Improving ResNet-based Feature Extractor for Face Recognition via Re-ranking and Approximate Nearest Neighbor

Sheng-Hsing Hsiao
Dept of CSIE, National Taiwan University, Taiwan
kevin.hsiao@mirlab.org

Jyh-Shing Roger Jang
Dept of CSIE, National Taiwan University, Taiwan
roger.jang@gmail.com

Abstract

This paper proposes a framework for face recognition based on feature extractor from ResNet, together with other steps for performance improvement, including face detection, face alignment, face verification/identification, and re-ranking via Approximate Nearest Neighbor Search (ANNS). First, we evaluate two face detection algorithms, MTCNN, and FaceBoxes on three common face detection benchmarks, and then summarize the best usage scenario for each approach. Second, with certain preprocessing and postprocessing, our system selects the ResNet-based feature extractor, which achieves 99.33% verification accuracy on the LFW benchmark. Third, we use the penalty curve to determine the best configuration and obtain improved results of face verification. Based on the proposed preprocessing and post-processing, our method not only boosts accuracy from 84.3% to 86.5% in large inter-class variation datasets (CASIA-WebFace) but improves Rank-1 accuracy from 86.6% to 87.7% in large intra-class variation datasets (FG-NET).

1. Introduction

Face recognition is a generic term that could imply either face verification and identification or both. Verification tries to answer the question "*Is this person who they say they are?*" A synonym for verification is authentication, which checks whether or not the identity is in the database. Identification tries to answer the question "*Who is this person?*" The facial information is collected and compared to all the identities in a database.

In general, there are three steps for face recognition, 1) face detection, 2) face alignment and 3) feature embedding. Face detection is a fundamental step for many face-related applications. The essential properties of a face detector include detecting faces with a high degree of variability in scale, pose, illumination and expression. Alignment is the module that localizes facial landmarks including eyes, nose, lips, etc. These landmarks are used to align faces through rotation or scaling. Third, the face feature extractor encodes the facial information to a feature. These features are used to measure the degree of similarity between two faces.

Deep Convolutional Neural Networks (DCNNs) are now widely used because they learn hierarchical levels of features that correspond to different levels of abstraction. Increasingly large datasets are needed to provide a large number of faces featuring large variations. Widely used datasets include CASIA-WebFace[49], VGGFace[32], MS-Celeb-1M [15] and WIDER Face[48].

This study presents the following three contributions:

(1) By comparing MTCNN [52] and FaceBoxes [53] on major face detection benchmarks, our work chooses the suitable detector in different scenarios.
(2) This work uses CASIA-FaceV5[6] dataset as the authorized users and the Helen dataset as the intruders to evaluate face verification performance in different configurations.
(3) We present an ANNS-based re-ranking method to improve the performance on CASIA-WebFace [49] and FG-NET[11].

2. Related Work

We first introduce the pipeline of face recognition, including face detection, face alignment, and feature learning. We then discuss some methods for ANNS.

2.1. Face Detection

A good face detection method should be robust to variations in pose, rotation, illumination, scale, etc. DCNN based methods can be divided into two sub-classes: region-based and sliding window-based.

Region-based methods first generate a set of object-proposals and use CNN to classify each proposal as a face or not. R-CNN [12] obtained region proposals by selective search [42]. This approach has inspired some recent face detectors such as HyperFace [35] and All-in-One Face [36]. Faster R-CNN [12] uses a Region Proposal Network (RPN) to generate region proposals. Therefore, Li *et al.*[25] proposed a multi-task face detector based on Faster R-CNN framework.

Sliding window-based approaches directly output every face detection on multi-scale feature maps. Each detection is composed of the detection confidence and a bounding box. This approach does not need a separate region

proposal step and thus is faster than region-based approaches. Some methods [10,34] create an image pyramid at multiple scales to detect faces. Li *et al.* [24] used a cascade CNNs architecture for multiple resolutions. MTCNN adopted a cascaded CNNs that predict faces and landmark locations in a coarse-to-fine manner.

2.2. Face Alignment

Face alignment is the process of transforming a face into a canonical view and is usually done through the localization of facial keypoints. Facial keypoint detection methods can be divided into model-based and regression-based. **Model-based** methods create a representation of shapes during training and use models to fit facial landmarks during testing, including the classic Active Appearance Model (AAM) [30] and the Constrained Local Model (CLM) [4]. **Regression-based** methods directly fit the image appearance with the target output. Kazemi *et al.*[20] showed an ensemble of regression trees can be used to estimate the face's landmark positions and achieve real-time performance. CFAN [51] used a cascade of a few successive stacked auto-encoder networks.

2.3. Face Identification and Verification

In this section, we introduce some related works based on a deep network architecture for face recognition. There are two main parts of a face recognition system: face representation and similarity measure.

