Article

Face Verification with Multi-Task and Multi-Scale Features Fusion

Xiaojun Lu [1], Yue Yang [1], Weilin Zhang [2], Qi Wang [1,*] and Yang Wang [1]

[1] College of Sciences, Northeastern University, Shenyang 110819, China; luxiaojun@mail.neu.edu.cn (X.L.); YangY1503@163.com (Y.Y.); wangy_neu@163.com (Y.W.)

[2] New York University Shanghai, 1555 Century Ave, Pudong, Shanghai 200122, China; wz723@nyu.edu

* Correspondence: wangqimath@mail.neu.edu.cn; Tel.: +86-024-8368-7680

Abstract: Face verification for unrestricted faces in the wild is a challenging task. This paper proposes a method based on two deep convolutional neural networks(CNN) for face verification. In this work, we explore to use identification signal to supervise one CNN and the combination of semi-verification and identification to train the other one. In order to estimate semi-verification loss at a low computation cost, a circle, which is composed of all faces, is used for selecting face pairs from pairwise samples. In the process of face normalization, we propose to use different landmarks of faces to solve the problems caused by poses. And the final face representation is formed by the concatenating feature of each deep CNN after PCA reduction. What's more, each feature is a combination of multi-scale representations through making use of auxiliary classifiers. For the final verification, we only adopt the face representation of one region and one resolution of a face jointing Joint Bayesian classifier. Experiments show that our method can extract effective face representation with a small training dataset and our algorithm achieves 99.71% verification accuracy on LFW dataset.

Keywords: deep convolutional neural networks; identification; semi-verification; multi-scale features; face verification

---

1. Introduction

In recent years, face verification methods based on deep convolutional neural networks(CNN) have achieved high performance [1-4]. Some researchers combine deep face representation and verification into one system, that is learning to map faces into similarity space directly [1]. However, it is much harder to learn the mapping in terms of a lack of training data. In this paper, we use the deep CNN as feature extractor and adopt extra classifier to make face representation more discriminative as in [2-4].

Face representation learning plays an important role in face recognition. It not only needs to have the ability to distinguish different identities, but also needs to make the distance of the same identity small enough. Inspired by DeepID2 [3] and Facenet [1], we design two CNN to extract face features which can have strong abilities of identification and verification. CNN1, which is only supervised by identification signal, is designed for setting different identity apart. And CNN2 is supervised by the combination of identification and semi-verification signals, which can make the distance of the same person small enough. Semi-verification is inspired by Triplet-loss in Facenet and verification signal in DeepID2, which represents the distance of pairs from the same identity.

In face pre-processing, it is hard to do great normalization for faces with variation caused by poses. In [5], Stan Z. Li proposes to use the distance of landmarks instead of eye centers for face normalization, which is said to be relatively invariant to pose variations in yaw poses. In our system, we combine this method with the most used eye centers method to do face normalization.

Inspired by [7][20], we add auxiliary classifiers to assist the training of CNN. Besides, these auxiliary classifiers provide multi-scale features for recognition. So a stronger feature can be obtained by concatenating these multi-scale features. Recently, most face verification methods catenate face representations of multi-resolutions and multi-regions based on deep CNNs to construct a feature

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 March 2017 doi:10.20944/preprints201703.0152.v1

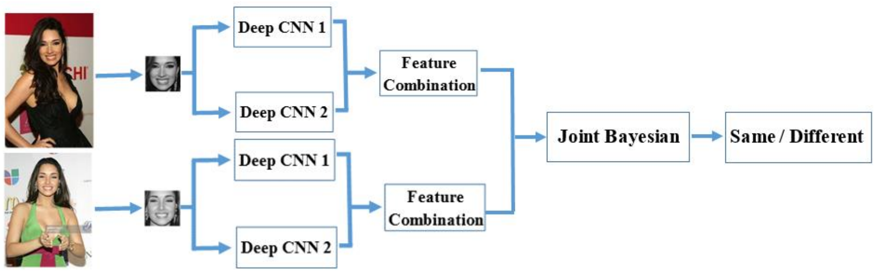


Figure 1. The overall framework of face verification.

with high dimension [3,4]. This will conduct high computation and a large burden of storage. In our work, we combine the face representations of two networks, and draw a compact feature as the final face representation. For each network, only one resolution and one region of a face are used. Due to the final feature combined multi-scale features come from two CNN trained by different signals, we called this multi-task and multi-scale features fusion.
The overall framework of our face verification method is illustrated in Figure 1. And our effective face representation joint Joint Bayesian classifier achieves high performance (99.71%) on LFW dataset with small training database.
The rest of this paper is constructed as follows: we introduce the semi-verification signal in section 2, which is used for supervising the training of one deep CNN. In section 3, we present two deep CNNs and the training algorithm. Face verification based on the proposed framework will be presented in section 4. In section 5, we present the performance of our method comparing with others based on deep CNN. Discussion and conclusion will be drawn in section 6.

