Feature Aggregation Network for Video Face Recognition

Zhaoxiang Liu  
CloudMinds  
robin.liu@cloudminds.com

Huan Hu  
CloudMinds  
hans.hu@cloudminds.com

Jinqiang Bai  
Beihang University  
baijinqiang@buaa.edu.cn

Shaohua Li  
CloudMinds  
shaohua.li@cloudminds.com

Shiguo Lian  
CloudMinds  
sg.lian@163.com

Abstract

This paper aims to learn a compact representation of a video for video face recognition task. We make the following contributions: first, we propose a meta attention-based aggregation scheme which adaptively and fine-grained weighs the feature along each feature dimension among all frames to form a compact and discriminative representation. It makes the best to exploit the valuable or discriminative part of each frame to promote the performance of face recognition, without discarding or despising low quality frames as usual methods do. Second, we build a feature aggregation network comprised of a feature embedding module and a feature aggregation module. The embedding module is a convolutional neural network used to extract a feature vector from a face image, while the aggregation module consists of cascaded two meta attention blocks which adaptively aggregate the feature vectors into a single fixed-length representation. The network can deal with arbitrary number of frames, and is insensitive to frame order. Third, we validate the performance of proposed aggregation scheme. Experiments on publicly available datasets, such as YouTube face dataset and IJB-A dataset, show the effectiveness of our method, and it achieves competitive performances on both the verification and identification protocols.

1. Introduction

Video face recognition has become more and more significant in the past few years [41, 39, 31, 18, 25, 27, 42, 28, 19, 23, 38, 12, 44, 11, 32], which plays an important role in many practical applications such as visual surveillance, access control, person identification, video search and so on. Compared to single still image-based face recognition, further useful information of a single face can be exploited in the video. However, the video faces exhibit much richer uncontrolled variations, e.g., out-of-focus blur, motion blur, occlusion, varied illuminations and a large range of pose variations, which make video face recognition a challenging task. Hence, how to design a feature model which can effectively represent the video face across different frames becomes a key issue of video face recognition.

In video face recognition task, each subject usually has a varied number of face images. A straightforward approach would be to represent a video face as a set of face descriptors extracted by a deep neural network, compare every pair of face descriptors between two face videos [31, 34], and fuse the matching results across all pairs. However, this method would be considerably memory-consuming and inefficient especially for a large-scale recognition task. Consequently, an effective aggregation scheme, requiring minimal memory storage and supporting efficient similarity computation, is desired for this task, to generate a compact representation for a face video. And what is more, the aggregated representations should be discriminative, i.e., they are expected to have smaller intra-class distance than inter-class distance under a suitably chosen metric space.

So far, a variety of efforts on integrating information across different frames have been dedicated [18, 25, 27, 42, 28, 11, 32, 7, 8, 1]. Besides max pooling [8], average pooling [18, 25, 32, 8] may be the most common aggregation technique. However, it considers all frames of equal importance during feature aggregation, in which case the low quality frames with some misleading feature would degrade the performance of recognition. Considering of this problem, some other methods either just focus on high quality frames, i.e., feature-rich frames, while ignoring low quality frames, such as blurred faces, occluded faces and large pose faces [27, 12] or adaptively high weigh high quality frames while framesdown weigh low quality frames [42, 44].

Despite that those aggregation strategies have been shown to be effective in the previous works, we believe that an optimal aggregation strategy should not simply and
crude despise the low quality frames, because the low quality frames might even contain local discriminative features which can be complementary to high quality frames. In some sense, the low quality frames may be beneficial to video face recognition. Thus, the best aggregation result should be the composition of local discriminative features from low quality frames and other parts from high quality frames. Our intuition is simple and straightforward: an ideal algorithm should be able to emphasize the valuable part of the frame feature while suppressing the worthless part of the frame feature irrespective of the face quality during aggregation, i.e., it adaptively deals with each dimension of frame feature with different importance, not like NAN [42] that treats each dimension of equal importance for frame feature when aggregating. Let us imagine an extreme case: with some poor quality face images, e.g., a variety of large pose faces each with different pose, it is possible to aggregate these faces into a discriminative face representation for video face recognition.