2.3.1. Face Representation

Deep networks have been shown to learn differences between two different faces. Huang *et al.*[17] proposed combining deep learning with traditional methods, such as Local Binary Patterns (LBP), achieving comparable results on the LFW [18] dataset.

In 2014, DeepFace [41] achieved state-of-the-art performance on the LFW benchmark. They used a proprietary face dataset consisting of 4M faces belonging to more than 4,000 identities. Similarly, in 2015, Google's FaceNet [37] used over 200M training samples for 3M people. It directly optimized the embedding itself by using triplets of roughly aligned matching/non-matching face patches. Parkhi *et al.*[32] designed a procedure to collect a large-scale dataset from the Internet, and trained this dataset via triplet loss function, achieving competitive results on both LFW and YTF [46] datasets.

2.3.2. Discriminative Metric Learning

Learning a classifier or a similarity metric is the next step after obtaining face features. For verification, features of two faces from the same person should be similar while features from different persons should be dissimilar.

Inheriting from object classification networks such as AlexNet [22], cross-entropy based softmax-loss are widely used for feature learning. But in recent years, the softmax loss has been found to often cause bias in the sample distribution. Several modified studies have been proposed to explore discriminative loss functions for more robust face representation. Prior to 2017, Euclidean-distance-based loss was the mainstream approach, but some loss functions like Angular/ cosine-margin-based loss were later designed to facilitate training procedures.

Euclidean-distance-based Loss: This loss is a metric learning method that maps features into Euclidean space to cluster the same person while differentiating other people. The most intuitive way is contrastive loss [40,49], using pairs of images to train a feature embedding where positive pairs are closer and negative pairs are farther apart. But the margin parameters of the contrastive loss are difficult to choose. Triplet loss, however, tries to enforce a margin between each pair of faces for one identity to all other faces. FaceNet [37] from Google and Parkhi *et al.*[32] both used triplet loss to embed features into a discriminative space and achieved improvements for face verification. Nevertheless, both contrastive loss and triplet loss require considerable time to achieve converge due to their ineffective sampling policies. Therefore, Wen *et al.*[45] introduced the Center loss, which provides a learned center for each class and penalizes the distances between the deep features and their corresponding class centers.

Angular/cosine-margin-based Loss: In 2016, Liu, *et al.*[27] proposed the large-margin softmax (L-Softmax) loss, reformulated from the original softmax loss. L-softmax makes the classification more rigorous to produce a decisive margin. The following year, SphereFace [26] proposed angular softmax (A-Softmax) loss to further normalize the weight by its L2 norm such that the normalized vector will lie on a hypersphere to learn angularly discriminative features. To solve the difficulty of optimizing L-Softmax and A-Softmax, ArcFace [8] added the angular margin while Wang *et al.*[43] added a cosine margin. These are easy to implement and can converge without softmax supervision.

2.4. Approximate Nearest Neighbor Search

Rapid and constant increases in data volumes significantly increases computation time and complexity. K-Nearest Neighbor Search (K-NNS) is a common approach for information retrieval. A naïve brute force approach is to compute the distance between the query and every element in the dataset. But the complexity of the approach scales linearly with the number of elements in storage. Approximate Nearest Search (ANNS) was proposed to overcome the "*curse of dimensionality*", by providing a good approximation rather than the exact nearest element. ANNS methods can be divided into three

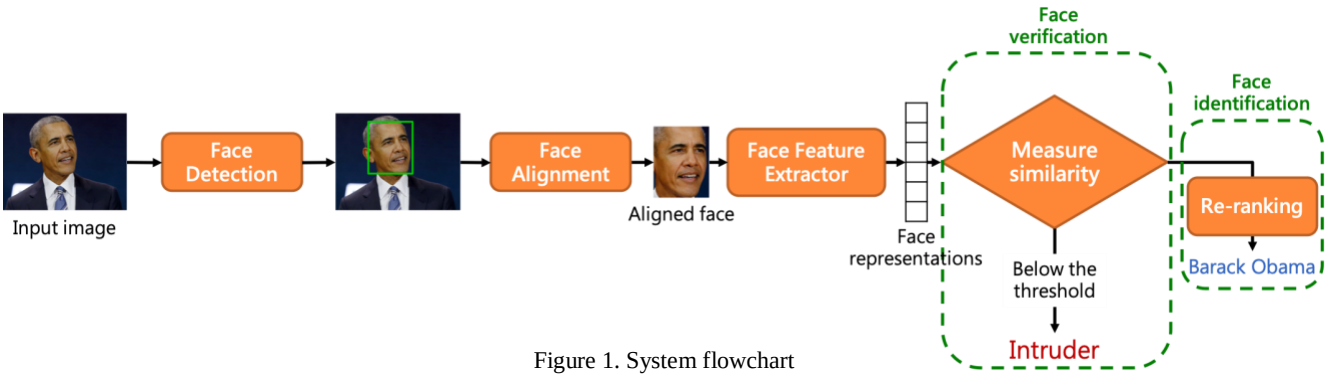


Figure 1. System flowchart

classes: Hashing-based, Partition-based and Graph-based.

