2020 IEEE 5th International Conference on Cloud Computing and Big Data Analytics

A Compressed Deep Convolutional Neural Networks for Face Recognition

Li Yao
Key Laboratory of Public Security Information Application Based on Big-data Architecture Ministry of Public Security, Zhejiang Police College
Hangzhou, China
e-mail: yaoli@zjjcxy.cn

***Abstract*—Recent years, deep convolutional neural network has led to significant improvements in face recognition and becomes one of the most popular techniques in computer vision community. However, deep CNN model requires vast amounts of data and time for training and deploying. To solve this problem, we present a deep compact convolutional neural network for face representation. First we apply PCA for initializing convolution filter. Then we adopt DCT and binary hashing for extracting face features. The models are trained on the CASIA-webFace datasets under Caffe framework. Experimental results show that the proposed methods achieve competitive accuracy on the LFW verification benchmark.**

Keywords-convolution neural network; deep learning; face recognition

I. INTRODUCTION

Machine learning is a type of artificial intelligence that provides computers with the ability to learn without being explicitly programmed. However, conventional machine-learning techniques still find it difficult to learn meaningful information about new applications for existing knowledge representation models [1]. During the last few years, deep learning [2], which is a part of a broader family of machine learning methods, have been turned out to be effective against various domains in computer vision community. Especially, it benefits a lot from dealing with large scale training data sets.

The core of deep learning is feature learning. It aims at acquiring feature information on hierarchical network, so as to solve the important problems which need artificial design before. Deep learning is a framework that contains several important algorithms, such as Convolutional Neural Networks (CNN) [3], Auto Encoder [4], Sparse Coding [5], Restricted Boltzmann Machine (RBM) [6], Deep Belief Networks (DBN) [7], Recurrent neural Network (RNN) [8]. Recently, CNN has achieved significant improvements and becomes one of the most popular and powerful tools in diverse image recognition tasks including image classification [9], domain adaptation, scene recognition, image retrieval [10], and so on. The main advantage of CNN is an ability to extract complicated hidden features from high dimensional data with complex structure. Inspired by the biological neural network, CNN shared weights network to reduce both the complexity of models and the numbers of weights. Moreover, CNN can automatically learn features jointly with the classification, while most existing counter-anti-forensic techniques have to separate the feature extraction and classification.

For face recognition tasks, CNN model is also widely used. In order to gain high accuracy, face recognition algorithms improve the fitting capabilities usually by increasing the numbers of both model layers and model parameters [11] [12]. Though these neural networks are very powerful, the large number of weights consumes considerable storage and memory bandwidth. There are many real-time scenarios, which require the huge amount of face data to be processed in a short time. Even though high speed computing equipment such as graphics processing unit (GPU) is used, the speed of face recognition system still needs to be improved [13]. Another considerable problems with these complex CNN models is that the adjustment of network weights requires a lot of label data. When the amount of data is small, the model cannot be fully trained. So it is only suitable for large datasets with much tag data.

In this paper, we propose a new compression method for convolutional neural networks. We consider to develop a practical face recognition system that can give high accuracy with low computational expense. The main feature of the proposed method can be summarized as follows: (1) we use principal component analysis (PCA) for initializing convolutional filter of CNN. (2) We covert convolutional filter weight into frequency domain by discrete cosine transform (DCT) for compression.

The remainder of this paper is organized as follows: Section 2 reviews previous studies on face recognition and convolutional neural networks. We illustrate proposed compressed CNN architecture in details in Section 3. Section 4 describes the steps for training and evaluation on different datasets. Our accuracy and performance are evaluated and compared to other face recognition techniques. Finally, Section 5 concludes this paper.

II. PREVIOUS WORK

This paper focuses on face recognition in images and videos, a problem that has received significant attention in the recent decades. Among the large number of methods proposed in the literature, we distinguish the ones that do not use deep learning, which we call “non-deep”, from ones that do, that we call “deep”. Non-deep methods can be categorized into two main categories: feature-based and holistic. Feature-based approaches first process the face image and extract distinctive facial features using

978-1-7281-6024-5/20/$31.00 ©2020 IEEE

handcrafted local image descriptors such as SIFT, SURF, LBP, HOG [14] [15] [16]. Then they employ standard statistical pattern recognition techniques to match or verify faces using these measurements.