To this end, we propose a new attention-based aggregation network which adaptively and fine-grained weighs the feature along each feature dimension among all frames to form a compact and discriminative face representation. Different from previous methods, we neither focus only on high quality frames nor simply weigh the feature on frame-level. Instead, we design a neural network which is able to adaptively and fine-grained measure the importance of each dimension of the feature among all frames.

Our major contributions can be summarized as follows:

- We propose a novel feature aggregation scheme for video face recognition, and reveal why it could work better. It is a generalized feature aggregation scheme, and may also serve as a feature aggregation scheme for other computer vision tasks.
- Based on the proposed aggregation scheme, we construct a feature aggregation network (as shown in Figure 1) composed of two modules trained end-to-end or one by one separately. One is the feature embedding module which is a frame-level feature extractor using deep CNN model. The other is the aggregation module which adaptively integrates the feature vectors of all the video frames together. Our feature aggregation network inherits the main advantages of the pooling techniques (e.g., average pooling and max pooling), could handle arbitrary input size and produces order-invariant, fixed-size feature representation.

- We demonstrate the effectiveness of our proposed aggregation scheme in video face recognition by various comparative experiments. Trained on publicly available dataset, such as YouTube face dataset and IJB-A dataset, our method takes a lead over the baseline methods and is a competitive method compared to the state of the art methods.

2. Related works and preliminaries

Since our work is concerned with order-insensitive video or image set face recognition, any other methods exploiting the temporal information of video sequence will not be considered here.

Early traditional studies attempt to represent the face videos or image sets as manifolds [1, 14, 16, 36, 35, 37] or convex hulls [6] and compute their similarities under corresponding spaces. While those methods may work well under constrained scenarios, they are usually incapable of dealing with large face variations.

Some other methods extract the local features of frames and aggregate them across multiple frames to represent the videos [18, 17, 24]. For example, PEP-based methods [18, 17] take a part-based representation by extracting and merging LBP or SIFT descriptors, and the method in [24] applies Fisher vector encoding to represent each frame by extracting RootSIFT [3, 22] and fuses across multiple different video frames to form a video-level representation.
These years, still image-based face recognition has gained great success thanks to deep learning techniques [31, 38, 34, 10, 20]. Based on this, some simple aggregation strategies are adopted in video face recognition. The methods in [31] and [34] utilize pairwise frame feature similarity computation and then fuse the matching results. Max- or average-pooling is used to aggregate the frame features in [25, 11, 7, 8]. Though DAN [27] proposes a GAN-like aggregation network which takes the video clip as input and reconstructs a single image as output to represent the video, the average pooling result of the video frames is employed to supervise the aggregation training. What is more, DAN is not suitable to tackle image set face recognition due to that a video face discriminator is used inside the GAN.

Recently, a few methods take a lead over the simple pooling techniques. The method in [12] utilizes discrete wavelet transform and entropy computation to select feature-rich frames from a video sequence and learns a joint feature from them. GhostVLAD [44] employs a modified NetVLAD [2] layer to downweigh the contribution of low quality frames. NAN [42] proposes an attention mechanism to adaptively weigh the frames, so that the contribution of low quality frames to the aggregation is down weighed. However, NAN considers each dimension of the feature vector to be of equal importance. These methods may lose some valuable information of the low quality images. This motivates us to seek a better solution in this paper.

Our work is inspired by NAN [42]. However, our aggregation scheme is a more generalized strategy, can fine-grained handle the feature vector on dimension level. Now, let us review the feature aggregation scheme of NAN [42]. Consider the video face recognition task on $n$ pairs of video face data $(S^i, y_i)_{i=1}^n$, where $S^i$ is a face video sequence or image set with varying image number $K_i$, i.e., $S^i = \{x^i_1, x^i_2, ..., x^i_{K_i}\}$ in which $x^i_k$, $k = 1, 2, ..., K_i$ is the $k$-th frame in the video, and $y_i$ is the corresponding subject ID of $S^i$. Each frame $x^i_k$ has a corresponding normalized feature representation $F^i_k$ extracted from the feature embedding module, and the aggregated feature representation becomes

$$r = \sum_{k=1}^{K_i} a^i_k F^i_k,$$  (1)

where $a^i_k$ is the linear weight generated from all feature vectors of a video, it can be formulated as

$$a^i_k = \frac{\exp(e^i_k)}{\sum_{j=1}^{K_i} \exp(e^i_j)},$$  (2)

where $e^i_k$ is the corresponding significance yielded via dot product with a kernel filter $q$ for each feature vector, it can be formulated as

$$e^i_k = q^T F^i_k.$$  (3)

Obviously, if $a^i_k = \frac{1}{K_i}$, Eq. (1) degrades to average pooling strategy.

