Hindawi
Scientific Programming
Volume 2020, Article ID 7846264, 9 pages
https://doi.org/10.1155/2020/7846264

Research Article

FaceFilter: Face Identification with Deep Learning and Filter Algorithm

Mohammed Alghaili,[1] Zhiyong Li,[1] and Hamdi A. R. Ali[2]

[1]*College of Information Science and Engineering, Hunan University, Changsha 410000, China*
[2]*Computer Science Department, Hajjah University, Hajjah, Yemen*

Correspondence should be addressed to Zhiyong Li; zhiyong.li@hnu.edu.cn

Received 21 November 2019; Accepted 2 April 2020; Published 1 August 2020

Academic Editor: Kifayat Ullah Khan

Although significant advances have been made recently in the field of face recognition, these have some limitations, especially when faces are in different poses or have different levels of illumination, or when the face is blurred. In this study, we present a system that can directly identify an individual under all conditions by extracting the most important features and using them to identify a person. Our method uses a deep convolutional network that is trained to extract the most important features. A filter is then used to select the most significant of these features by finding features greater than zero, storing their indices, and comparing the features of other identities with the same indices as the original image. Finally, the selected features of each identity in the dataset are subtracted from features of the original image to find the minimum number that refers to that identity. This method gives good results, as we only extract the most important features using the filter to recognize the face in different poses. We achieve state-of-the-art face recognition performance using only half of the 128 bytes per face. The system has an accuracy of 99.7% on the Labeled Faces in the Wild dataset and 94.02% on YouTube Faces DB.

1. Introduction

Recently, deep neural networks and especially convolutional neural networks (CNNs) have become the most commonly used method for feature representation and have achieved good results in face recognition problems. Face recognition can be divided into two categories: face verification, where two faces are presented and the system needs to verify whether these two faces belong to the same person, and face identification, where a face image is presented with an unknown identity and the system needs to determine this identity.

Most existing works that have focused on face recognition have achieved a high level of success [1–13]. However, if the pose is significantly changed or the face is presented at an angle, the individual cannot be identified.

Previous approaches to face recognition that are based on the discriminative classification model (face identification) are trained on a dataset of known identities, and an intermediate bottleneck layer is used as a representation for recognition. This approach generalizes a very large representation for each face, but some works have tried to reduce this dimensionality using PCA [10].

Another approach used in FaceNet [14] directly trained its output to obtain 128-D embedding using a triplet-based loss function based on LMNN [9]. These triplets comprise two matching faces and a nonmatching face. The aim of the triplet loss function is to separate positive results from negative ones by a certain distance margin.

In contrast, our approach uses an unsupervised learning technique to obtain 128 bytes per face and then passes these bytes to a filter in order to find the most suitable representation for each face. We then reduce the dimensionality of the representation to half of the 128 bytes, to match the original face with other faces to find the identity. This approach can identify a given face in different poses and can identify other faces that are most similar to the original identity.

Figure 1: Different poses of a single face with different types of illumination; all of these are identified correctly by the system.

As an illustration, Figure 1 shows a picture of a single individual at different angles and in different poses.

The remainder of this paper is organized as follows: Section 2 discusses the most important related work in face recognition. Our method is presented in Section 3, including a description of deep neural networks and our algorithm for handling the features. Sections 4 and 5 present some quantitative results and an evaluation of these.

2. Related Works

Our approach is similar to other recent works [3, 10, 14] in that it learns its representation directly from the face. However, instead of using a vector of features for reidentification, we reduce the vector representation to half of the features extracted for each face. We use a deep convolutional neural network architecture inspired by the NN4 FaceNet [14] and OpenFace [15] networks, but we remove the L2 normalization layer and instead use another fully connected layer.

There are an enormous number of studies of face recognition, and we will briefly discuss the most relevant works.

Huang et al. [16] proposed a convolutional deep belief network based on local convolutional restricted Boltzmann machines to learn a face representation. The learning method was unsupervised learning and the training was on an unlabeled natural image dataset. After that, they transfer the learned representation to a face identification through a classification method such as SVM.

Another attempt for face recognition was proposed by Taigman et al. [17]. This approach called DeepFace and it is one of the earlier large-scale applications of a 3D model for face recognition. They extracted the face representation using a nine-layer DeepFace model which mainly consists of two convolutional layers, three locally connected layers, and two fully connected (FC) layers with more than 120 million parameters using several locally connected layers without weight sharing. Their system was trained on 4.4 M 2D facial images of 4,030 identities and they achieved an accuracy of 97.35% on the benchmark LFW [18] dataset.