Holistic approaches try to extract global features on the entire face images, rather than on local features of the face. These methods are extended to the mixed linear subspace and nonlinear subspace. Turk [17] first represents faces as a combination of eigenvectors which is called Eigenfaces. It is a face description and recognition technique derived from the principal component analysis (PCA). Belhumeur [18] uses Fisherfaces based on linear discriminant analysis (LDA). The experimental results show that Fisherfaces is outperformed than Eigenfaces. Traditional PCA methods only rely on second-order relationship of pixels of the face image, while the important information may be contained in the high-order relationship between pixels. It is natural to consider using these higher-order statistics will find better recognition results. In [19], Bartlett proposes a generalized PCA method, called independent component analysis (ICA), for facial feature extraction.

Neural Network is another approach used for face recognition. It is more reliable as compared to linear methods. Artificial Neural Network (ANN) is a very first method that is used with single layer adaptive network [20]. In [21] the authors use Self Organizing Maps (SOM) to identify and verify faces on the basis of ear and hand geometry. The results taken on different databases concludes its efficiency and reliance. Lin [22] proposes a Probabilistic Decision Based Neural Network (PDBNN) for face recognition. The basic idea is to use virtual samples for reinforcement and anti-reinforcement learning, so as to obtain ideal probability estimation models. Then it is used for accelerating the network.

Convolutional neural network is a special kind of neural network model which is developed and improved from the back propagation (BP) neural network. Similar as BP network, CNN uses hierarchical network structure. Neurons in adjacent layer are connected to each other. As suggested by Saxe et al. [23], the excellent performance of CNN can be largely attributed to these properties as certain structures with random weights could also achieve good results.

CNN has two main features, known as local connectivity and parameter sharing. Traditional BP neural network uses full connection between input layers and hidden layers. It is feasible to calculate features from relatively small images. But it consumes substantial time and storage while dealing with images of large scale via “full connection” network. A simple way to solve such problems in convolution layer is to restrict the connection between hidden units and input units. For example, each hidden unit connects only a small piece of adjacent area of the input image. Second, some neurons in the same layer connection weights can be shared. Parameter sharing can reduce the number of free variables, so as to extract features more effectively. By controlling the scale of the model, CNN achieves good generalization ability to visual problems.

In CNN, initialization is important for providing a reasonable starting weights value of the training network. Improper initialization may cause the network difficult to converge. Random initialization is the most used method for initialization of convolution kernel. Each parameter is randomly initialized on the basis of some probability distribution, such as standardized normal distribution. In 2010, Glorot proposed Xavier algorithm [24] to get random initial network weights following uniform distribution or Gauss distribution with mean zero. Experimental results show that the initialized weights can be trained to convergence in a short time, while avoiding the uncertainty of artificial interference over the initialization.

Although these CNN based methods have achieved ultimate accuracy on various face datasets, the computational cost is a tough issue due to deep architecture and multiple local patches. It is important to seek a CNN model with small size and low dimension for real-time applications.

In this paper, we propose a new convolutional neural networks structure based on CAFFE [25]. Compared to the traditional feature-based methods or matching-based methods, our method learns deep characteristics including edge, color, texture, and the overall feature in an implicit way. Furthermore, we perform a visualization of network layers to adjust our network parameters. Experiments and comparison with state-of-the-art methods are conducted on different scales of databases to demonstrate the effectiveness of our deep learning framework.

III. Approach

In this section, we describe our compressed convolutional neural network for face recognition in details.

A. Network Structure

Convolutional neural network is a multilayer neural network. Each layer is composed of two dimensional planes, and each plane is composed of multiple independent neurons. Fig. 1 shows a typical flowchart of the deep convolutional neural network. The input image is convoluted with three trainable filters and additive biases. After convolution, three feature maps are generated in C1 layer, then the four pixels in each set of feature maps are summed and weighted. Three feature maps of S2 layers are get by Sigmoid function with bias. These maps are filtered into C3 layer. This process will repeat until certain parameters configuration is satisfied. Finally, these pixels are rasterized and output.