### 3. Method

#### 3.1. The proposed aggregation scheme

We argue that each dimension of the feature vector shares the common weight as NAN does is not optimal. The ideal strategy should be able to adaptively weigh each dimension of feature vector separately. So we leverage a kernel matrix $Q$ to filter the feature vector $F^i_k$ via product, yielding a significance vector $E^i_k$, which describes the importance of each dimension of $F^i_k$. Assuming $F^i_k$ is an $M$-dimension vector, then, we can formulate $Q$ as

$$Q = \begin{bmatrix} q^T_1 \\ q^T_2 \\ \vdots \\ q^T_M \end{bmatrix}_{M \times M},$$  (4)

and formulate $E^i_k$ as

$$E^i_k = \begin{bmatrix} e^i_{1k} \\ e^i_{2k} \\ \vdots \\ e^i_{MK} \end{bmatrix}_{M \times 1} = QF^i_k = \begin{bmatrix} q^T_1 F^i_k \\ q^T_2 F^i_k \\ \vdots \\ q^T_M F^i_k \end{bmatrix}_{M \times 1}.$$  (5)

After softmax operation along each dimension, a positive

![Figure 2. Element-wise weighted sum of features.](image)


weight vector $A_k^i$ is generated as following

$$A_k^i = \begin{bmatrix} a_{1k}^i \\ a_{2k}^i \\ \vdots \\ a_{Mk}^i \end{bmatrix}_{M \times 1} \cdot \begin{bmatrix} \frac{\exp(e_{1k}^i)}{\sum_{j=1}^{K_i}\exp(e_{jk}^i)} \\ \frac{\exp(e_{2k}^i)}{\sum_{j=1}^{K_i}\exp(e_{jk}^i)} \\ \vdots \\ \frac{\exp(e_{Mk}^i)}{\sum_{j=1}^{K_i}\exp(e_{jk}^i)} \end{bmatrix}_{M \times 1},$$

(6)

where $a_{mk}^i$ denotes the linear weight of that $m$-th dimension of the feature vector contributes to aggregation result, and $\sum_{k=1}^{K_i}a_{mk}^i = 1, \forall m \in \{1, 2, \ldots, M\}$. So that the aggregated feature representation becomes

$$r = \sum_{k=1}^{K_i}A_k^i \odot F_k^i,$$

(7)

where $\odot$ represents element-wise product. Figure 2 shows the calculating process of $r$. $r$ turns out to be $\tilde{r}$ after $L_2$-normalization. Either cosine or $L_2$ distance can be used to compute the similarity.

From above formulas and Figure 2, we can clearly see the difference between our method and NAN is that we use a kernel matrix instead of a kernel vector to adaptively weigh the feature. Therefore, we can measure the importance of feature on dimension level without constraining each dimension to share the same weight just as NAN [42] does. Compared to NAN and other pooling techniques, our method is more flexible, and can make each dimension of one feature vector adaptively contribute to the aggregation feature. In theory, it can realize optimal feature aggregation after well trained. So, our method can deal with every frame fairly regardless of face quality, and make the best to exploit its any valuable or discriminative local feature to promote the video face recognition.

What is more, our method is a more generalized feature aggregation scheme. Obviously, if $a_{1k}^i = a_{2k}^i = \cdots = a_{Mk}^i$, Eq. (7) degrades to NAN, and if $a_{mk}^i = \frac{1}{K_i}$, Eq. (7) degrades to average pooling. And max pooling can also be regarded as a special case of our method.