Schroff et al. proposed a CNN-based approach used for face recognition and clustering. This approach is called FaceNet [14] which is based on eleven convolutional and three FC layers. They have trained a deep convolutional network on a dataset of 200 M faces and 8 M identities and triplet loss function to directly optimize the embedding instead of an intermediate bottleneck layer as in the previous works. They have used triplets of roughly aligned matching/nonmatching face patches using an online triplet mining method, and they achieved the performance of state-of-the-art face recognition with 128 bytes for each face.

Sun et al. proposed another framework called DeepID [5, 6, 10] for face identification and verification. Their approach utilized an ensemble of shallower and smaller deep convolutional networks than DeepFace, i.e., every DCNN has four convolutional layers and uses 39, 31, and 1 patches, respectively, as an input. Their framework was trained on 202,599 images of 10,177 subjects. Their approach is considered as the first approach that achieved results that surpass human performance for face verification on the LFW dataset.

Parkhi et al. [19] collected a face dataset of 2.6 M 2D faces from 2,622 identities by proposing a new method for crawling the faces from the web. They presented a VGG-Face model consisting of 16 convolutional layers and three fully connected (FC) layers. The authors claimed that they achieved 98.95% accuracy on the LFW [18] dataset.

Deep 3D face recognition results have been represented by Kim et al. [20]. They fine-tuned the VGG-Face network [19] on 3D depth images. After that, they reported their results on three public datasets. They used an augmented dataset of 123,325 depth images to fine-tune VGG-Face. After that, they tested the model on Bosphorus [21], BU3DFE [5], and 3D-TEC (twins) [22] datasets. But their

results do not perform as the state-of-the-art results of the convolutional methods.

3. Method

3.1. Deep Convolutional Networks. We used a deep neural network structure called an NN4 neural network. Before they were input to the network, we resized all images to a size of $96 \times 96 \times 3$. These were used as input to the first convolutional layer, which has 64 kernels of size $7 \times 7 \times 3$ with stride 2. The second convolutional layer has 64 kernels of size $1 \times 1 \times 3$ with stride 2, and in the third convolutional layer, 192 kernels are used with size $3 \times 3 \times 3$ and stride 2. After these layers, an inception architecture was used in which there were six blocks labeled inception 3a, inception 3b, inception 3c, inception 4a, inception 4e, and inception 5a [23].

Since the input of the network was $96 \times 96 \times 3$ and the receptive field was small, the computational requirement was drastically reduced. The total number of parameters was 3,743,925, and the number of trainable parameters was 3,734,613, with 9,312 nontrainable parameters. We trained the network using a stochastic gradient descent (SGD) algorithm with a learning rate starting from 0.05 on a GPU. The model was trained on 202,599 face images of 10,177 subjects. Table 1 shows the network structure. Figure 2 depicts the model diagram while Figure 3 illustrates in detail the structure of inceptions used in this study.

Before training, we used the FaceNet [14] weights as a baseline in our network which used the triplet loss function in its training. Then, we used the Kullback–Leibler (KL) divergence loss functions to train our model as in Variational Feature Learning (VFL) [24] loss function. The difference between our loss function and VFL loss function is that, in VFL, they used the same input and output for two fully connected layers to be used to predict the mean μ and standard deviation σ of a Gaussian distribution. The mean μ and standard deviation σ are used to calculate the loss function which employed the Kullback–Leibler (KL) divergence loss. But in our training, since all input and output for the two fully connected layers are the same, we used one fully connected layer "fc1" in the network to be used to predict the mean μ and standard deviation σ of a Gaussian distribution. The mean μ and standard deviation σ are used to calculate the loss function as follows:

$$\mathrm{KL} = -\frac{1}{2}\sum_{i=1}^{n}\left(1 + \log\left(\sigma_i\right) - \mu_i^2 - \sigma_i\right), \tag{1}$$

where n denotes the output vector size, i.e., 128 in our training.

The network is trained with a softmax classifier for 200 epochs by using an Adam optimizer [25] and learning rate starting from 0.05. The training dataset divided into 70% for the training set and 30% for the validation set.

3.2. Face Reidentification Equations. Each original image x that we want to predict is represented by $f(x) \in R$ as a vector of 128 bytes indexes from 1 to 128. This can be expressed as in (1):

$$f\left(x_o\right) = \left(v^1, v^2, v^3, \ldots, v^n\right), \quad \text{where } n = 128, \tag{2}$$

where x_o is the original image that we want to predict and n is the number of features v in that vector. The vectors of the identities in the dataset will also be extracted, as it is expressed as in (2), and kept in a separate model file:

$$f\left(x_{id}^i\right) = \left(v_{id}^1, v_{id}^2, v_{id}^3, \ldots, v_{id}^n\right), \tag{3}$$

where $n = 128$ and x_{id}^i is the image of a particular identity id and i refers to the number of that identity in the dataset. After extracting the vectors, we will pass the vector of the original image to a filter to extract the most important values that can represent the original image. The filter works as a net to select the highest values among the features in the vector of the original image. It takes the values greater than zero with their corresponding position, i.e., indices of each value:

$$\mathrm{val},\ \mathrm{ind} = \sum_{i}^{n} f\left(v_o^i,\ \mathrm{index}_o^i\right), \quad \text{where } x_o^i > 0, \tag{4}$$

where n is the number of features v_o^i in the vector of the original image, i.e., 128, and i is the index of each feature in the vector. The selected features v_o^i which have values greater than zero will be stored in val while their corresponding indices will be stored in ind. So, we can select all features of each image in the identities of the dataset with the same indices of the selected features of the original image:

$$\mathrm{val}_{id}^i = \sum_{i=1}^{ni\,d} f\left(v_{id}^i,\ \mathrm{index}_{id}^i\right), \quad \text{for each } \mathrm{index}_{id}^i = \mathrm{index}_o^i, \tag{5}$$

where id refers to a particular identity in the set of identities i and $ni\,d$ is the number of features in each image of the identities. The selected features v_{id}^i of the identity id will be chosen if its indices index_{id}^i is equal to the indices of the selected features of the original image index_o^i and will be stored in val_{id}^i. Here, we do not need to select the values greater than zero for each identity in the dataset; rather, we just take the values corresponding to the indices of the largest values in the original image. This step is very important as the features of an eye, for example, may store in a particular index; consequently, we need to take the feature of that eye in each image in the dataset.

To recognize the identity, we will calculate the distance between the filtered values of the original image and the corresponding values of each identity image in the dataset. The lowest distance between the filtered values of the original image and a particular identity image both will have the same identity:

$$\mathrm{iden} = \min\left(\mathrm{val} - \begin{pmatrix} \mathrm{val}_{id}^1 \\ \mathrm{val}_{id}^2 \\ \mathrm{val}_{id}^3 \\ \vdots \\ \mathrm{val}_{id}^i \end{pmatrix}\right), \tag{6}$$

where i refers to the number of identities. It should be noticed that we have only weights of the images of all identities obtained by the model where these weights

Table 1: The structure of the neural network.

Layer	Size in	Size out	Kernel	Feature map	No. of parameters
Input	96 × 96 × 3				
ZeroPadding2D	96 × 96 × 3	102 × 102 × 3			
Conv1	102 × 102 × 3	48 × 48 × 64	7 × 7 × 3, 2	64	9472
Norm	48 × 48 × 64	48 × 48 × 64			
ZeroPadding	48 × 48 × 64	50 × 50 × 64			
Max pool	50 × 50 × 64	24 × 24 × 64	7 × 7 × 3, 2		
Conv2	24 × 24 × 64	24 × 24 × 64	1 × 1 × 3, 2	64	4160
Norm	24 × 24 × 64	24 × 24 × 64			
ZeroPadding	24 × 24 × 64	26 × 26 × 64			
Conv3	26 × 26 × 64	24 × 24 × 192	3 × 3 × 3, 2	192	110784
Norm	24 × 24 × 192	24 × 24 × 192			
ZeroPadding	24 × 24 × 192	26 × 26 × 192			
Max pool	26 × 26 × 192	12 × 12 × 192	3 × 3 × 3, 2		
Inception 3a	12 × 12 × 192	12 × 12 × 256			
Inception 3b	12 × 12 × 256	12 × 12 × 320			
Inception 3c	12 × 12 × 320	6 × 6 × 640			
Inception 4a	6 × 6 × 640	6 × 6 × 640			
Inception 4e	6 × 6 × 640	3 × 3 × 1024			
Inception 5a	3 × 3 × 1024	3 × 3 × 736			
Average pool	3 × 3 × 736	1 × 1 × 736	3 × 3 × 3, 1		
fc1	1 × 1 × 736	128			
Fc	128	1			