Hashing-based: Hashing-based approaches project data to low-dimensional representations. Thus, each element could be encoded as a hash code. Locality-sensitive hashing (LSH) is a basic hashing-based approach, referring to a family of functions to hash similar data to the same bucket with high probability, while dissimilar data points are likely to be in different buckets. LSH-based methods require a good locality-sensitive hash function. In Euclidian distance measurement, many hash functions have been proposed, such as [1] and [2].

Partition-based: Methods in this category divide the high dimensional space into several disjointed regions hierarchically. If query q is located in a region r_q, then its nearby neighbors should be in region r_q or near r_q.

VP-Tree [50] and M-Tree [7] rely on the distance from the data point to pivots. Annoy[3] and FLANN [31] divide the space recursively by the hyperplane with random direction.

Graph-based: A representative of this type is the Navigable Small World graph (NSW) [29]. There are two types of edges in NSW: *short-range links* for greedy search, and *long-range links*, which define the small-world navigation property. The construction of NSW iteratively inserts new vertices into the graph. For each new vertex, we locate the position and then search for its k nearest neighbors in the current graph. The edges connecting the new vertex and its k nearest neighbors are defined as *short-range links*. At the i iteration, the *short-range links* of the $i-1$ iteration become *long-range links*. HNSW [28] is an extension of NSW and a multi-layer structure consisting of a hierarchical set of proximity graphs for nested subsets of the stored elements.

3. Methods

This section presents the pipeline of our system. An overview of the pipeline is displayed in Fig. 1. The two different face detectors are introduced in section 3.1. The use of 2D face alignment to obtain canonical representations of faces is described in section 3.2. Section 3.3 describes the ResNet-based face feature extractor used. Lastly, section 3.4 proposes the re-ranking method for the retrieval of face recognition using HNSW.

3.1. Face Detection

This section briefly describes the two face detection methods applied, MTCNN and FaceBoxes.

MTCNN: Multi-task cascaded CNN (MTCNN) includes three-stage coarse-to-fine CNNs trained on WIDER Face [48]. MTCNN initially resizes the image to different scales to build an image pyramid before passing an image to the first CNN. Stage 1 is the Proposal Network (P-Net), to obtain the candidate windows. After that, we use the estimated bounding box regression vectors to calibrate the candidates, and then use non-maximum suppression (NMS) to eliminate highly overlapped windows. Stage 2 uses a Refine Network (R-Net) to reject false candidates and performs bounding box calibration and NMS. Stage 3: Output Network (O-Net) is similar to R-Net but aims to locate the positions of five facial landmarks.

FaceBoxes: FaceBoxes consists of Rapidly Digested Convolutional Layers (RDCL) and Multiple Scale Convolutional Layers (MSCL). RDCL is designed to rapidly shrink the input spatial size by choosing a suitable kernel size. Furthermore, adoption of the CReLU [38] activation function can double the number of output channels by simply concatenating negated outputs while significantly increasing the processing speed with a negligible decline in accuracy. MSCL consists of several layers and aims to enrich the receptive fields and discretizing anchors over different layers to handle various scales of faces.

3.2. Face Alignment

Our system applies a 2D face alignment procedure. The ensemble of regression trees [20] is used to find 68 facial landmarks, and implementation of the algorithm is available in the dlib [21] toolkit. As seen in Fig. 2, we first use the eye region landmarks to compute the center of each eye and calculate the angle between the two eyes. We then use the midpoint between the two eyes to rotate the face. Finally, we get the canonical view of a face after rotating

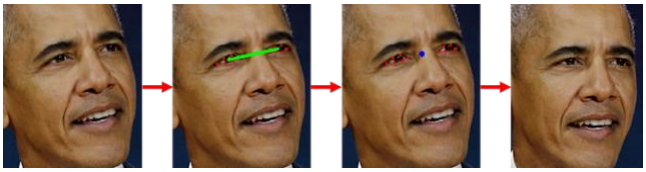


Figure 2. The process of rotating a face such that the eyes lie on a horizontal line.

the image.

3.3. Face Feature Extractor

The VGGFace2 [5] dataset contains 3.31 million images from 9131 celebrities (8631 for training, 500 for evaluation) with an average of 362 images for each subject. This dataset shows large variations in terms of pose, angle, age, and illumination. It also provides a ResNet-based [16] model pre-trained on MS-Celeb-1M and fine-tuned on VGGFace2 dataset. We remove the last layer as the face descriptor and also compare the performance of ResNet-50 to VGG16 [39] in section 4.

The training scheme of VGGFace2 removes the fully connected layer from a pre-trained model trained on MS-Celeb-1M, and then is fine-tuned on the VGGFace2 dataset as an 8631-classes classifier. Finally, we removed the classifier of the model as the face feature extractor.