### 3.2. The proposed feature aggregation network

Based on proposed aggregation scheme, we construct a feature aggregation network comprised of two modules. As shown in Figure 1, the network can be fed with a set of face images of a subject and produces a single feature vector as its representation for the recognition task. It is built upon a modern deep CNN model for frame feature embedding, and adaptively aggregates all frames in the video into a compact vector representation.

The image embedding module of our network adopts the backbone network of ArcFace [10] which greatly advances the image-based face recognition recently. The embedding module mainly consists of a ResNet50 which has an improved residual unit: BN-Conv-BN-PReLU-Conv-BN structure, while using BN-Dropout-FC-BN after the last convolutional layer. The embedding module produces 512-dimension image features which are first normalized to be unit vectors then fed into the aggregation module.

In order to obtain a better aggregation representation, a cascaded two attention blocks with nonlinear transfer is designed inside aggregation module as shown in Figure 3. Each attention block consists of a kernel filter and a nonlinear transfer. The kernel filter is implemented with a $FC$ layer, while nonlinear transfer with a $\tanh$ activation layer. Then $E_k^i$ becomes

$$E_k^i = \tanh(Q_2 F_k^i + b_2),$$

(8)

where $E_k^i$ is the output of the first block, it can be formulated as

$$E_k^i = \tanh(Q_1 F_k^i + b_1).$$

(9)

Therefore, besides kernel matrices $Q_1$ and $Q_2$, biases $b_1$ and $b_2$ are also trainable parameters of aggregation module. We have to point out that our cascaded attention blocks are totally different from NAN [42]’s in that our attention block uses an importance matrix while NAN uses an importance vector to weigh the feature vectors. In comparison, our method is more fine-grained than NAN to aggregate feature vectors. Furthermore, NAN aggregates the feature vectors twice, where the second attention block takes the aggregation result of the first attention block as input. However, our method only makes aggregation once.

In addition, our network has several other favorable properties. First, it is able to tackle arbitrary number of images for one subject. Second, the aggregation result $r$ which is of the same size as a single feature $F_k^i$ is invariant to the image order, keeps unchanged when the image sequence are reshuffled or even reversed, i.e., our network is insensitive to the temporal information of the video or image set. Third, it is adaptive to the input faces and whose all parameters are trainable through supervised learning with standard backpropagation and gradient descent.

![Figure 3. Cascaded two attention blocks.](image-url)
3.3. Network training

To make the training faster and more stable, we divide it into three stages (as shown in Fig.4). Firstly, we train the embedding module for single image face recognition task. In this stage, the cleaned MS-Celeb-1M dataset [10, 13] is used. Secondly, we train the whole network end-to-end for set-based face recognition task, and the VGGFace2 dataset [5] is used in this stage. In order to boost the capability of handling images of varying quality that typically occur in the wild, the VGGFace2 dataset is augmented in the form of image degradation, such as blurring or compression. Finally, we finetune the whole network end-to-end on the training set of the benchmark dataset.

Figure 4. Network Training. In stage 1, only embedding module is trained; then the trained embedding module is copied to stage 2 for end-to-end training; finally, the whole network is copied to stage 3 for end-to-end finetuning.

4. Experiments

4.1. Datasets and protocols

We conduct experiments on two widely used datasets including the YouTube Face dataset (YTF) [40], IJB-A dataset [15]. In this section, we will first introduce our implementation details, and then report the performance of our method on above two datasets.

4.2. Training details

Embedding module training: As aforementioned, the cleaned MS-Celeb-1M dataset [10, 13] which contains about 3.8M images of 85k unique identities is used to train our feature embedding network for the single image face recognition task. MTCNN [43] is employed to detect 5 facial landmarks in the face images. The faces are aligned to $112 \times 112$ by using similarity transformation according to the landmarks detected, and then fed into embedding network for training. The Additive Angular Margin Loss [10], which is a kind of modified softmax loss is used to supervise the training. After training, the classification loss layer is removed from the trained network. The rest network is fixed and used to extract a single fixed-size representation for the face image.