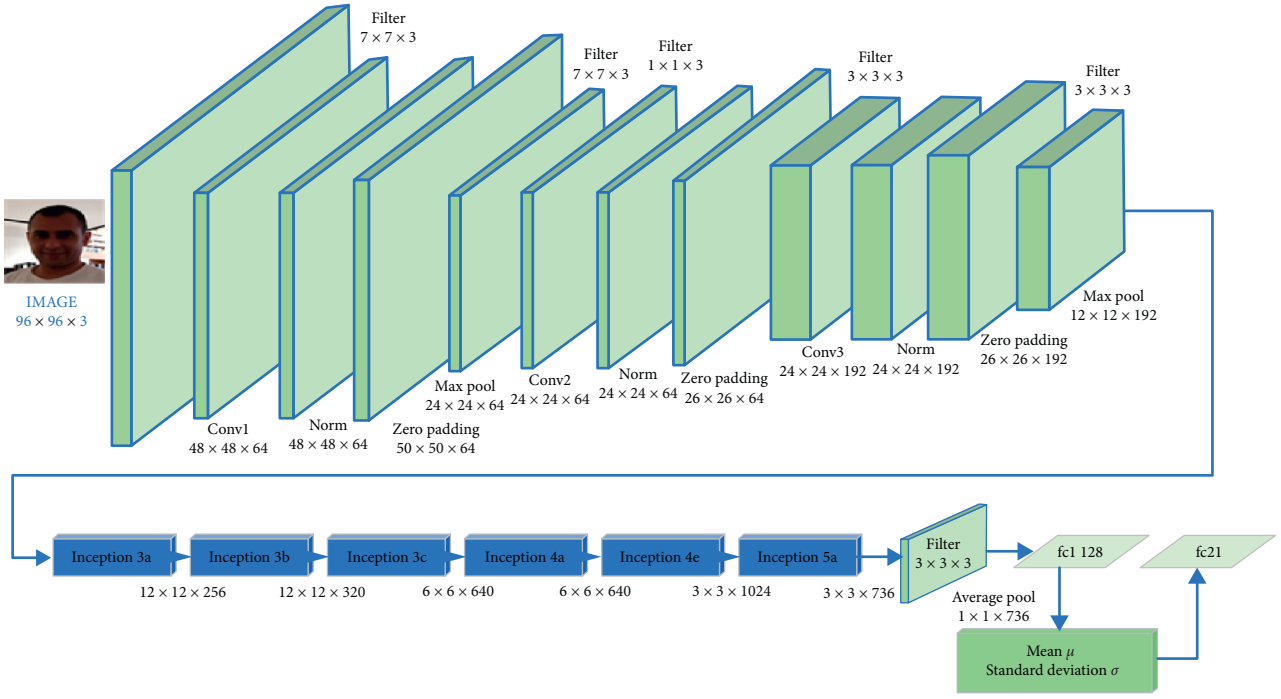

Figure 2: The structure of the model diagram.

have been kept in another model file called acknowledge base.

3.3. Image Aligning with Face Reidentification. Face detection and recognition still have many problems to identify the face especially when the face is aligned to down or to any other angle in the image. This problem can be solved by searching for a face in the image. If the image does not have a face, we will rotate the image step by step from 0 to 360°, where each step is rotated for 14°, until we find a face in that image and pass it as a new image. Therefore, the total number of steps is 25. In case we could not find a face in the image after rotating it, we will pass the image without