In general, the C layer is feature-extraction layer, where each neuron in a local area is connected to the input layer to extract features. Once the local feature is extracted, its relationship to other characteristics of the position will be determined. The S layer is feature-mapping layer, where each layer is composed of multiple feature maps. The feature-mapping structure uses the sigmoid function to active convolution network, which makes the feature map displacement-invariant.

In addition, as the neurons on a mapping surface share weights, the number of free parameters of the network is reduced, and the complexity of network parameter selection is also reduced. Each feature extraction layer in convolution neural network is closely followed by an S layer. This unique

feature extraction structure makes the network have high distortion tolerance for input samples.

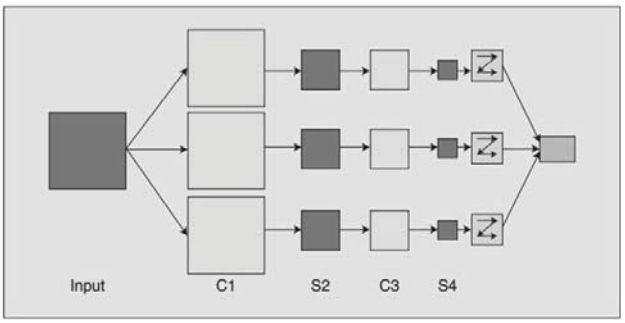

Figure 1. Conceptual description of Convolutional Neural Networks.

As described above, the most important forward propagation operations of convolutional neural networks are convolution and down sampling. The detailed process is shown as follows:

(1) Convolution layer.

In a convolution layer, the feature graph of the previous layer is convoluted by a learnable convolution kernel, which is combined with these convolution operations, and then an output feature graph is obtained by an excitation function.

The convolution layer in the convolution neural network consists of a number of convolution units, and the parameters of each convolution unit are optimized by the back propagation algorithm. The purpose of the convolution operation is to extract different features of input. The first layer of convolution layer may only extract some low-level features such as edges, lines and angles. And subsequent layers of the network can extract more complex features from low-level features.

At the end, activation function is applied for rectified linear units.

The process can be described as follows:

$$x_j^l = f\left(\sum_{i \in M_j} x_i^{l-1} * k_{ij}^l + b_j^l\right) \quad (1)$$

where x_j^l denotes the *j*-th feature map of the *l*-th layer, and *f*() denotes the activation function. M_j denotes the input images set. b_j^l is the bias term of the *j*-th feature map. k_{ij}^l denotes the convolution kernel connecting the *j*-th feature map of the *l*-th layer and the *i*-th feature map of the $(l-1)$th layer, and * denotes the convolution operation.

(2) Pooling layer.

Normally, high-dimensions features can be obtained after convolution stage. These features are split into several regions, and the maximum or average values are used for getting new features with low-dimensions. With regard to pooling layer, the number of input images is identical with output ones excepting with smaller size. The whole process can be described as follows:

$$x_j^l = f\left(\beta_j^l down(x_j^{l-1}) + b_j^l\right) \quad (2)$$

where *down*(•) denotes down-sampling function. β_j^l denotes the coefficient of the *j*-th feature map of the *l*-th layer and b_j^l is the bias term.

(3) Full-connected layer.

This layer combines all the local features into a global feature to calculate the score of the final class. The output of the convolution and pooling layers represents high-level features of the input image. The purpose of the fully connected layer is to use these features to classify the input image based on the training data set. In addition to classification, adding a fully connected layer is another simple way to learn non-linear combinations of these features.

Similar as traditional multilayer perceptron, it use softmax function or other SVM classifiers for activation in the output layer. The sum of the output probabilities of the fully connected layer is 1. This can be guaranteed at the output layer using softmax as the activation function. The softmax function takes a vector of values greater than 0 and converts them to a numeric vector between 0 and 1, with the sum being 1.

B. Initialize Convolution Filter by PCA

The essence of model training is to update neural network weights, which requires corresponding initial values for each parameter. There are two types of parameters to initialize in convolution layer of CNN, including the weight matrices and the bias vectors for each convolution filter. A well-chosen optimization starting point can speed up the convergence of gradient descent and help learning to converge to a good generalizing minimum.