2. The proposed Loss Function

Recently, there have been a lot of methods to add the verification information to the CNN for face verification task, such as contrastive loss[3], triplet loss[1], Lifted structured embedding[6]. The CNN trained with verification information can adjust the parameters end-to-end, so that the features generated from these CNN have greater discriminant power than those from normal networks which just use the cross entropy loss for objective. But contrastive loss and triplet loss need to pair the training sample. Contrastive loss[3] requires not only the positive pairs, but also negative pairs (where the positive pair refers to two different face images belong to one person, and the negative pair refers to two different face images belong to different persons). However, the number of positive pairs and the number of negative pairs are extremely unbalanced. For a dataset containing n individuals and m face images per person, the number of positive pairs is nC_m^2, and the number of negative pairs is $m^2C_n^2$. When $m \ll n$, $nC_m^2 \ll m^2C_n^2$, which means the number of negative pairs is much larger than the number of positive pairs. So unreasonable pairing can not improve the performance or even worse. Triplet loss[1] proposed online and offline methods for selecting training pairs, and each anchor uses semi-hard sample as its corresponding negative sample. Although Lifted structured embedding[6] does not need to pair the samples in a complex method, if the batchsize is N, a high cost $O(N^2)$ is entailed. The research community still does not have reasonable ways to pair samples. In order to solve the above problems, we propose Semi-verification Signal and corresponding pair selection method so that the verification information can be added to the CNN reasonably and efficiently.
Semi-verification signal means that only the pairs of the same identity will be used to compute verification loss. It minimizes the L2-distance between the face images of the same identity.

$$Semi\text{-}verification\ Signal = \frac{1}{\mid S \mid} \sum_{(i,j)\in S}^{|S|} \parallel f_i - f_j \parallel_2^2 \tag{1}$$

where S is an index set of face pairs belonging to the same identity. It does not contain pairs of different identities, which is the difference from verification signal. And it is the reason why we call it semi-verification signal. Because the negative pairs do not need to be selected, the imbalance between positive and negative pairs talked above exists no more. Reducing the intra-class variations and keeping the separable inter-class differences unchanged can also achieve the same purpose as the contrastive loss[3] does.
Supposing there are n different face images from one person, it will be C_n^2 positive pairs by getting faces two paired off. In this view, we only want to use a part of these pairs. However, randomly selected sample pairs cannot establish close relationships between all samples.

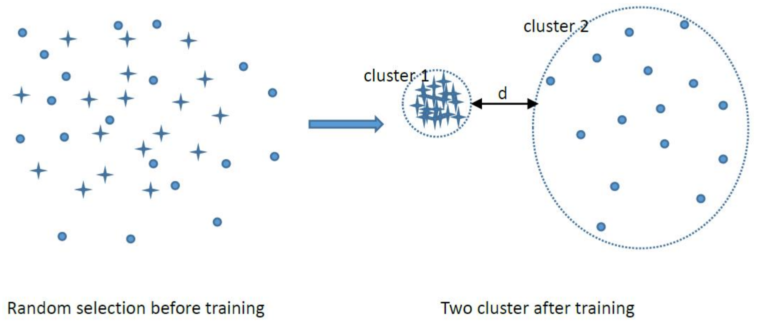


Figure 2. An example of random selection. Rounds present image pairs, which are selected for computing semi-verification loss. Diamonds are not selected. They belong to the same identity and there is no connection between circles and diamonds.

Supposing we randomly select m pairs from C_n^2 pairwise combination and there will be such a situation that some images do not appear in selected pairs any more. As shown in Figure 2, it will make images of this person divide into two clusters after training. As a result, the distance between m pairs of face images is small enough in one cluster, but in the other one will not. And the distance between two clusters will do not be small enough.

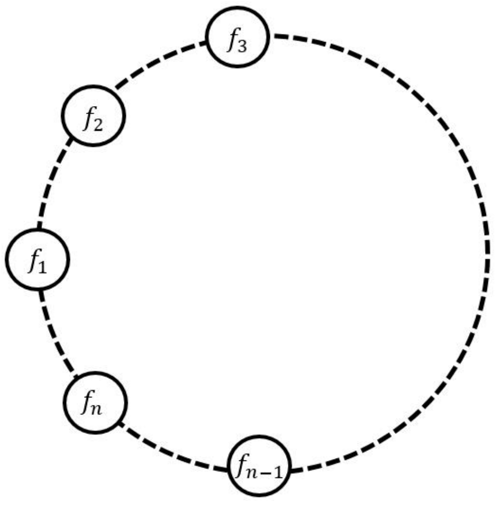


Figure 3. The method of instituting image pairs. $f_i (i = 1, 2, \cdots, n)$ present face image of a certain person.

