%) FACED WITH DIFFERENT TEST SET ON AR DATABASE

	Illumination	Expression	Disguise	Dis+Illum
PCA+SRC	86.67	78.50	56.50	30.37
LBP	93.83	81.33	91.25	79.63
P-LBP	97.50	80.33	91.25	88.58
PCA-Network	99.50	95.83	99.50	**99.12**
SPCANet	**100.00**	**96.33**	**99.75**	**99.12**

model achieves best performance when the block size (for histogram computation) is approximately equal to quarter of image's size.

B. Face Recognition on AR Dataset

We further evaluate the performance of our proposed SPCANet to cope with the intra-class variabilities in reality using MultiPIE-based filter kernels on AR Dataset [23]. The intra-class variability includes illumination, expression and malicious occlusions. The AR database consists of over 4,000 face images for 126 subjects. For each individual, pictures are taken under 4 different expressions, 3 illumination conditions, 2 disguise cases, 4 illumination and disguise cases during two separate sessions. We choose a subset of the database consisting of 50 male and 50 female subjects. The facial images are cropped into 165×120 and converted to gray scale. The images of frontal illumination and neural are used as the gallery images, and the rest all forms the testing set. We also use filter kernels learned from MultiPIE, the non-overlapping block size is $[h_1, h_2] = [8, 6]$. The NN classifier with the chi-square distance measure is applied for face recognition. We compare the performance of our SPCANet with PCA-Network [21], LBP [24], P-LBP [25] and sparse representation classification (SRC) with PCA [26]. The results are given in Table II.

One can see that the recognition rate of our SPCANet is almost perfect when it faced with various condition test sets, and achieve 100% for illumination variations test set. We can draw a conclusion that our proposed SPCANet is insensitive to changes of illumination, expression and malicious occlusion.

Fig. 3. The illustration of various level of occluded face image

C. Face Recognition on Extended Yale B Database

To challenge ourselves, we simulate various levels of contiguous occlusion in this experiment, using the Extended Yale B Database [27]. The Extended Yale B Database consists of 2414 frontal images for 38 individuals. The facial images are captured under various illumination conditions. We normalize and cropped the origin images into 96×84, and choose Subset 1 and 2 (from normal to moderate lighting conditions) for training and Subset 3 (more extreme lighting conditions) for testing. It would be a relatively easy face recognition task without artificial block occlusion. We simulate various levels

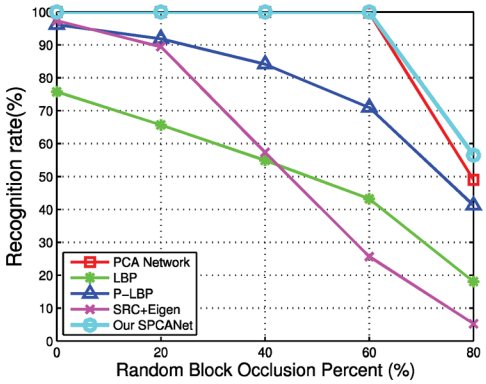

Fig. 4. Recognition rate (%) faced with block occlusion on Yale-B database

(0%-80%) of block occlusion by replacing a random region of testing images with an unrelated image. Figure 3 is a example. The patch size is $p_1 = p_2 = 5$, the number of filters is $K_1 = K_2 = 8$ and we learn filter kernels from database itself. In the pooling layer, the non-overlapping block size is $h_1 = h_2 = 7$. The NN with chi-square distance is applied.

The results are given in Fig 4. We see that our algorithm again achieves almost perfect accuracy, even sustaining more than half accuracy when 80% pixels are occluded. The SPCANet filter kernels learned from database itself provide sufficient robustness to artificial block occlusion. The reason is that the responses of unrelated patches were weakened through cascaded PCA filter kernels, there weren't passed onto nonlinear processing layer and feature pooling layer. Another issue worth mentioning is that DL-based method (ours and PCA-Network) dramatically outperform the hand-crafted feature, which validates the hypothesis mentioned above that our SPCANet model could be the solution to remedy the limitation of hand-crafted features.

IV. Conclusion

In this paper we propose a simplified version of CNN, termed as Stacked PCA Neural Network (SPCANet). The network follows the basic architecture of CNN. Learning filter kernels go through PCA instead of SGD, then performing the nonlinear computation of the output of convolutional layer by hashing method and pooling the decimal-valued image using block-wise histogram technique in each stage. At last, we stack the output of multiple stages as final feature. There is no nonlinear processing layer until images are processed through all convolutional filter layer which is a main characteristic of our model that is different from others. Extensive experiments demonstrate our model achieves almost perfect performance when faced with various intra-class variability. So SPCANet can effectively improve the performance and robustness of face recognition.

Acknowledgment

The work presented in this paper was supported by the National Natural Science Foundation of China (Grants No. NSFC-61402046,61403024), President Funding of Beijing University of Posts and Telecommunications (Grants No. 2013XZ10).

References

[1] Wei Ge, Zhiwen Xu, Chunlei Shi, and Weida Zhan, "Recognition of expression-variant faces using sift method," in *Computational Intelligence and Communication Networks (CICN), 2012 Fourth International Conference on*, Nov 2012, pp. 398–401.