Random initialization is the most common method for CNN and works well when the amount of training data is sufficiently large. Pre-training is an effective initialization method for early training neural networks. An easy-to-understand example is to use greedy layer wise auto-encoder to do unsupervised pre-training before fine-tuning. Evidence show that unsupervised pre-training guides the learning towards better generalization from the training data set [26]. The specific process can be briefly described as follows. In the first stage, each layer of the neural network is taken out to construct an auto-encoder for training. In the second stage fine-tuning, each layer of pre-training is put back into neural network, and the model is adjusted by using the initial parameters and training data obtained in pre-training stage. The parameters are further updated to form the final model.

In this paper, we use principal component analysis (PCA) for initiating convolution filter parameters. PCA is an unsupervised learning method for feature extraction and dimensionality reduction. It can convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables. By using unsupervised pre-training of PCA to initialize the convolution neural network, the hidden layer representation of the image to be identified is obtained by minimizing the reconstruction error. Then the filter set containing the statistical characteristics of the training data is learned.

We first use offline data set samples to calculate PCA basis vectors. Then use them as convolution kernels to compute features of the input pictures. The process of computing PCA basis vectors offline can be described as follows: for all input pictures, we first perform a fragmentation operation to find the small pieces at all pixel

positions. Then we calculate the PCA basis vector on these small piece sets, and the small piece size is exactly the convolution kernel size.

Assume we are given N input training images $\{I_i\}_{i=1}^{N}$ of size $m \times n$, and the convolution filter size is $k_1 \times k_2$. Slide the window over all the images to obtain a set of small slices, and each slice is expanded into a k-dimensional vector:

$$x_{i,1}, x_{i,2}, \cdots, x_{i,mn} \in \boldsymbol{\mathcal{R}}^{k_1 k_2} \tag{3}$$

where $x_{i,j}$ denotes the jth vectorized patch in I_i. Each patch subtracts the mean value to get a mean-removed matrix:

$$\boldsymbol{X}_i = [x_{i,1}, x_{i,2}, \cdots, x_{i,mn}] \tag{4}$$

Suppose that the number of ith filters in layer is L_i, we apply PCA for getting a family of orthonormal filters orthonormal filters, such as:

$$\min_{V \in \boldsymbol{\mathcal{R}}^{k_1 k_2 \times L_1}} \|\boldsymbol{X} - \boldsymbol{V}\boldsymbol{V}^T\boldsymbol{X}\|_F^2, s.t.\ \boldsymbol{V}\boldsymbol{V}^T = I_{L_1} \tag{5}$$

where I_{L_1} is identity matrix of size $L_1 \times L_1$. And L_1 is the principal eigenvectors of $\boldsymbol{X}\boldsymbol{X}^T$. The PCA filters can be described as follows:

$$\boldsymbol{W}_l^1 = \mathrm{mat}_{k_1,k_2}\left(q_l(\boldsymbol{X}X^T)\right) \in \boldsymbol{\mathcal{R}}^{k_1 k_2}, l = 1,2,\cdots,L_1 \tag{6}$$

where $\mathrm{mat}_{k_1,k_2}(v)$ is a function that maps v to a matrix $\boldsymbol{W} \in \boldsymbol{\mathcal{R}}^{k_1 k_2}$, and $q_l(\boldsymbol{X}\boldsymbol{X}^T)$ is the lth principal eigenvector of $\boldsymbol{X}\boldsymbol{X}^T$. This equation means to extract the eigenvectors corresponding to the first L_1 largest eigenvalues of the covariance matrix of $\boldsymbol{X}$ to form feature mapping matrix. The information of these zero-mean training samples is retained in these principal components.

C. DCT and Binarization Hash

To further compact the CNN framework, we adopt discrete cosine transform (DCT) for reconstructing the convolutional filter. As the pixels of feature maps after convolution are usually smooth. And filters in CNN tend to be smooth due to spatial locality. We transform these filters into the frequency domain with the discrete cosine transform (DCT) [27]. In frequency space, the filters are naturally dominated by low frequency components. This mathematical operation converts each frame of the video source from the spatial (2D) domain into the frequency domain. A perceptual model based loosely on the human psychovisual system discards high-frequency information.