For the purpose of solving the problems mentioned above, we institute positive pairs by creating a circle as pair selection method. Suppose there are N training samples of class i in the training data set, we number these samples $1, 2, \cdots, N$. CNN extracts features $f_j (j = 1, 2, \cdots, N)$ for these N samples.

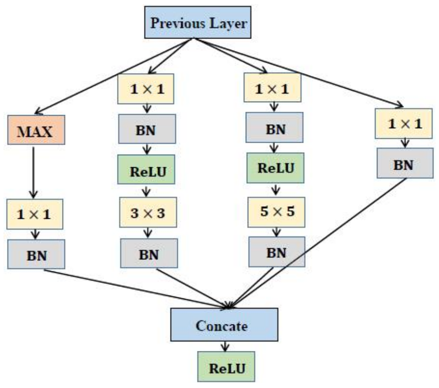


Figure 4. Inception used in NN1

As shown in Figure 3, one feature corresponding to one image is connected with its directly connected neighbors and there are no extra connections between it and other features. In other words, f_j only pairs with f_{j-1} or f_{j+1}. By this way, we can easily solve the problem above. On the one hand, it reduces the amount of computation to a certain extent $O(N)$. On the other hand, it establishes direct or indirect relationships between all face images.
In order to make the facial features extracted by CNN have strong identification and verification performance, two kinds of loss functions are used in this paper. One is identification loss, the other is joint identification and semi-verification loss.

$$Identification_{L}oss = -\sum_{i=1}^{n} -p_i log\hat{p}_i \tag{2}$$

where p_i is the target probability distribution, and $\hat{p}_i$ is the predicted probability distribution. If t is the target class, then $p_t = 1$, and $p_j = 0$ for $j \neq t$.
The joint identification and semi-verification loss can be formulated as follows:

$$Joint\ Loss = -\sum_{i=1}^{n} -p_i log\hat{p}_i + \frac{\lambda}{2 \mid S \mid} \sum_{(i,j)\in S}^{|S|} \parallel f_i - f_j \parallel_2^2 \tag{3}$$

where $-\sum_{i=1}^{n} -p_i log\hat{p}_i$ represents identification part, and $\parallel f_i - f_j \parallel_2^2$ denotes semi-verification signal. S is a index set of face pairs belonging to the same identity, and λ is a hyper-parameter used to balance the contributions of two signals.

3. The CNN Architectures and Detailed Parameters

Our face representation is a combination of features from two deep convolutional neural networks. The first CNN(CNN1) is supervised by identification signal only and the second one(CNN2) is supervised by joint identification and semi-verification signals.

3.1. Deep CNNs for Face Representation

Our deep CNNs contains two CNNs. CNN1 is constructed by ordinary convolution in shallow layers and Inception architectures in deep layers. Inception can be traced back to GoogleNet [7]. It is used for solving the problem of the increasement of high computation cost in the process of making a deeper network. It can not only increase the depth and width of convolution neural network at a certain computation cost, but also can extract multi-scale features for face representation. The framework of Inception used in CNN1 is shown in Figure 4.
As shown in Figure 4, we concatenate different sizes of convolutional layers $(1 \times 1,\ 3 \times 3,\ 5 \times 5)$ and Max-Pooling in one layer. A small sized of convolutional layer can focus more on local information,

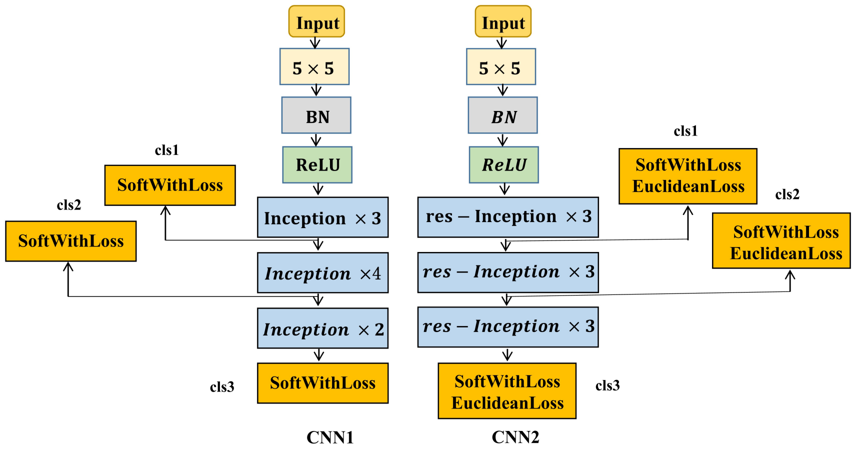


Figure 5. The overall frameworks of NNs.