3.4. Re-ranking via ANNS

3.4.1. Hierarchical Navigable Small World (HNSW)

HNSW [28] can be seen as a coarse-to-fine hierarchical NSW, where the ground layer has all data points and the higher layer contains fewer points. Similar to NSW, HNSW is created by inserting new data points one-by-one. For each insertion, an integer maximum layer l is randomly selected and there are two phases to the insertion process. The first step starts from the top layer to $l+1$ by greedily traversing the graph to find the closest neighbor to the inserted data point in the layer, which is used as the enter-point to continue the search in the next layer. The second step adds the new data point to all layers from layer l to the ground layer. M nearest neighbors are found and are connected with the new point. The search starts from the upper layer which has longer links and greedily traverse the upper layer until a local minimum is reached. After that, the search switches to the lower layer (which has shorter links), restarting the traversal to the local minimum.

3.4.2. Re-ranking policy

In this section, we introduce our re-ranking approach in two different configurations, **Without average** and **With average**.

Without average: Define our feature extractor $f(x)$, which returns face feature. Given a probe person p and compared with whole gallery $\mathcal{G}$ which contains M images in N identities $\mathcal{G} = \{ g_i^j | \, i = 1,2, \dots N;\ j = 1,2 \dots n_i \}$, where each identity has n_i images in the gallery, the distance between p and g_i^j is measured by Cosine distance,

$$d(p, g_i^j) = 1 - \frac{\sum_1^n V_p V_{g_i^j}}{\sqrt{\sum_1^n V_p^2} \sqrt{\sum_1^n V_{g_i^j}^2}} \quad (1)$$

where $V_p = f(p)$ and $V_{g_i^j} = f(g_i^j)$ respectively represent the face vector after of probe person p and gallery image g_i^j. The initial ranking list $L(p, G) = \{g_1^0, g_2^0, \dots g_M^0\}$ can be obtained by the original Cosine distance between p and g_i^j, where $d(p, g_i^0) < d(p, g_{i+1}^0)$. Our goal is to re-rank the original $L(p, G)$, so that the more positive samples move to the top of the list. We define $N(p, k)$ as the k-nearest neighbors of probe p, $N(p, k) = \{g_1^0, g_2^0, \dots g_k^0\}$, which can be obtained by HNSW algorithm (*i.e.*, the top-k elements of the ranking list). After that, we define p' as the mirror image of p, while $N(p', k)$ is the top-k elements of p'. Their union can be defined as $U(p, k) = N(p, k) \cup N(p', k)$, and then we average the distance of each identity appearing in $U(p, k)$. By this operation, the ranking list can be renewed more comprehensively because it can obtain more information from the mirror face.

With average: In this configuration, given a probe image p, we update our original gallery $\mathcal{G}$ to $\bar{\mathcal{G}}$ by averaging every feature of the images belonging to the identity in the dataset (*i.e.* $\bar{\mathcal{G}} = \{ \bar{V}_i | \, i = 1,2, \dots N \}$, where $\bar{V}_i = \frac{\sum_{j=1}^{n_i} f(g_i^j)}{n_i}$). The Cosine distance between p and $\bar{V}_i$,

$$d(p, \bar{V}_i) = 1 - \frac{\sum_1^n V_p \bar{V}_i}{\sqrt{\sum_1^n V_p^2} \sqrt{\sum_1^n \bar{V}_i^2}} \quad (2)$$

where V_p represents the face vector of p.

As discussed above, two ranking list $N(p, k)$ and $N(p', k)$ can be obtained by HNSW algorithm. After that, similar to the method we apply in the without average situation, we merge them according to the averaged distance of each identity in $N(p, k)$ or in $N(p', k)$.

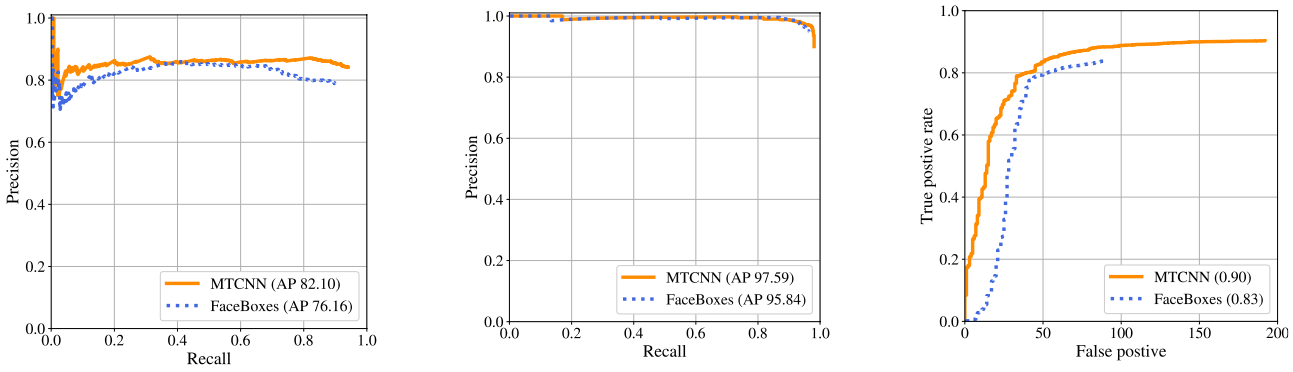
4. Experiments and Results

In section 4.1, we first compare the performance of two different face detectors and their runtime efficiencies. Sections 4.2 and 4.3 respectively describe the face verification and face identification performance.