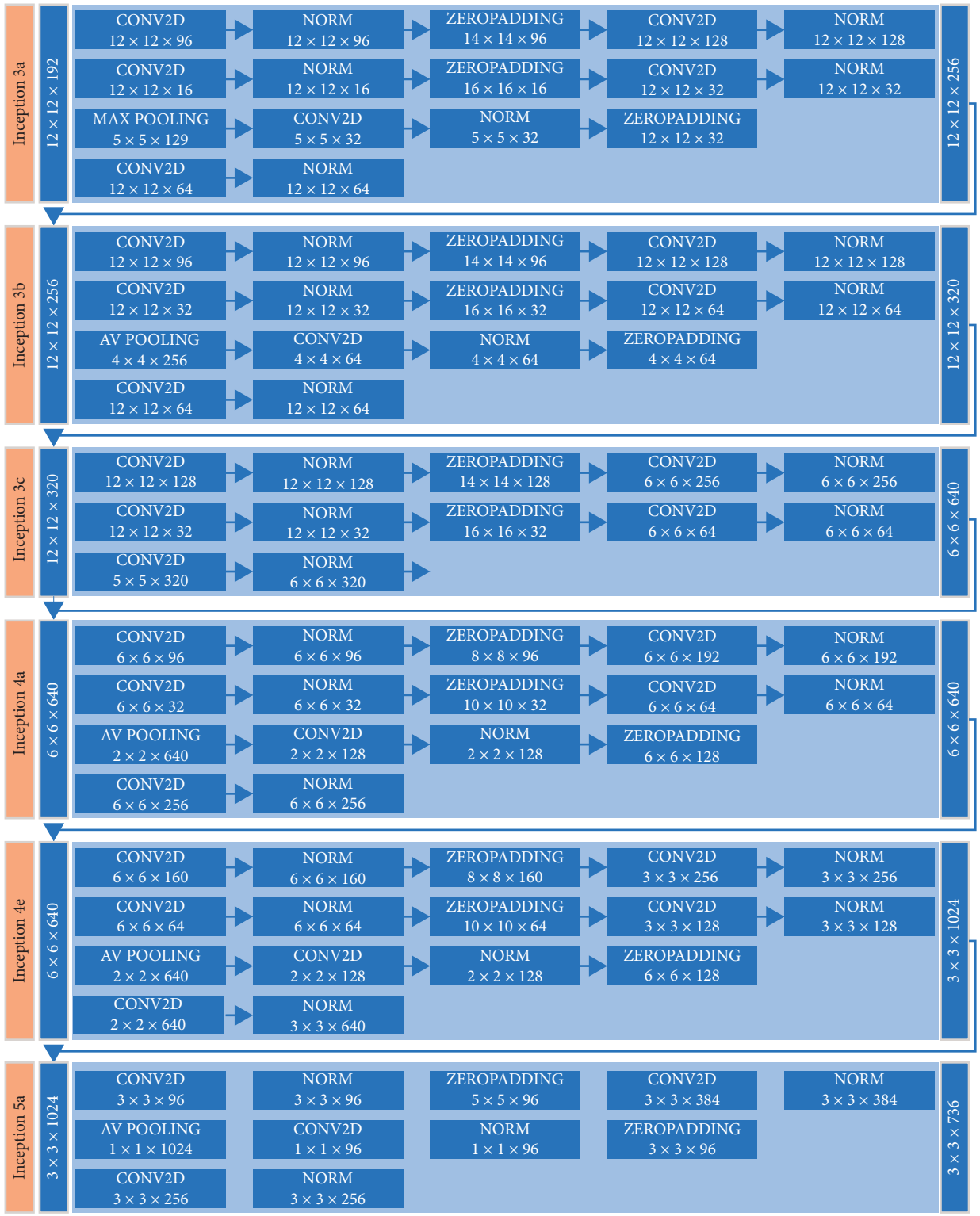

Figure 3: The structure of the inceptions used in the model.

rotating because there may be a face in the image where the face is in different poses and cannot be detected. Figure 4 shows an image with a face that the face detector cannot detect it, but after the rotations, we find a face while Figure 5 shows a face that cannot be detected after 360 rotations, so the original image will not be changed.

4. Evaluation

We used a neural network to extract the features of the faces. Feature extraction takes 128 bytes for each face and then finds the weights greater than zero from the original image with their corresponding indices and finds the other weights

Figure 4: The first image is the original image. The detector could not detect the face, so we rotated it. After two steps of rotations (rotate 1 and rotate 2), the detector still could not detect the face. After 16 steps of rotations, the detector detected the face successfully; therefore, the original image will be the image in (rotate 16). The identity of the face in rotate 16 has been recognized effectively by the system.

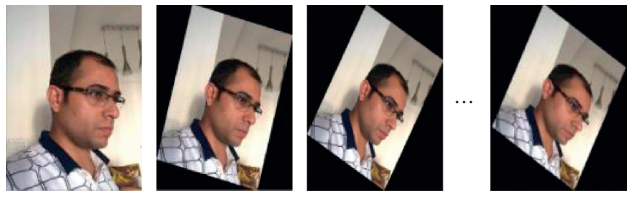

Figure 5: The first image is the original image. The detector could not detect the face, so we rotated it. After two steps of rotations (rotate 1, rotate 2), the detector still could not detect the face. After 25 steps of rotations (rotate 25), the detector still could not detect the face, and therefore, the original image will remain the first image. The identity of the face in rotate 16 has been recognized effectively by the system.

of the identities with the corresponding indices of the original image. The process of selecting weights larger than zero with their corresponding indices is called a filter process where the dimensional of the vector will be reduced to half of 128 bytes. After that, the distance of the filtered bytes of the original image with the bytes of each identity in the same indices of the original image is calculated to find the minimum number. The minimum number will refer to the identity of the original image. We evaluated the network on the Labeled Faces in the Wild and YTF [26] datasets.These two datasets have been used in most previous works which got a state-of-the-art results in their evaluation process. We achieved good results on these two datasets.

In the evaluation process, we extracted the features of each image in the dataset where each image has 128 features and stored them in a separate file. Then, we divided the weights into blocks by dividing the total weights by 128 to find the number of identities ind as in (7). Each block will contain 128 weights and will be treated as a single block for a single identity:

Table 2: Face verification of different methods on LFW.

Method	Accuracy (%)
DeepFace [17]	97.35
FaceNet [14]	99.63
High-dim LBP [27]	95.17
TL Joint Bayesian [28]	96.33
GaussianFace [29, 30]	98.52
DeepID [5]	97.45
DeepID2 [6]	99.15
DeepID2+ [10]	99.47
DeepID3 [30]	99.53
Our method	**99.70**

Table 3: Face verification of different methods on YTF.