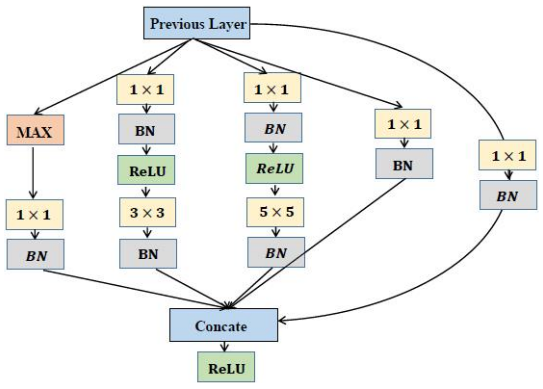


Figure 6. Res-Inception used in NN2.

and a lager one focuses on the global. It is the reason why Inception can extract multi-scale features. Besides, 1 × 1 reduction is used before 3 × 3 and 5 × 5, we also adopt Batch Normalization(BN)[8] after each convolution. BN can help our algorithm coverage at a high speed and solve the problem of overfitting.
When it comes to the activation function, we explore to use ReLU [9] after BN for each convolution in Inception. But for the output of Inception, which concatenates the results of multi-scale convolution, we adopt ReLU after concatenating layer. By this way, information can propagate more from the former layers to the later and the back propagation is more smooth [10]. The overall framework of CNN1 can be seen in Figure 5.
One difference between CNN1 and CNN2 in architecture is that we explore to use extra residual networks in CNN2. Residual network [11] is not only used for ordinary convolution but also for Inception, which is called res-Inception. The framework of res-Inception is shown in Figure 6. The reason why we want to introduce residual network in CNN2 is that it can make information propagation much smooth from former to later. And it also solves the problems of overfitting and low speed of coverage in training process. The overall framework of CNN2 is shown in Figure 5.
The deepening of the network will mostly cause vanishing gradient. Although residual network has been introduced, the difficulty of training CNN is still a problem. Inspired by [7][20], we explore two auxiliary classifiers in both deep CNNs as shown in Figure 5. We call them cls1 and cls2. For consistency, the original classifier is called cls3. Recently many researches discover the features from intermediary layers have a great complement power with feature from the top layer.

Table 1. The detailed parameters of convolution and Max-Pooling layers.

Type	size /stride	channel	#1 $\times$ 1	#3 $\times$ 3 reduction	#3 $\times$ 3	#5 $\times$ 5 reduction	#5 $\times$ 5	residual network	#Pooling reduction
Conv	$5 \times 5/1$	64							
Pooling	$3 \times 3/2$								
Conv Conv	$1 \times 1/1$ $3 \times 3/1$	64 92						92	
Pooling	$3 \times 3/2$								
res-Incep		256	64	96	128	16	32	256	32
res-Incep		480	128	128	192	32	96	480	64
Pooling	$3 \times 3/2$								
res-Incep		512	192	96	208	16	48	512	64
res-Incep		512	160	112	224	24	64	512	64
res-Incep		512	128	128	256	24	64	512	64
res-Incep		832	256	160	320	32	128	832	128
res-Incep		512	192	96	208	16	48	512	64
Pooling	$3 \times 3/2$								
res-Incep		832	256	160	320	32	128	832	128
res-Incep		1024	384	192	384	48	128	1024	128
Pooling	$6 \times 6/6$								

In our work, each cls has fully connected layer and loss layer, so we can get three features for each CNN. These three features produced from different layers correspond to different scales. In order to extract multi-scale face representation features, the final face representation of each CNN is formed by concatenating these three features. By this way, features from the former layers contain more local information and the latter ones are much more global. We can see in section 5.2, concatenating features from different layers will achieve a great improvement than using just one feature.

3.2. Detailed Parameters for NNs and Training Algorithm

The detailed parameters of CNN1 and CNN2 are shown in Table 1. We should point out that there is no usage of residual networks in CNN1, so the parameter of residual network for CNN1 should be none.

The training of CNN1 and CNN2 base on gradient descent algorithm with different supervisory signals. CNN1 is supervised by identification signal. Identification signal can make the face representation has a strong ability to distinguish different identities, and it is formulated in Eq. (2)

Although CNN supervised by identification can be used for verification, it cannot make distance of faces from the same identity small enough. Enlarge the distance between different identities and decrease that of the same are important for face verification task. So we joint identification and semi-verification signals to train CNN2. Unlike verification signal, semi-verification only uses samples from the same identity to compute loss. It can either decrease the distance of the intra-identity. The loss used for CNN2 is formulated in Eq. (3). So multi-tasks information can be obtained by concatenating features which come from CNN1 and CNN2.