4.1. Evaluation of Face Detection

4.1.1. Benchmark evaluation

We evaluate the FaceBoxes and MTCNN on three face detection benchmarks, including PASCAL Face [9], Annotated Faces in the Wild (AFW) [33] and Face Detection Data Set and Benchmark (FDDB) [19].



(a) Pascal Face (b) AFW (c) FDDB

Figure 3. Performance evaluation on the different benchmarks

PASCAL Face dataset is collected from the test set of the PASCAL person layout dataset, which consists of 1335 faces in 851 images. Fig. 3(a) shows the precision-recall curves on this dataset, where MTCNN significantly outperforms FaceBoxes.

AFW dataset is built using Flickr images. It has 205 images with 473 faces. As can be seen in Fig. 3(b), no significant performance differences are found between MTCNN and FaceBoxes on this dataset.

FDDB dataset contains 5,171 faces in 2,845 images taken from news articles on Yahoo websites. FDDB provides the bounding ellipses, and thus our data preprocessing converts the ellipses to minimum bounding rectangles. The results shown in Fig. 3(c) indicate that MTCNN outperforms FaceBoxes with a large margin on the discontinuous ROC curve.

4.1.2. Face Detection Runtime efficiency

We measure the speed on the UTKFace [54] dataset. During inference, we filter the boxes by a confidence threshold of 0.5 before applying NMS, and then perform NMS with an IOU of 0.5.

UTKFace is a large-scale face dataset of over 20,000 face images, with only a single face in each image. The detectors only return a bounding-box with the highest confidence score. The result is listed in Tab. 1, where MTCNN gets better performance (95%) on mean average precision (mAP) However, FaceBoxes achieves real-time face detection at 42FPS on the GPU and 20FPS on the CPU with adequate mean average precision.

Approach	mAP(%)	FPS (on GPU)	FPS (on CPU)
MTCNN	**95.1**	8.86	6.25
FaceBoxes	89.3	**42.91**	**20.07**

Table 1. Comparison of FPS and mAP on different approaches.

4.1.3. Face Detection Conclusion

In different scenarios, we can choose the corresponding approach to meet the requirements. Considering the high accuracy in the preprocessing stage of the face feature extraction, we use MTCNN to detect as many faces as possible. However, FaceBoxes is fast enough to satisfy many practical applications, thus we use it on the user interface to smoothly detect user faces. Note that, in the following section, if not otherwise specified, we applied MTCNN as the face detection algorithm.

4.2. Evaluation of Face Verification

In section 4.2.1, we evaluate two different backbone feature extractors, VGG16 and ResNet-50 on the LFW [18] dataset and select the better architecture. In section 4.2.2, we simulate a scenario to evaluate face authentication performance. Lastly, we present the performance for different preprocessing and postprocessing, including the integration of user features and face alignment.

4.2.1. Performance on LFW

Labeled Faces in the Wild (LFW) contains 13,233 images with 5,749 identities and is the standard benchmark for automatic face verification. We follow the standard protocol for unrestricted, labeled outside data. LFW provides 10 sets, each with 300 matched pairs and 300 mismatched pairs. Nine sets are used to select the cosine distance threshold. Verification is then performed on the tenth set. The selected optimal thresholds for VGG16 and ResNet-50 are respectively 0.694 and 0.463.

Table 2 shows the accuracy and time consumption. ResNet-50 significantly outperforms VGG16 with a verification accuracy of 99.33%; thus we choose ResNet-50 as our face descriptor backbone.

4.2.2. Experiments on CASIA-FaceV5 and Helen

Backbone	Accuracy (%)	Time (s)
VGG16	95.33	**129**
ResNet-50	**99.33**	139

Table 2. Performance evaluation on LFW dataset

After choosing the face feature extractor, we use the CASIA-FaceV5 dataset as the authorized user dataset and the Helen [23] dataset as the intruder dataset to evaluate performance in distinguishing between authorized users and intruders.

CASIA-FaceV5 contains 500 subjects with 5 color facial images for each identity. We randomly chose one image as the probe image and the others as gallery images. The **Helen** dataset is a high-resolution dataset originally built for facial keypoint localization. We use 2000 Helen images as the intruder dataset.