Method	Accuracy (%)
Face reidentification [12]	90.41
DeepFace [17]	91.4
Face representation [11]	92.24
Deeply learned face [10]	93.2
FaceNet [14]	95.12
Our method	**94.02**

$$ind = \text{total} \times \frac{\text{weights}}{128}. \tag{7}$$

For the original image that we want to identify it, we extracted its 128 features using our model and passed these features to the filter to find the most important features for representation and reduce the dimensionality to the half. After choosing the positive values of each feature of the original image and taking their corresponding indexes, we will extract the features of each block from the features of the dataset according to the indexes of the positive features of the original image as in the following equation:

$$\text{TW} = \sum_{i1=0}^{128} W_{i\,dn}^{o_{i2}}, \quad \text{if } o_{i2} = i\,dn_i, \tag{8}$$

where o_{i2} is the indexes of the filter weights of the original image and the values of $i2$ are indexed from 0 to half of 128. W is the weight of the identity $i\,dn$. Finally, we applied (6) to identify the image.

5. Experiments

5.1. Dataset. We have used Celeb Faces Attributes Dataset (CelebA) as the training faces in our training. It consists of 10,177 identities and 202,599 faces. Before training, we extract the face of each image in the dataset using a face detector and then we resized it to the input size of our neural networks which is $96 \times 96 \times 3$ pixels.

LFW and YTF datasets used in the evaluation process. LFW is a database of face photographs for studying the problem of unconstrained face recognition. This database contains 13,233 images of 5,749 people detected and centered by the Viola–Jones face detector and collected from the web. YTF is a database of face videos designed to be used for

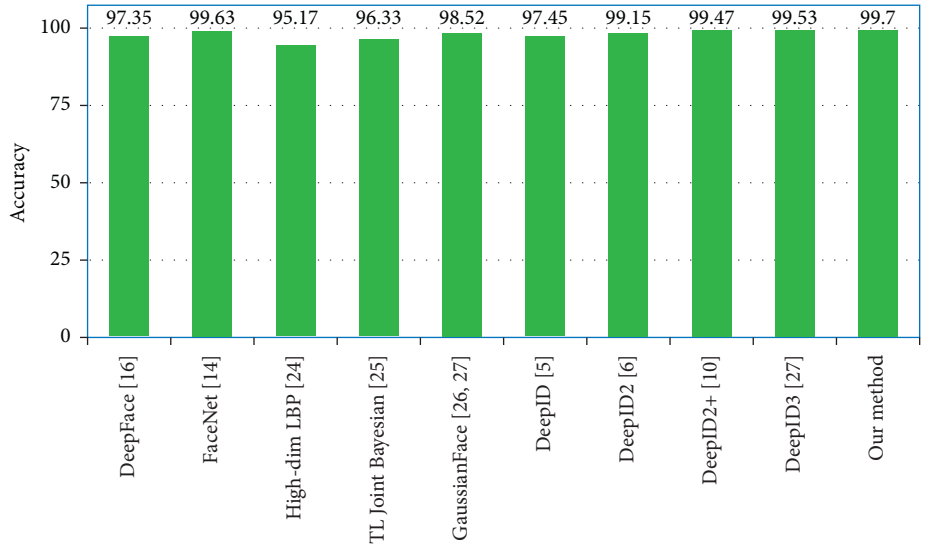

Figure 6: Chart illustrates the classification accuracy for different methods on LFW.

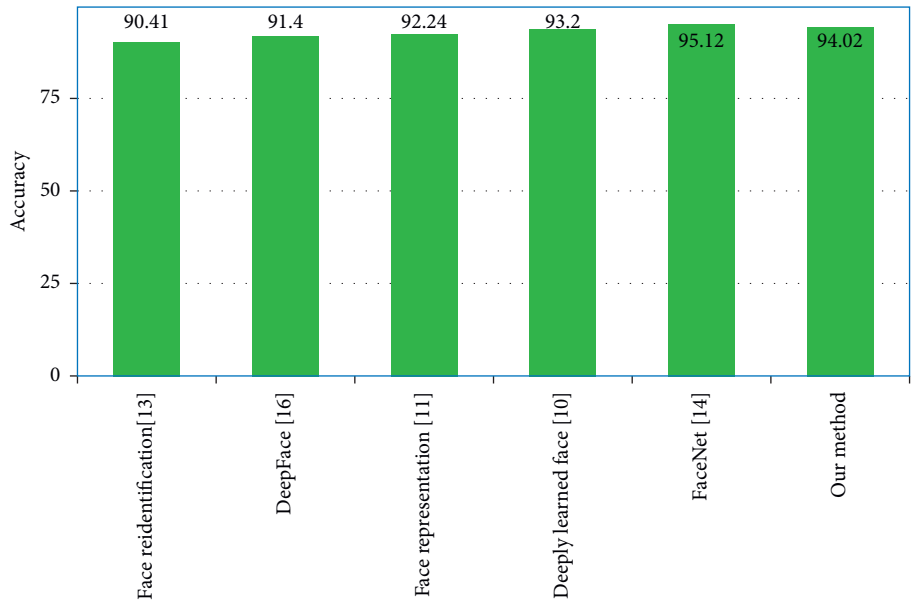

Figure 7: Chart illustrates the classification accuracy for different methods on YTF.

studying the problem of unconstrained face recognition in videos. This dataset contains 3,425 videos of 1,595 different people downloaded from YouTube. The shortest video contains 48 frames while the longest video contains 6,070 frames.