DCTs are widely used in signal and image processing, especially for lossy compression where small high-frequency components can be discarded. As most of the signal information after DCT tends to be concentrated in a few low-frequency components, the small high-frequency components can be discarded to gain high compression ratio.

Let $\boldsymbol{X} \in \boldsymbol{\mathcal{R}}^{N \times N}$ denote the weight matrix for the convolutional filter. We perform one-dimensional DCT along the rows and then along the columns. The corresponding matrix in frequency domain after DCT is given by the formula as:

$$\boldsymbol{X}_{k_1,k_2} = \sum_{n_1=0}^{N-1} \sum_{n_2=0}^{N-1} \boldsymbol{x}_{n_1,n_2} \cos\left[\frac{\pi}{N}\left(n_1 + \frac{1}{2}\right)k_1\right] \cos\left[\frac{\pi}{N}\left(n_2 + \frac{1}{2}\right)k_2\right] \tag{7}$$

We simply denote the weight matrix in the frequency domain as:

$$\boldsymbol{V}^{kl} = f_{dct}(\boldsymbol{V}^{kl}) \in \boldsymbol{\mathcal{R}}^{d \times d} \tag{8}$$

After the lossless conversion via DCT, filters in the frequency domain remain the same size as in the spatial domain. We propose to add a hidden layer to construct binary hash codes. Binary hash codes can effectively compress high-dimensional feature vectors, thereby improving computing efficiency and saving storage space.

For the h-dimensional output of the activation layer, we convert the feature vector into binary codes by a threshold value of 0.5, such as follows:

$$H^j = \begin{cases} 1 & Out^j(H) \geq 0.5 \\ 0 & otherwise \end{cases} \tag{9}$$

where $Out^j(H)$ is the value of the j-th node corresponding to the output vector of hashing layer, for each bit $j = 1 \ldots h$ (where h is the number of nodes in the hashing layer).

At the final stage, we compute the loss function of softmax classifier. It is measured by Euclidean distance between their corresponding features vectors of two face images. The smaller the Euclidean distance is, the higher level the similarity of the two images is.

IV. Experimental Results

In this section, we demonstrate the benefits of our approach. We start with introducing the deep learning framework and datasets we use for training and testing. We present our experimental results and verify the accuracy and the efficacy of our approach with comparison to several state-of-the-arts in the literature.

A. Deep Learning Tool Selection

Caffe (Convolutional Architecture for Fast Feature Embedding) is a fully open-source project that affords access to deep architectures. Caffe provides multimedia scientists and engineers with a clean and efficient framework for deep learning algorithms. It also contains multiple classic convolutional neural network models to be deploying. This paper adopts the reference CNN model supported by Caffe for training, testing and fine-tuning.

B. Datasets for Training CNN Model

CASIA-WebFace [28] is a large scale database that collects face images from Internet containing 10,575 subjects and 494,414 images. So far as we know, it is the largest dataset that is publicly accessible. Based on the database, the proposed deep CNN model can be implemented for learning face representations.

In order to speed up the training process, we covert the face images to the levelDB database format that Caffe framework supports. When training, the data is generally subjected to some standardized processing. A common method is to subtract the mean image of all images. Then we only need to generate the mean image for data preprocessing during subsequent training.

C. Datasets for Face Recognition

LFW (Labeled Faces in the Wild Home) is one of the most popular and challenging datasets for face recognition in

the wild [29]. The face pictures provided are all from natural scenes in life. More than 13,000 images of faces collected from the web. Each face has been labeled with the name of the person pictured. 1680 of the people pictured have two or more distinct photos in the data set. It makes face recognition more difficult in LFW because of the influence of multiple pose, light, expression, age and occlusions. Even the photos of the same person are very different.

LFW randomly selected 6000 pairs of faces to form a face recognition picture pair, among which 3000 pairs belong to the same person with 2 face pictures, 3000 pairs belong to different people with 1 face picture. Given a pair of photos during the test, the system need to give the answer of "yes" or "no" whether the two photos in the test are the same person. The face recognition accuracy can be obtained by comparing the ratio of the system answer to the real answer of the 6000 pairs of face test results. The LFW verification tests predict whether given pairs of images are from the same person.