For back propagation process, we compute the partial derivative of loss about parameters (w, b). Then, parameters can be updated through the partial derivative and learning rate η. Since CNN2 needs to calculate the distance between paired sampled features, we use the following method to construct each batch sample to facilitate calculation. Suppose that we have N (N is an even number) samples in a batch , numbering these N samples $1, 2, \cdots, N$. We pair the first $N/2$ samples of the batch with the last $N/2$ samples, ie. the sample image $i (i < N/2)$ in the batch belongs to the same person as the $N/2 + i$ sample. So when training CNN2, what we need to do is slicing the features

Table 2. The learning algorithm of NN.

```
Input: Training database $(x_i, y_i), i = 1, 2, \ldots, n$, where $x_i$ denotes face image
        and $y_i$ is identity label, $batchsize = 32$.
Initialized parameters $\theta_c$, learning rate $\eta(t)$, $t = 1$
While not coverage do:
        (1) $t = t + 1$, sample training set from training database $(x_i, y_i)$, the size
           of each training set is equal to batch size.
        (2) Compute forward process:
                $f_i = Conv(x_i, \theta_c)$
        (3) Slice the output of the last convolutional layer at the point $\lfloor batchsize/2 \rfloor$.
        (4) Compute identification loss and verification loss respectively:
                $IdentificationLoss = -\sum_{i=1}^{batchsize} -p_i log \hat{p}_i$
                $VerificationLoss = \frac{1}{\lfloor batchsize/2 \rfloor} \sum_{i=1}^{\lfloor batchsize/2 \rfloor} \parallel f_i - f_{i+\lfloor batchsize/2 \rfloor} \parallel_2^2$
        (5) Compute the value of loss function:
                $Loss = IdentificationLoss$ (for NN1)
                $Loss = IdentificationLoss + \lambda VerficationLoss$ (for NN2)
        (6) Compute gradient:
                $\bigtriangledown\theta_c = \frac{\partial loss}{\partial \theta_c}$
        (7) Update network parameters:
                $\theta_c = \theta_c - \eta(t) \bigtriangledown \theta_c$
End While
Output: Weight parameters
```

according to the the "batch" dimension and then calculating the distance between pairs. The details of our learning algorithm is shown in Table 2.

4. Face Verification with Classifier

For face representation, we adopt CNN1 and CNN2 to extract features of normalized faces respectively. And then, two extracted features are concatenated to be a relatively high dimension feature. The final representation of a face is formed by computing PCA reduction of the catenated feature. After face representation, we explore classifier to improve the discriminative ability of features.

4.1. Face Representation

Identifying faces with different poses is one of the hardest tasks in face verification, especially ones in yaw angles. As shown in Figure 7, compared with ordinary canonical faces, the distance between two eyes is smaller when faces in yaw angles, face normalization with eye centers can conduct the results to be just parts of faces especially. It has negative effects for face verification. However, Normalization by the distance between two landmarks of the nose is relatively invariant to yaw angles. But accurate landmarks of the nose is much harder to be detected than eyes.

In order to solve two problems mentioned above, we adopt different methods to deal with face normalization, and the flow of this strategy is described in Figure 8. First of all, an image is detected by a face detector based on CNN [13]. And then, we estimate the pose and locate face landmarks of the detected face through 3D poses algorithm [14]. For those faces, whose pose belongs to $[-15, 15]$, we adopt landmarks of eye centers for normalization. And for others, we use the distance of landmarks from noses for alignment. By this method, it can not only ensure the accuracy of face normalization for faces with no or small pose variance, but also ensure that results of faces with poses in yaw angles are the whole face regions.

After face pre-processing, we can obtain images only contain face regions. The normalized face

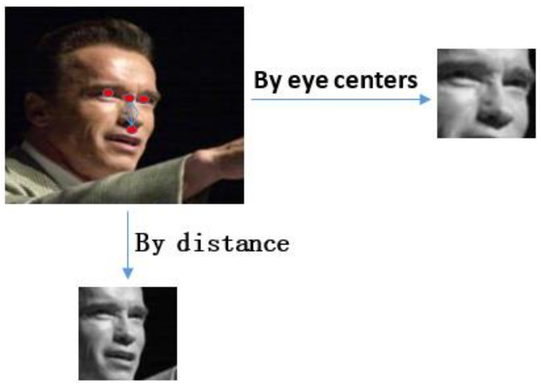


Figure 7. Face normalization by eye centers and distance.