[2] A. Ramirez Rivera, Rojas Castillo, and Oksam Chae, "Local directional number pattern for face analysis: Face and expression recognition," *Image Processing, IEEE Transactions on*, vol. 22, no. 5, pp. 1740–1752, May 2013.

[3] F. Juefei-Xu and M. Savvides, "Subspace-based discrete transform encoded local binary patterns representations for robust periocular matching on nist's face recognition grand challenge," *Image Processing, IEEE Transactions on*, vol. 23, no. 8, pp. 3490–3505, Aug 2014.

[4] Xiaoli Li, Qiuqi Ruan, Gaoyun An, and Yi Jin, "Automatic 3d facial expression recognition based on polytypic local binary pattern," in *Signal Processing (ICSP), 2014 12th International Conference on*, Oct 2014, pp. 1030–1035.

[5] Di Huang, M. Ardabilian, Yunhong Wang, and Liming Chen, "3-d face recognition using elbp-based facial description and local feature hybrid matching," *Information Forensics and Security, IEEE Transactions on*, vol. 7, no. 5, pp. 1551–1565, Oct 2012.

[6] Yue Ming, "Robust regional bounding spherical descriptor for 3d face recognition and emotion analysis," *Image and Vision Computing*, vol. 35, no. 0, pp. 14 – 22, 2015.

[7] B. Chu, S. Romdhani, and Liming Chen, "3d-aided face recognition robust to expression and pose variations," in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, June 2014, pp. 1907–1914.

[8] Yue Ming, "Rigid-area orthogonal spectral regression for efficient 3d face recognition," *Neurocomputing*, vol. 129, no. 0, pp. 445 – 457, 2014.

[9] Yue Ming and Qiuqi Ruan, "Robust sparse bounding sphere for 3d face recognition," *Image and Vision Computing*, vol. 30, no. 8, pp. 524 – 534, 2012, Special Section: Opinion Papers.

[10] Meina Kan, Shiguang Shan, Hong Chang, and Xilin Chen, "Stacked progressive auto-encoders (spae) for face recognition across poses," in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, June 2014, pp. 1883–1890.

[11] G. Chiachia, A.X. Falcao, N. Pinto, A. Rocha, and D. Cox, "Learning person-specific representations from faces in the wild," *Information Forensics and Security, IEEE Transactions on*, vol. 9, no. 12, pp. 2089–2099, Dec 2014.

[12] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 35, no. 8, pp. 1798–1828, Aug 2013.

[13] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[14] Koray Kavukcuoglu, Pierre Sermanet, Y-Lan Boureau, Karol Gregor, Michaël Mathieu, and Yann L Cun, "Learning convolutional feature hierarchies for visual recognition," in *Advances in neural information processing systems*, 2010, pp. 1090–1098.

[15] Y. Taigman, Ming Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, June 2014, pp. 1701–1708.

[16] Yi Sun, Xiaogang Wang, and Xiaoou Tang, "Deep learning face representation from predicting 10,000 classes," in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, June 2014, pp. 1891–1898.

[17] G.B. Huang, Honglak Lee, and E. Learned-Miller, "Learning hierarchical representations for face verification with convolutional deep belief networks," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, June 2012, pp. 2518–2525.

[18] Yi Sun, Yuheng Chen, Xiaogang Wang, and Xiaoou Tang, "Deep learning face representation by joint identification-verification," in *Advances in Neural Information Processing Systems*, 2014, pp. 1988–1996.

[19] Yi Sun, Xiaogang Wang, and Xiaoou Tang, "Hybrid deep learning for face verification," in *Computer Vision (ICCV), 2013 IEEE International Conference on*, Dec 2013, pp. 1489–1496.

[20] Vinod Nair and Geoffrey E Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 2010, pp. 807–814.

[21] Tsung-Han Chan, Kui Jia, Shenghua Gao, Jiwen Lu, Zinan Zeng, and Yi Ma, "Pcanet: A simple deep learning baseline for image classification?," *arXiv preprint arXiv:1404.3606*, 2014.

[22] R. Gross, I. Matthews, J. Cohn, T. Kanade, and S. Baker, "Multi-pie," in *Automatic Face Gesture Recognition, 2008. FG '08. 8th IEEE International Conference on*, Sept 2008, pp. 1–8.

[23] Aleix Martínez and Robert Benavente, "The ar face database," Jun 1998.

[24] T. Ahonen, A. Hadid, and M. Pietikainen, "Face description with local binary patterns: Application to face recognition," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 28, no. 12, pp. 2037–2041, Dec 2006.

[25] Xiaoyang Tan and B. Triggs, "Enhanced local texture feature sets for face recognition under difficult lighting conditions," *Image Processing, IEEE Transactions on*, vol. 19, no. 6, pp. 1635–1650, June 2010.

[26] J. Wright, A.Y. Yang, A. Ganesh, S.S. Sastry, and Yi Ma, "Robust face recognition via sparse representation," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 31, no. 2, pp. 210–227, Feb 2009.

[27] A.S. Georghiades, P.N. Belhumeur, and D. Kriegman, "From few to many: illumination cone models for face recognition under variable lighting and pose," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 23, no. 6, pp. 643–660, Jun 2001.