We define Ground Truth $g(x) = 1$ for authorized user and $g(x) = 0$ for intruder. **False Accept (FA)** means our system $f(x)$ will incorrectly accept an access attempt by an unauthorized user. **False Reject (FR)** means failing to recognize an authorized person and rejecting that person as an intruder. We defined the face verification penalty as:

$$\boldsymbol{Verification\ Penalty}\ (\boldsymbol{VP}) = FA \cdot \alpha + FR \cdot \beta$$

where α and β are respectively the penalty for FA and FR. A good face verification system is more concerned about the penalty of FA; thus we fix $\beta = 1$ and alter the FA penalty α. From Fig. 4, it can be seen that the penalty increases with α, and that Cosine similarity obtains a lower penalty than Euclidean distance for the same α.

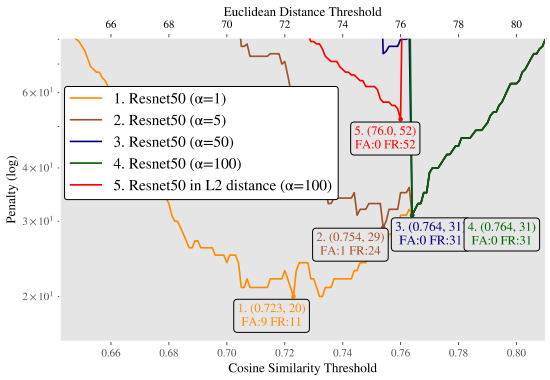


Figure 4. Penalty curves of different false accept penalties.

4.2.3. Alignment and Average

This section presents the results of different preprocessing for face verification. Note that α is fixed at 100 using the same dataset as mentioned in section 4.2.2.

Alignment: According to the component analysis of [32], we performed 2D alignment in the process of building the database and also aligned the test images.

Average: (i) **Without average**: compares every image of the authorized user in the dataset, i.e. returning the identity name which is most similar to the probe image. (ii) **With average**: for each authorized user, we averaged the face features embedded by the ResNet-50.

Fig. 5 shows the average performance is better than comparing all gallery images (orange and blue line). The probable reason is that averaging the representations may obtain more robust and general facial features, thus avoiding transient effects of angle, hairstyle or glasses. Furthermore, performing 2D face alignment gives a boost in performance (solid and dotted line).

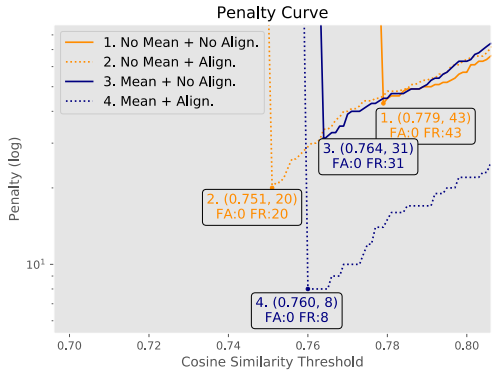


Figure 5. Performance comparison for different configurations.

4.3. Evaluation of Face Identification

This section reports the experimental results for face identification. Section 4.3.1 discusses the performance on a large inter-class variation dataset, CASIA-WebFace [49]. The result for the large intra-class age-invariant dataset, FG-NET, is given in section 4.3.2. Unless otherwise specified, HNSW is used to speed up the 1: N retrieval time and present the Rank-1 accuracy.

4.3.1. Experiments on the CASIA-WebFace Dataset

The CASIA-WebFace consists of 0.5M images of 10K celebrities, collected from the IMDb website. We use the washed CASIA-WebFace, which removes 27,703 wrong images and we select identities represented by five or more images. We apply a 3-fold strategy in which one image is used as the probe image, and the remaining are used as gallery images. As observed in Tab. 3, average and re-ranking both improve accuracy, while alignment is not conducive for this dataset.

4.3.2. Experiments on the FG-NET Dataset

The FG-NET database contains 1002 color or grayscale face images of 82 subjects, ranging in age from 0 to 69. Tab. 4 shows the leave-one-out result for FG-NET, indicating the optimal settings differ from that for CASIA-WebFace, possibly because single image comparison can handle significant intra-personal variation and find the image most closely reflecting the subject's age. Combining all gallery image vectors of a person might lead to representation bias toward the subject's middle-aged years.

We also compared our result with some state-of-the-art approaches [13,14,44,47], finding that, on this dataset, the gap between our proposed method and specialized

Align.	Average	Re-rank	Mean Acc.
×	×	×	0.8439 ± 0.0011
×	×	✓	0.8491 ± 0.0016
×	✓	×	0.8620 ± 0.0005
×	✓	✓	**0.8659 ± 0.0005**
✓	×	×	0.8282 ± 0.0014
✓	×	✓	0.8343 ± 0.0011
✓	✓	×	0.8504 ± 0.0009
✓	✓	✓	0.8601 ± 0.0005

Table 3.
Comparison of different configurations on CASIA-WebFace.