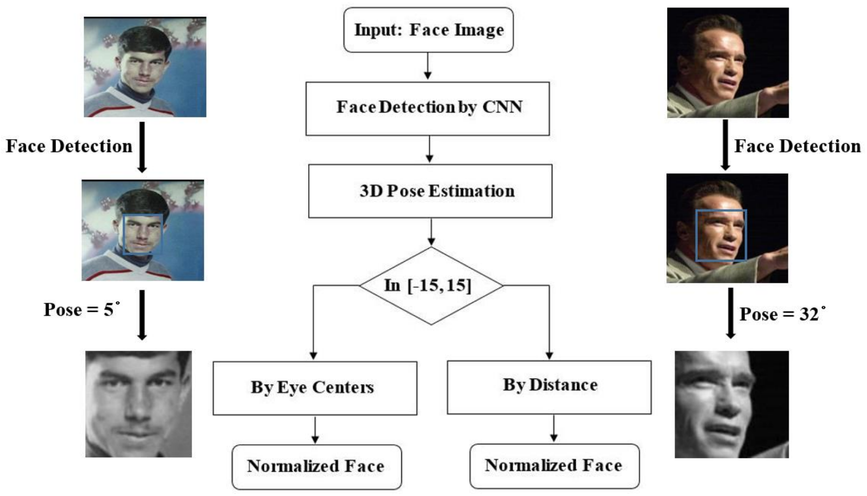


Figure 8. The flow chart of face normalization.

image is taken as the input of two networks, and outputs are concatenated to form the final face representation with PCA reduction.

4.2. *Face Verification by Two Classifiers*

In order to increase the ability of discriminant of face representation, we explore Cosine Distance and Joint Bayesian[21] respective. These two classifiers both compute the similarity of a pair of features f_i and f_j. That is,

$$CosineDistance = \frac{\langle f_i, f_j \rangle}{\|f_i\| \|f_j\|} \tag{4}$$

$$JointBayesian = -\log \frac{p(f_i, f_j | H_E)}{p(f_i, f_j | H_I)} \tag{5}$$

5. Experiments

Our experiments are based on Caffe[15], with NVIDIA GT980X GPU with 4GB of onboard memory, using single GPU. We train our model on TesoFaces database, which contains 10,000 face images of 15340 celebrities from the internet. And we evaluate our method on LFW [12], which is a challenging dataset for face verification in the wild.

5.1. *Experiment on hyper-parameter λ*

We research the balance between identification and semi-verification signals by a hyper-parameter λ. In our experiment, we try to explore 5 different values of $\lambda(\lambda = 0, 0.0005, 0.005, 0.05, 0.5)$. With the increasement of λ, the contribution of semi-verification to loss function is much greater. If $\lambda = 0$, then only identification signal will take effects.
We decide the value of λ in two views. In the first view, the decrease of loss function is used for measuring the performance of different values. And the second view is to use face verification accuracy on LFW dataset to determine λ is zero or not. Figure 9 shows the curve of the decrease of loss with different hyper-parameters. As we can see, the large value of $\lambda = 0.5$ and $\lambda = 0.05$ cause the loss doesn't fall any more. And loss of $\lambda = 0.005$ decrease very slowly. In contrast to the results of the other three values, the loss of $\lambda = 0.0005$ is decreasing at a high speed and will converge finally.

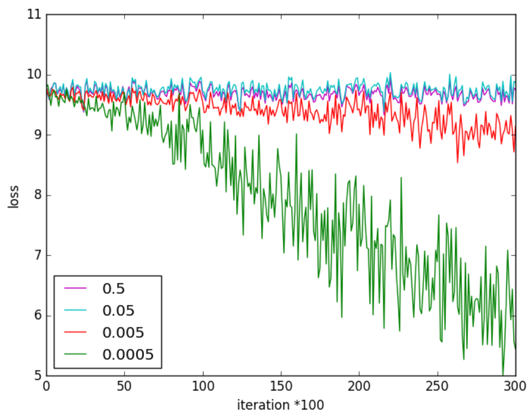


Figure 9. Loss of different values of λ.

Table 3 shows the face verification rate on LFW dataset by cosine distance when $\lambda = 0$ and $\lambda = 0.0005$. We can see that training with the combination of identification and semi-verification, that is $\lambda = 0.0005$, has a good performance. What's more, the weight of semi-verification should be

Table 3. Accuracy with different values of λ by Cosine Distance.

Signal	Identification ($\lambda = 0$)	Identification & Semi-verification ($\lambda = 0.0005$)
Accuracy	97.77%	98.25%

Table 4. Accuracy of different combined manners by Cosine Distance.

CNN1		CNN2	
combination manner	accuracy	combination manner	accuracy
cls 1	95.73%	cls 1	95.92%
cls 2	96.60%	cls 2	96.42%
cls 3	97.85%	cls 3	98.07%
cls 1 & cls 2	96.78%	cls 1 & cls 2	96.68%
cls 1 & cls 3	98.10%	cls 1 & cls 3	97.93%
cls 2 & cls 3	98.22%	cls 2 & cls 3	98.12%
cls 1 & cls 2 & cls 3	98.25%	cls 1 & cls 2 & cls 3	98.18%

a little small. That is to say, identification signal take much more important role in CNN training process. And semi-verification plays a role of regularization to a certain extent.