End-to-End training: We use the VGGFace2 dataset [5] to train the whole network end-to-end for the set-based face recognition task. The VGGFace2 Dataset [5] consists of about 3 million images, covering 8631 identities, and there are on average 360 face images for each identity. To perform set-based face recognition training, the image sets are built by repeatedly sampling a fixed number of images which belong to the same identity. All the images sampled are aligned using the same way as in the embedding module training. After alignment, the data augmentation is performed by image degradation. Following the same strategy as in GhostVLAD [44], four methods: isotropic blur, motion blur, decreased resolution and JPEG compression are adopted to degrade the face image for training. The Additive Angular Margin Loss [10] is also adopted to supervise the end-to-end training. In order to speed up the training, we initialize all the parameters of the aggregation module to be zero. That means the aggregation module begins with average pooling to search the optimal parameters.

Finetuning: All the video face dataset are also aligned by using MTCNN [43] algorithm and similarity transformation. Then the whole network is finetuned on the training set of each video face dataset using the Additive Angular Margin Loss [10].

4.3. Baseline methods

Since average pooling is a widely used aggregation method in many previous works [18, 25, 32, 8], we choose average pooling as one of our baselines. For fairness, the average pooling method shares common embedding module with our method after the whole network is finetuned on each benchmark dataset. We also choose NAN [42] as our

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy(%)</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>EigenPEP[18]</td>
<td>84.80 ± 1.40</td>
<td>92.60</td>
</tr>
<tr>
<td>DeepFace-single[34]</td>
<td>91.40 ± 1.1</td>
<td>96.30</td>
</tr>
<tr>
<td>DeepID2+[33]</td>
<td>93.20 ± 0.20</td>
<td>92.30</td>
</tr>
<tr>
<td>FaceNet[31]</td>
<td>95.12 ± 0.39</td>
<td>92.30</td>
</tr>
<tr>
<td>Wen et al.[38]</td>
<td>94.90</td>
<td>92.30</td>
</tr>
<tr>
<td>TBE-CNN[11]</td>
<td>94.96 ± 0.31</td>
<td>92.30</td>
</tr>
<tr>
<td>NAN[42]</td>
<td>95.72 ± 0.64</td>
<td>98.80</td>
</tr>
<tr>
<td>ADR[29]</td>
<td>96.52 ± 0.54</td>
<td></td>
</tr>
<tr>
<td>Deep FR[25]</td>
<td>97.30</td>
<td>92.30</td>
</tr>
<tr>
<td>AvgPool</td>
<td>95.70 ± 0.61</td>
<td>98.69</td>
</tr>
<tr>
<td>NAN*</td>
<td>95.93 ± 0.62</td>
<td>98.92</td>
</tr>
<tr>
<td>Ours</td>
<td>96.21 ± 0.63</td>
<td>99.1</td>
</tr>
</tbody>
</table>

Table 1. Performance evaluation on YTF benchmark. (NAN* represents the NAN [42] method we reproduce with our embedding module.)
another baseline. We reproduce the NAN which consists of cascaded two attention blocks as [42] describes. The reproduced NAN is trained in the same way as our method. The two baselines as well as our method produce 512-d feature representation for each video and compute the similarity in \(O(1)\) time. Besides the above two baselines, we also compare with some other state-of-the-art methods.

4.4. Results on YouTube Face Dataset

We first evaluate our method on the YouTube Face Dataset [40] which contains 3425 videos of 2595 different subjects. The lengths of videos vary from 48 to 6070 frames and the average length is 181.3 frames per video. The dataset is splitted into 10 folds, and each fold consists of 250 positive (intra-subject) pairs and 250 negative (inter-subject) pairs. We follow the standard verification protocol to test our method.

Table 1 shows the results of our method, the baseline and some other state of the art methods. We can see that our method outperforms the two baselines, reducing the error of the best-performing baseline: NAN* by 6.88%. This can be regarded as a proof of the effectiveness of our method. Our method also performs better than all the other state-of-the-art methods (including the original NAN [42]) except the deep FR methods and ADRL method [29]. The reason is that, the deep FR method benefits a lot from front face selection and triplet loss embedding with carefully selected triplets, and ADRL method [29] benefits from exploiting the temporal information from the video sequence. Compared to deep FR method, our aggregation method is more straightforward and elegant without hand-crafted rules. And compared to ADRL method [29], our method is order-invariant, can be used in more potential scenarios. It is noteworthy that our reproduced NAN also performs better than original NAN [42]. That is because both the embedding module and aggregation module of the reproduced NAN is trained end-to-end instead of separately, and compared to separate training, more training data is used during the end-to-end training stage.