D. Evaluation

In the following, we evaluate the accuracy of the proposed network using the LFW verification database. After several hundred thousand of iterations' training, we calculate the accuracy of our deep learning mode. Table I shows the LFW verification accuracy of the proposed approach in comparison with the baseline method of VGGNet [30]. We achieve face verification accuracy of 94.8%.

TABLE I. ACCURACY COMPARISONS ON LFW BENCHMARK

Method	Accuracy
VGGNet	93.6%
Proposed Framework	94.8%

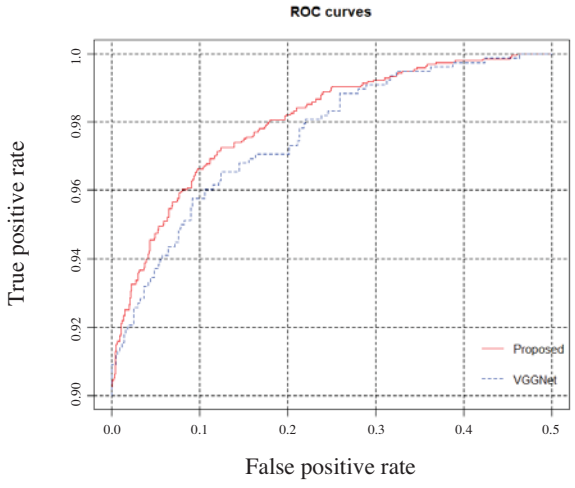

Figure 2. ROC curves comparison on LFW benchmark.

In addition to accuracy, we depict ROC curves as assistant indicators for comparison. The ROC curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings. ROC analysis is related in a direct and natural way to cost analysis of diagnostic decision making. Fig. 2 compares the ROC curves. Our approach gets a relative large area under the curve and achieves better performance.

V. CONCLUSION

This paper proposes a compressed convolutional neural network for face recognition. Our work not only provides a compact deep learning framework, but also significantly improves face recognition performance. In the implementation stage, we employ Caffe toolkit for training and deploying. With the traditional face verification pipeline, we achieved an extremely effective system with 94.8% face verification accuracy on LFW. Comparison results on various face recognition tasks demonstrate that the proposed compressed CNN framework has potential application in face recognition systems.

ACKNOWLEDGMENT

This work was supported by the Basic Research Foundation of the Education Department of Zhejiang Province, China (Grant No. Y201636459).

REFERENCES

[1] S. Omar, A. Ngadi, H. Jebur H, "Machine Learning Techniques for Anomaly Detection: An Overview," International Journal of Computer Applications, vol. 79, no. 2, pp. 33-41, 2013.

[2] L. Yann, Y. Bengio, and G. Hinton, "Deep learning," Nature, 521(7553), pp. 436-444, 2015.

[3] Y. Lécun, L. Bottou, Y. Bengio, P. Haffner, "Gradient-based learning applied to document recognition," in Proceedings of the IEEE, 1998, vol. 86, no. 11, pp. 2278-2324.

[4] R. Weng, J. Lu, Y. P. Tan, J. Zhou, "Learning cascaded deep auto-encoder networks for face alignment," IEEE Transactions on Multimedia, vol. 18, no. 10, pp. 2066-2078, 2016

[5] Y. He, K. Kavukcuoglu, Y. Wang, A. Szlam, Y. Qi, "Unsupervised feature learning by deep sparse coding," Computer Science, 2013.

[6] I. Sutskever, G. Hinton, G. Taylor, "The Recurrent Temporal Restricted Boltzmann Machine," in International Conference on Neural Information Processing Systems, 2008, vol.20, pp.1601-1608.

[7] X. L. Zhang, J. Wu, J, "Deep belief networks based voice activity detection," IEEE Transactions on Audio Speech & Language Processing, vol. 21, no. 4, pp. 697-710, 2013.

[8] M. Schuster, K. K. Paliwal, "Bidirectional recurrent neural networks," IEEE Transactions on Signal Processing, vol. 45, no. 11, pp. 2673-2681, 1997.