5.2. Various Dimensionalities. Various embedding dimensionalities were explored in previous studies [14], and accordingly, the dimension 128 has been selected as it gives the best accuracy. The comparison between four embedding dimensionalities, 64, 128, 256, and 512, shows that the difference in the performance is small. In this study, we explored the best dimensionality, i.e., 128, before and after applying the filter. After applying the filter to the dimension of 128, the dimensionality has reduced to the half of 128 with higher accuracy of the dimension of 128 by using our new algorithm.

5.3. Acknowledge Base Identities. In order to increase the number of identities without looking at the picture of any identity in the dataset again, acknowledge base model has been created to save the features of each identity. The features of any new identity will be saved in the acknowledge base model. This acknowledge base model will be used to know any other unseen face picture to predict the identity.

5.4. Effect of Face Detection. Most of the face detection frameworks have shown good results in face detection, whereas

there are still some limitations. Many faces have not been detected incorrectly using the most widely framework used for face detection. This limitation can affect negatively the results. Therefore, face detection still needs some improvements.

5.5. Performance on LFW and YTF. During the evaluation, the feature of every identity is extracted and kept in the acknowledge base. Any other extraction for any identity will be added to the acknowledge base with its corresponding label of that identity. Every time in the evaluation step, we took 200k images for test and kept their features with their corresponding labels in the acknowledge base with any previous features extracted for any identity. That means the acknowledge base model can store the features of all images in the dataset and it can find the single identity of any face among all these identities. We achieved a classification accuracy of 99.70% on the LFW dataset and 94.02% on the YTF dataset. Table 2 and Table 3 show the classification accuracy with some methods as compared to our classification accuracy on LFW and YTF. Figures 6 and 7 demonstrate a comparison chart for previous studies with LFW and YTF.

6. Conclusions

Deep neural network is used in this paper for face reidentification. The filter technique is used to select the most important features from the features extracted by the model. This method can identify the face in different poses and different levels of illumination. The rotation technique for 360° is used for the images that have the face in different angles, while this kind of rotation cannot be done in the augmentation method in deep learning.

We noticed that deep learning is very important to extract the features, but with well-prepared mathematical operations on the extracted features from the deep learning, it can increase the accuracy of the model.

Data Availability

The model, extraction of weight code, features saved in acknowledge base, and the equations for evaluation code are available in the following URL: https://drive.google.com/open?id=1pXMkhAOx9zV4n8ynmer2xlF5lLeQZ3Rz.

Disclosure

The funding sponsors had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; and in the decision to publish the results.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Authors' Contributions

Mohammed Al-Ghaili performed programming and wrote the manuscript. Zhiyong Li supervised the study. Hamdi A.R. Ali proofread the article.

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China (nos. 61672215 and 61976086), National Key R&D Program of China (no. 2018YFB1308604), and Hunan Science and Technology Innovation Project (no. 2017XK2102).

References

[1] Z. Zhu, P. Luo, X. Wang, and X. Tang, "Deep learning identity-preserving face space," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1489–1496, IEEE, Sydney, NSW, Australia, December 2013.

[2] Z. Zhu, P. Luo, X. Wang, and X. Tang, "Hybrid deep learning for face verification," in *Proceedings of the International Conference on Computer Vision*, pp. 113–120, IEEE, Sydney, NSW, Australia, December 2013.

[3] W. Liu, Y. Wen, Z. Yu et al., "Sphereface: deep hypersphere embedding for face recognition," in *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[4] Z. Zhu, P. Luo, X. Wang, and X. Tang, "Recover canonical-view faces in the wild with deep neural networks," pp. 1404–3543, 2014, https://arxiv.org/abs/1404.3543.

[5] Y. Sun, X. Wang, and X. Tang, "Deep learning face representation from predicting 10,000 classes," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pp. 1891–1898, IEEE, Columbus, OH, USA, June 2014.

[6] Y. Sun, Y. Chen, X. Wang, and X. Tang, "Deep learning face representation by joint identification-verification," *Advances in Neural Information Processing Systems*, pp. 1988–1996, 2014, https://arxiv.org/abs/1406.4773.

[7] Y. Taigman, M. Yang, M. A. Ranzato, and L. Wolf, "Web-scale training for face identification," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pp. 2746–2754, IEEE, Boston, MA, USA, June 2015.