4.5. Results on IJB-A Dataset

The IJB-A Dataset [15] contains 5712 images and 2085 videos, covering 500 subjects in total. The average numbers of images and videos per subject are 11.4 images and 4.2 videos. This dataset is more challenging than YouTube Face dataset [40] due to it covers large range of pose variations and different kinds of image conditions.

We follow the standard benchmark procedure for IJB-A to evaluate our method on both the ‘compare’ protocol for 1 : 1 face verification and the ‘search’ protocol for 1 : \(N\) face identification. The true accept rates (TAR) vs. false positive rates (FAR) are reported for verification, while the true positive identification rates (TPIR) vs. false positive identification rates (FPIR) and the Rank-N accuracies are reported for identification. Table 2 and Table 3 show the evaluation results of different methods for verification task and identification task respectively.

From above two tables, we can see that our method outperforms the two baselines by appreciable margins in both verification task and identification task, especially reducing the error of best-performing baseline by 7.12%, 11.97% and 17.83% at FAR=0.001, FAR=0.01 and FAR=0.1 respectively in verification task. These solidly prove the effectiveness of our method. Compared to all the state-of-the-art methods except for Ranjan et al. [26], our method performs a little better at FAR=0.001 and FAR=0.01, performs on par with them at FAR=0.01 where TAR values have almost saturated to a 99% mark, and beats all of them except on Rank-10 metric where our method is on par with them and the TPIR values have saturated to a 99.4% mark. Though Ranjan et al. [26] performed better than our method in three of all the eight metrics, they fused two deeper networks for feature embedding. One is ResNet101, the other is Inception ResNet-v2 that has 244 convolution layers, both of which are much deeper than our backbone ResNet50. What is more, they used more diverse datasets than us to train the feature embedding module. Not only the still image dataset but also other additional video dataset which is beneficial to video face recognition were utilized to train the networks. Besides, the reproduced NAN also outperforms the original NAN [42] as on YTF benchmark. It is noteworthy that the gap between our method and the original NAN [42] on IJB-A dataset is larger compared to the results on YTF dataset. This is because the face variations in IJB-A dataset is much larger than in YTF dataset, our method can extract more beneficial information for video face recognition.

5. Conclusion

We introduced a new feature aggregation network for video face recognition. Our network can adaptively and fine-grained weigh the input frames along each dimension of the feature vector and fuse them organically into a compact representation which is invariant to the frame order. Our aggregation scheme can make the best to exploit any valuable part of the features regardless of the frame quality to promote the performance of video face recognition. Experiments on YTF and IJB-A benchmarks show our method is a competitive aggregation method.

References

Table 2. Performance evaluation for verification on IJB-A benchmark. The true accept rates (TAR) vs. false positive rates (FAR) are reported. (NAN* represents the NAN [42] method we reproduce with our embedding module.)