5.2. Learning Effective Face Representation

In order to learn the most effective face representation, we evaluate various combinations of features from auxiliary classifiers by Cosine Distance for face verification. We train each deep CNN with three auxiliary classifiers, and there are seven kinds of possible combinations of features for each CNN. As shown in Table 4, adding more features from auxiliary classifiers can improve the performance.
As the result, the feature of deeper layer has the much stronger ability of classification. And the face verification accuracy is much higher with the increasement of the number of features. Combining three features increases the accuracy by 0.40% and 0.11% over the best single feature for each CNN, respectively. And the trends of the performance show that with more auxiliary classifiers are used, the accuracy may be improved.
Furthermore, we compare face verification rate of each CNN and that of the combination of two CNNs. The result is shown in Table 5. It is obvious that the combination of two features from CNNs has the best performance. So our finally effective face representation is formed by concatenating features from CNN1 and CNN2, and each feature is a combination of three outputs of auxiliary classifiers. It shows that multi-task and multi-scale features fusion has a great power in face verification.

5.3. Evaluation on Classifiers

Learning more compact and discriminative features is the key for face verification task. For final face verification, we explore Cosine Distance and Joint Bayesian to improve the discriminative ability of features. The verification rates of two methods are shown in Table 6.
As the result, Joint Bayesian seems to be more appropriate for our face representation and it has a much better performance on face verification task. The reason why Joint Bayesian is better

Table 5. Accuracy of different face representation by Cosine Distance.

CNN1	CNN2	CNN1 & CNN2
98.25%	97.77%	98.5%

Table 6. Performance of Cosine Distance and Joint Bayesian.

Classifier	Cosine Distance	Joint Bayesian
Accuracy	98.50%	99.71%

Table 7. Accuracy of different methods on LFW dataset.

Method	Accuracy	Identities	Faces
Baidu[2]	99.77%	18K	1.2M
Ours	**99.71%**	**15K**	**0.1M**
FaceNet[1]	99.63%	NA	260M
DeepID3[16]	99.53%	16K	NA
Face++[17]	99.50%	16K	NA
DeepID2+[18]	99.47%	NA	NA
DeepID2[4]	99.15%	10K	NA
DeepID[3]	97.45%	10K	NA
DeepFace[19]	97.35%	NA	NA

than Cosine Distance is that the former one has taken the variance of intra and inter-identity into consideration. In other words, it can further increase differences of inter-identity and reduce that of intra-identity after face representation learning.

5.4. Comparision with Other Methods

To show the performance of our algorithm, we compare pair-wise accuracy on LFW dataset with the state-of-art deep methods.
In Table 7, we show the results of comparisons and the scales of the database used for training in different methods. As the result, our method achieves 99.71% test accuracy and it outperforms most deep face verification algorithms. The method in [2] is only 0.06% higher than us, but the number of faces they used for training is 12 times the amount of data we have. So our face verification method is a high product with a small cost.
Figure 10 compares ROC curves of different methods, and the curve of our algorithm is much more smooth than others. In the experiment, there are 17 wrong pairs in which 3 of them are wrongly labeled. So our the final pair-wise accuracy is 99.75%. For safety, in some occasions such as financing institutions, a large true positive rate when false accept rate is small is important. Though Baidu[2] got a better accuracy than us, according to Figure 11, we can see when false accept rate is small, our method will get a better true positive rate.

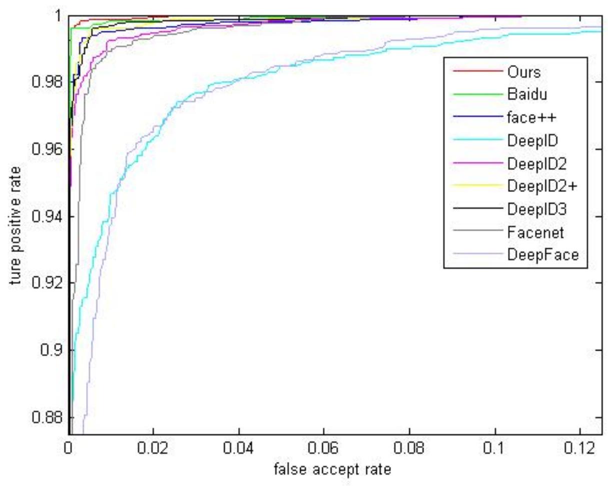
6. Conclusion and Discussion

In this paper, we propose a face verification method based on multi-task and multi-scale features fusion with Joint Bayesian classifier. And our algorithm has achieved high performance(99.75%) on LFW dataset. What's more, we only use one region and one resolution in our face representation process. And the training database we used is small. So our method is more practical in a real-life scenario.