Align.	Average	Re-rank	Acc. (%)
×	×	×	86.62
×	×	✓	86.82
×	✓	×	83.43
×	✓	✓	83.63
✓	×	×	87.32
✓	×	✓	**87.72**
✓	✓	×	84.13
✓	✓	✓	85.03

Table 4.
Comparison of different configurations on FG-NET.

Method	**Acc. (%)**
HFA [10]	69.0
MEFA [11]	76.2
CAN [45]	86.5
LF-CNNs [42]	88.1
Ours	**87.72**

Table 5.
Performance of different methods on FG-NET.

age-invariant face recognition methods is not significant. The comparative results are reported in Tab. 5.

5. Conclusion

A ResNet-based face recognition system is devised using a postprocessing method, comparing results for two different face detectors to ensure the most suitable method is used for different requirements. Moreover, a specific preprocessing is provided to obtain the best performance on identity authentication. Experimental results show our proposed method performs comparably to the state-of-the-art. Re-ranking postprocessing has a positive effect on performance. Future work will focus on improving the re-ranking policy and developing more efficient models on edge devices.

References

[1] Andoni, Alexandr and Piotr Indyk. "Near-Optimal Hashing Algorithms for Approximate Nearest Neighbor in High Dimensions." *Communications of the ACM*, vol. 51, no. 1, 2008, p. 117.

[2] Andoni, Alexandr and Ilya Razenshteyn. "Optimal Data-Dependent Hashing for Approximate near Neighbors." *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, ACM, 2015, pp. 793-801.

[3] ANNOY. https://github.com/spotify/annoy.

[4] Asthana, Akshay et al. "Robust Discriminative Response Map Fitting with Constrained Local Models." *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2013, pp. 3444-3451.

[5] Cao, Qiong et al. "Vggface2: A Dataset for Recognising Faces across Pose and Age." *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*, IEEE, 2018, pp. 67-74.

[6] CASIA-FaceV5. http://biometrics.idealtest.org/.

[7] Ciaccia, Paolo et al. "M-Tree: An E Cient Access Method for Similarity Search in Metric Spaces." *Proceedings of the 23rd VLDB conference, Athens, Greece*, Citeseer, 1997, pp. 426-435.

[8] Deng, Jiankang et al. "Arcface: Additive Angular Margin Loss for Deep Face Recognition." *arXiv preprint arXiv:1801.07698*, 2018.

[9] Everingham, Mark et al. "The Pascal Visual Object Classes (Voc) Challenge." *International journal of computer vision*, vol. 88, no. 2, 2010, pp. 303-338.

[10] Farfade, Sachin Sudhakar et al. "Multi-View Face Detection Using Deep Convolutional Neural Networks." *Proceedings of the 5th ACM on International Conference on Multimedia Retrieval*, ACM, 2015, pp. 643-650.

[11] FG-NET. www.fgnet.rsunit.com/.

[12] Girshick, Ross. "Fast R-Cnn." *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440-1448.

[13] Gong, Dihong et al. "Hidden Factor Analysis for Age Invariant Face Recognition." *Proceedings of the IEEE international conference on computer vision*, 2013, pp. 2872-2879.

[14] Gong, Dihong et al. "A Maximum Entropy Feature Descriptor for Age Invariant Face Recognition." *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 5289-5297.

[15] Guo, Yandong et al. "Ms-Celeb-1m: A Dataset and Benchmark for Large-Scale Face Recognition." *European Conference on Computer Vision*, Springer, 2016, pp. 87-102.

[16] He, Kaiming et al. "Deep Residual Learning for Image Recognition." *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770-778.

[17] Huang, Gary B et al. "Learning Hierarchical Representations for Face Verification with Convolutional Deep Belief Networks." *2012 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, 2012, pp. 2518-2525.

[18] Huang, Gary B et al. "Labeled Faces in the Wild: A Database Forstudying Face Recognition in Unconstrained Environments." *Workshop on faces in'Real-Life'Images: detection, alignment, and recognition*, 2008.

[19] Jain, Vidit and Erik Learned-Miller. "Fddb: A Benchmark for Face Detection in Unconstrained Settings." UMass Amherst Technical Report, 2010.

[20] Kazemi, Vahid and Josephine Sullivan. "One Millisecond Face Alignment with an Ensemble of Regression Trees." *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 1867-1874.

[21] King, Davis E. "Dlib-Ml: A Machine Learning Toolkit." *Journal of Machine Learning Research*, vol. 10, no. Jul, 2009, pp. 1755-1758.

[22] Krizhevsky, Alex et al. "Imagenet Classification with Deep Convolutional Neural Networks." *Advances in neural information processing systems*, 2012, pp. 1097-1105.