[8] Z. Zhu, P. Luo, X. Wang, and X. Tang, "Deep learning and disentangling face representation by multi-view perceptron," in *Proceedings of the NIPS*, Quebec, Canada, December 2015.

[9] K. Q. Weinberger and L. K. Saul, "Distance metric learning for large margin nearest neighbour classification," *Journal of Machine Learning Research*, vol. 10, pp. 207–244, 2009.

[10] Y. Sun, X. Wang, and X. Tang, "Deeply learned face representations are sparse, selective, and robust," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pp. 2892–2900, IEEE, Boston, MA, USA, June 2015.

[11] D. Yi, Z. Lei, S. Liao, and S. Z. Li, "Learning face representation from scratch," pp. 1411–7923, 2014, https://arxiv.org/abs/1411.7923.

[12] Y. Wang, J. Shen, S. Petridis, and M. Pantic, "A real-time and unsupervised face re-identification system for human-robot interaction," *Pattern Recognition Letters*, vol. 128, 2018.

[13] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proceedings of the International Conference on Computer Vision*, pp. 3730–3738, IEEE, Santiago, Chile, December 2015.

[14] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: a unified embedding for face recognition and clustering," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pp. 815–823, IEEE, Boston, MA, USA, June 2015.

[15] B. Amos, B. Ludwiczuk, and M. Satyanarayanan, "Openface: a general-purpose face recognition library with mobile applications," Tech. Rep. CMU-CS-16-118, CMU School of Computer Science, Pittsburgh, PA, USA, 2016.

[16] G. B. Huang, H. Lee, and E. Learned-Miller, "Learning hierarchical representations for face verification with convolutional deep belief networks," in *Proceedings of the Computer Vision and Pattern Recognition (CVPR)*, pp. 2518–2525, IEEE, Providence, RI, USA, June 2012.

[17] Y. Taigman, M. Yang, M. A. Ranzato, and L. Wolf, "Deepface: closing the gap to human-level performance in face verification," in *Proceedings of the conference on computer vision and pattern recognition*, pp. 1701–1708, IEEE, Columbus, OH, USA, June 2014.

[18] 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," in *Proceedings of the Workshop on Faces in "Real-Life" Images: Detection, Alignment, and Recognition*, University of Massachusetts, Amherst, MA, USA, October 2018.

[19] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition," in *Proceedings of the BMVC*, vol. 6, University of Oxford, Oxford, UK, September 2015.

[20] D. Kim, M. Hernandez, J. Choi, and G. Medioni, "Deep 3D face identification," in *Proceedings of the International Joint Conference on Biometrics (IJCB)*, pp. 1703–10714, IEEE, Denver, CO, USA, October 2017.

[21] A. Savran, N. Alyüz, H. Dibeklioğlu et al., "Bosphorus database for 3D face analysis," in *Proceedings of the European Workshop On Biometrics and Identity Management*, pp. 47–56, Springer, Roskilde, Denmark, May 2008, Lecture Notes in Computer Science.

[22] V. Vijayan, K. W. Bowyer, P. J. Flynn et al., "Twins 3D face recognition challenge," in *Proceedings of the International Joint Conference on Biometrics (IJCB)*, pp. 1–7, IEEE, Washington, DC, USA, October 2011.

[23] C. Szegedy, W. Liu, Y. Jia et al., "Going deeper with convolutions," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pp. 1–9, IEEE, Boston, MA, USA, June 2015.

[24] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," 2013, https://arxiv.org/abs/1312.6114ICLR.

[25] D. Kingma and J. Ba, "Adam: a method for stochastic optimization," *Clinical Orthopaedics and Related Research*, https://arxiv.org/abs/1412.6980, 2014.

[26] L. Wolf, T. Hassner, and I. Maoz, "Face recognition in unconstrained videos with matched background similarity," in *Proceedings of the Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 529–534, IEEE, Providence, RI, USA, June 2011.

[27] D. Chen, X. Cao, F. Wen, and J. Sun, "Blessing of dimensionality: high-dimensional feature and its efficient compression for face verification," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pp. 3025–3032, IEEE, Portland, OR, USA, June 2013.

[28] X. Cao, D. Wipf, F. Wen, G. Duan, and J. Sun, "A practical transfer learning algorithm for face verification," in *Proceedings of the International Conference on Computer Vision*, pp. 3208–3215, IEEE, Sydney, NSW, Australia, December 2013.

[29] C. Lu and X. Tang, "Surpassing human-level face verification performance on LFW with GaussianFace," *Association for the Advancement of Artificial Intelligence*, pp. 3811–3819, 2014, https://arxiv.org/abs/1404.3840.

[30] Y. Sun, D. Liang, X. Wang, and X. Tang, "Deepid3: face recognition with very deep neural networks," pp. 3811–3819, 2015, https://arxiv.org/abs/1502.00873.