Acknowledgement

This research is partially supported by National Natural Science Foundation of China (Grant No.31301086).

Bibliography



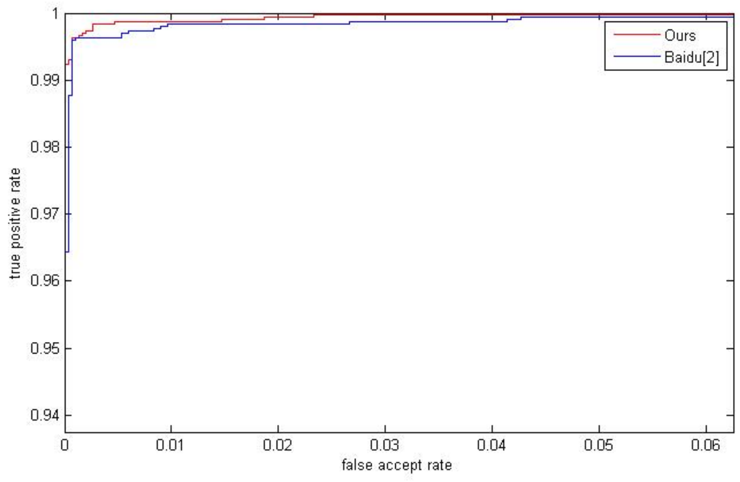


Figure 11. ROC curves of Baidu[2] and ours.

1. Schroff F., Kalenichenko D., Philbin J.: Facenet: A unified embedding for face recognition and clustering. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 815-823. (2015)
2. Liu J, Deng Y, Huang C. Targeting ultimate accuracy: Face recognition via deep embedding[J]. arXiv preprint arXiv:1506.07310, 2015.
3. Sun Y., Wang X., Tang, X.: Deep learning face representation from predicting 10,000 classes. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1891-1898. IEEE Press (2014)
4. Sun Y., Chen Y., Wang X.: Deep learning face representation by joint identification-verification. In: Advances in Neural Information Processing Systems, pp. 1988-1996. (2014)
5. Yi D, Lei Z, Liao S, et al. Learning face representation from scratch[J]. arXiv preprint arXiv:1411.7923, 2014.
6. Song H O, Xiang Y, Jegelka S, et al. Deep metric learning via lifted structured feature embedding[J]. arXiv preprint arXiv:1511.06452, 2015.
7. Szegedy C, Liu W, Jia Y, et al. Going deeper with convolutions[C]. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015: 1-9.
8. Ioffe S., Szegedy C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167. (2015)
9. Nair V., Hinton E.: Rectified linear units improve restricted boltzmann machines. In: Proceedings of the 27th International Conference on Machine Learning (ICML-10), pp. 807-814. (2010)
10. Szegedy C., Ioffe S., Vanhoucke V.: Inception-v4, inception-resnet and the impact of residual connections on learning. arXiv preprint arXiv:1602.07261. (2016)
11. He K., Zhang X., Ren S.:Deep Residual Learning for Image Recognition. arXiv preprint arXiv:1512.03385. (2015)
12. Huang B., Ramesh M., Berg T.: Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report pp. 07-49, University of Massachusetts, Amherst (2007)
13. Yang S, Luo P, Loy C C, et al. From facial parts responses to face detection: A deep learning approach[C]. Proceedings of the IEEE International Conference on Computer Vision. 2015: 3676-3684.
14. Ye M, Wang X, Yang R, et al. Accurate 3d pose estimation from a single depth image[C]. Computer Vision (ICCV), 2011 IEEE International Conference on. IEEE, 2011: 731-738.
15. Jia Y, Shelhamer E, Donahue J, et al. Caffe: Convolutional architecture for fast feature embedding[C]. Proceedings of the 22nd ACM international conference on Multimedia. ACM, 2014: 675-678.
16. Sun Y, Liang D, Wang X, et al. Deepid3: Face recognition with very deep neural networks[J]. arXiv preprint arXiv:1502.00873, 2015.
17. Zhou E, Cao Z, Yin Q. Naive-deep face recognition: Touching the limit of LFW benchmark or not?[J]. arXiv preprint arXiv:1501.04690, 2015.
18. Sun Y, Wang X, Tang X. Deeply learned face representations are sparse, selective, and robust[C]. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015: 2892-2900.
19. Taigman Y, Yang M, Ranzato M A, et al. Deepface: Closing the gap to human-level performance in face verification[C]. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2014: 1701-1708.
20. Lee C Y, Xie S, Gallagher P, et al. Deeply-Supervised Nets[C]. AISTATS. 2015, 2(3): 6.
21. Chen D, Cao X, Wang L, et al. Bayesian face revisited: A joint formulation[C]. European Conference on Computer Vision. Springer Berlin Heidelberg, 2012: 566-579.