[9] A. Krizhevsky, I. Sutskever, G. E. Hinton, " Imagenet classification with deep convolutional neural networks," In: Advances in neural information processing systems, 2012. pp. 1097–1105.

[10] F. Radenović, G. Tolias, O. Chum, "CNN Image Retrieval Learns from BoW: Unsupervised Fine-Tuning with Hard Examples," in ECCV 2016. pp. 3-20.

[11] Y. Taigman, M. Yang, M. Ranzato, L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," in CVPR 2014, pp. 1701–1708.

[12] C. Szegedy,W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, "Going deeper with convolutions," in CVPR 2015, pp.1-9.

[13] L. N. Huynh, R. K. Balan, Y. Lee, "DeepSense: A GPU-based Deep Convolutional Neural Network Framework on Commodity Mobile Devices," in workshop on Wearable Systems and Applications, 2016, pp. 25-30.

[14] Y. Gao,H. J. Lee, “Viewpoint unconstrained face recognition based on affine local descriptors and probabilistic similarity,” Journal of Information Processing Systems, vol. 11, no.4, pp. 643-654, 2015.

[15] T. Hassner, S. Harel, E. Paz, R. Enbar, R, “Effective face frontalization in unconstrained images,” in CVPR 2014, pp.4295-4304.

[16] T. Ojala, M. Pietikäinen, T. Mäenpää, “Multiresolution gray-scale and rotation invariant texture classification with local binary patterns,” IEEE Transactions on Pattern Analysis & Machine Intelligence, vol. 24, no. 7, pp. 971-987, 2014.

[17] M. Turk, A. Pentland, “Eigenfaces for recognition,” Journal of Cognitive Neuroscience, vol. 3, no. 1, pp. 71-86.

[18] P. N. Belhumeur, J. P. Hespanha, D. J. Kriegman, “Eigenfaces vs. fisherfaces: recognition using class specific linear projection,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 19, no. 7, pp. 711-720, 1997.

[19] M. S. Bartlett, J. R. Movellan, T. J. Sejnowski, “Face recognition by independent component analysis,” IEEE Transactions on Neural Networks, vol. 13, no. 6, pp. 1450-1464.

[20] T. J. Stonham, “Practical face recognition and verification with WISARD,” Aspects of Face Processing, pp. 426-441, 1984.

[21] A. S. Raja, ,V. Josephraj, “A new multimodal recognition technique without subject's cooperation using neural network based self organizing map,” in International Conference on Electronics and Information Engineering, 2010, vol.1, pp. 511 -515.

[22] S. H. Lin, S. Y. Kung, L. J. Lin, “Face recognition/detection by probabilistic decision-based neural network,” IEEE Transactions on Neural Networks, vol. 8, no. 1, pp. 114-132., 1997.

[23] J. Saxe J, K. Berlin, “Deep neural network based malware detection using two dimensional binary program features,” in IEEE International Conference on Malicious and Unwanted Software, 2015, pp. 11-25.

[24] GLOROT X, BENGIO Y. Understanding the difficulty oftraining deep feedforward neural networks[J]. Journal of Machine Learning Research, 2010, 9: 249-256.

[25] Jia Y , Shelhamer E , Donahue J , et al. Caffe: Convolutional Architecture for Fast Feature Embedding[J]. 2014.

[26] Erhan D , Bengio Y , Courville A C , et al. Why Does Unsupervised Pre-training Help Deep Learning?[J]. Journal of Machine Learning Research, 2010, 11(3):625-660.

[27] K. R. Rao and P. Yip. Discrete cosine transform: algorithms, advantages, applications. Academic press, 2014.

[28] D. Yi, Z. Lei, S. Liao, and S. Z. Li. Learning face representation from scratch. Computing Research Repository (CoRR), 2014. arXiv: 1411.7923.

[29] G. B. Huang, M. Mattar, T. Berg, and E. Learned-Miller. “Labeled faces in the wild: A database for studying face recognition in unconstrained environments”. 2007. 1, 2, 5.

[30] Simonyan, Karen, and Zisserman, Andrew. “Very Deep Convolutional Networks for Large-Scale Image Recognition. ”. arXiv 1409.1556.