<table>
<thead>
<tr>
<th>Method</th>
<th>1:1 verification TAR(%)</th>
<th>FAR=0.001</th>
<th>FAR=0.01</th>
<th>FAR=0.1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>86.8 ± 1.5</td>
<td>94.4 ± 0.9</td>
<td>-</td>
</tr>
<tr>
<td>DREAM [4]</td>
<td></td>
<td>81.3 ± 2.2</td>
<td>91 ± 1</td>
<td>96.4 ± 0.5</td>
</tr>
<tr>
<td>Triplet Embedding</td>
<td></td>
<td>83.6 ± 2.7</td>
<td>93.9 ± 1.3</td>
<td>97.9 ± 0.4</td>
</tr>
<tr>
<td>Template Adaptation</td>
<td></td>
<td>88.1 ± 1.1</td>
<td>94.1 ± 0.8</td>
<td>97.8 ± 0.3</td>
</tr>
<tr>
<td>NAN [42]</td>
<td></td>
<td>89.31 ± 3.92</td>
<td>94.20 ± 1.53</td>
<td>98.02 ± 0.55</td>
</tr>
<tr>
<td>VGGFace2 [5]</td>
<td></td>
<td>92.1 ± 1.4</td>
<td>96.8 ± 0.6</td>
<td>99.0 ± 0.2</td>
</tr>
<tr>
<td>GhostVLAD [44]</td>
<td></td>
<td>93.5 ± 1.5</td>
<td>97.2 ± 0.3</td>
<td>99.0 ± 0.2</td>
</tr>
<tr>
<td>Ranjan et al. [26]</td>
<td></td>
<td>95.2</td>
<td>96.9</td>
<td>98.4</td>
</tr>
<tr>
<td>AvgPool</td>
<td></td>
<td>88.82 ± 1.22</td>
<td>96.18 ± 0.92</td>
<td>98.16 ± 0.40</td>
</tr>
<tr>
<td>NAN*</td>
<td></td>
<td>93.12 ± 1.16</td>
<td>96.91 ± 0.83</td>
<td>98.71 ± 0.599</td>
</tr>
<tr>
<td>Ours</td>
<td></td>
<td>93.61 ± 1.51</td>
<td>97.28 ± 0.28</td>
<td>98.94 ± 0.31</td>
</tr>
</tbody>
</table>

Table 3. Performance evaluation for identification on IJB-A benchmark. The true positive identification rate (TPIR) vs. false positive rates (FAR) and the Rank-N accuracies are presented. (NAN* represents the NAN [42] method we reproduce with our embedding module.)

<table>
<thead>
<tr>
<th>Method</th>
<th>1:N identification TPIR(%)</th>
<th>FPIR=0.01</th>
<th>FPIR=0.1</th>
<th>Rank-1</th>
<th>Rank-5</th>
<th>Rank-10</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DREAM [4]</td>
<td></td>
<td>75.3 ± 3</td>
<td>86.3 ± 1.4</td>
<td>93.2 ± 1</td>
<td>-</td>
<td>97.7 ± 0.5</td>
</tr>
<tr>
<td>Triplet Embedding</td>
<td></td>
<td>77.4 ± 4.9</td>
<td>88.2 ± 1.6</td>
<td>92.8 ± 1.0</td>
<td>97.7 ± 0.4</td>
<td>98.6 ± 0.3</td>
</tr>
<tr>
<td>Template Adaptation</td>
<td></td>
<td>81.7 ± 4.1</td>
<td>91.9 ± 0.9</td>
<td>95.8 ± 0.5</td>
<td>98.0 ± 0.5</td>
<td>98.6 ± 0.3</td>
</tr>
<tr>
<td>NAN [42]</td>
<td></td>
<td>88.3 ± 3.8</td>
<td>94.6 ± 0.4</td>
<td>98.2 ± 0.4</td>
<td>99.3 ± 0.2</td>
<td>99.4 ± 0.1</td>
</tr>
<tr>
<td>VGGFace2 [5]</td>
<td></td>
<td>88.4 ± 5.9</td>
<td>95.1 ± 0.5</td>
<td>97.7 ± 0.4</td>
<td>99.1 ± 0.3</td>
<td>99.4 ± 0.2</td>
</tr>
<tr>
<td>Ranjan et al. [26]</td>
<td></td>
<td>92.0</td>
<td>96.2</td>
<td>97.5</td>
<td>98.6</td>
<td>98.9</td>
</tr>
<tr>
<td>AvgPool</td>
<td></td>
<td>86.43 ± 4.81</td>
<td>94.05 ± 1.02</td>
<td>95.69 ± 0.62</td>
<td>98.52 ± 0.45</td>
<td>99.04 ± 0.33</td>
</tr>
<tr>
<td>NAN*</td>
<td></td>
<td>87.92 ± 5.44</td>
<td>94.83 ± 1.01</td>
<td>97.23 ± 0.57</td>
<td>99.05 ± 0.58</td>
<td>99.24 ± 0.44</td>
</tr>
<tr>
<td>Ours</td>
<td></td>
<td>88.51 ± 5.86</td>
<td>95.18 ± 1.02</td>
<td>97.92 ± 0.32</td>
<td>99.23 ± 0.36</td>
<td>99.39 ± 0.25</td>
</tr>
</tbody>
</table>