[23] Le, Vuong et al. "Interactive Facial Feature Localization." *European conference on computer vision*, Springer, 2012, pp. 679-692.

[24] Li, Haoxiang et al. "A Convolutional Neural Network Cascade for Face Detection." *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 5325-5334.

[25] Li, Yunzhu et al. "Face Detection with End-to-End Integration of a Convnet and a 3d Model." *European Conference on Computer Vision*, Springer, 2016, pp. 420-436.

[26] Liu, Weiyang et al. "Sphereface: Deep Hypersphere Embedding for Face Recognition." *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 212-220.

[27] Liu, Weiyang et al. "Large-Margin Softmax Loss for Convolutional Neural Networks." *ICML*, vol. 2, 2016, p. 7.

[28] Malkov, Yury A and Dmitry A Yashunin. "Efficient and Robust Approximate Nearest Neighbor Search Using Hierarchical Navigable Small World Graphs." *IEEE transactions on pattern analysis and machine intelligence*, 2018.

[29] Malkov, Yury et al. "Approximate Nearest Neighbor Algorithm Based on Navigable Small World Graphs." *Information Systems*, vol. 45, 2014, pp. 61-68.

[30] Matthews, Iain and Simon Baker. "Active Appearance Models Revisited." *International journal of computer vision*, vol. 60, no. 2, 2004, pp. 135-164.

[31] Muja, Marius and David G Lowe. "Scalable Nearest Neighbor Algorithms for High Dimensional Data." *IEEE transactions on pattern analysis and machine intelligence*, vol. 36, no. 11, 2014, pp. 2227-2240.

[32] Parkhi, Omkar M et al. "Deep Face Recognition." *bmvc*, vol. 1, 2015, p. 6.

[33] Ramanan, Deva and Xiangxin Zhu. "Face Detection, Pose Estimation, and Landmark Localization in the Wild." *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Citeseer, 2012, pp. 2879-2886.

[34] Ranjan, Rajeev et al. "A Deep Pyramid Deformable Part Model for Face Detection." *2015 IEEE 7th international conference on biometrics theory, applications and systems (BTAS)*, IEEE, 2015, pp. 1-8.

[35] ---. "Hyperface: A Deep Multi-Task Learning Framework for Face Detection, Landmark Localization, Pose Estimation, and Gender Recognition." *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 1, 2019, pp. 121-135.

[36] Ranjan, Rajeev et al. "An All-in-One Convolutional Neural Network for Face Analysis." *2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017)*, IEEE, 2017, pp. 17-24.

[37] Schroff, Florian et al. "Facenet: A Unified Embedding for Face Recognition and Clustering." *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 815-823.

[38] Shang, Wenling et al. "Understanding and Improving Convolutional Neural Networks Via Concatenated Rectified Linear Units." *international conference on machine learning*, 2016, pp. 2217-2225.

[39] Simonyan, Karen and Andrew Zisserman. "Very Deep Convolutional Networks for Large-Scale Image Recognition." *arXiv preprint arXiv:1409.1556*, 2014.

[40] Sun, Yi et al. "Deepid3: Face Recognition with Very Deep Neural Networks." *arXiv preprint arXiv:1502.00873*, 2015.

[41] Taigman, Yaniv et al. "Deepface: Closing the Gap to Human-Level Performance in Face Verification." *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 1701-1708.

[42] Uijlings, Jasper RR et al. "Selective Search for Object Recognition." *International journal of computer vision*, vol. 104, no. 2, 2013, pp. 154-171.

[43] Wang, Feng et al. "Additive Margin Softmax for Face Verification." *IEEE Signal Processing Letters*, vol. 25, no. 7, 2018, pp. 926-930.

[44] Wen, Yandong et al. "Latent Factor Guided Convolutional Neural Networks for Age-Invariant Face Recognition." *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4893-4901.

[45] Wen, Yandong et al. "A Discriminative Feature Learning Approach for Deep Face Recognition." *European conference on computer vision*, Springer, 2016, pp. 499-515.

[46] Wolf, Lior et al. *Face Recognition in Unconstrained Videos with Matched Background Similarity*. IEEE, 2011.

[47] Xu, Chenfei et al. "Age Invariant Face Recognition and Retrieval by Coupled Auto-Encoder Networks." *Neurocomputing*, vol. 222, 2017, pp. 62-71.

[48] Yang, Shuo et al. "Wider Face: A Face Detection Benchmark." *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 5525-5533.

[49] Yi, Dong et al. "Learning Face Representation from Scratch." *arXiv preprint arXiv:1411.7923*, 2014.

[50] Yianilos, Peter N. "Data Structures and Algorithms for Nearest Neighbor Search in General Metric Spaces." *SODA*, vol. 93, 1993, pp. 311-321.

[51] Zhang, Jie et al. "Coarse-to-Fine Auto-Encoder Networks (Cfan) for Real-Time Face Alignment." *European conference on computer vision*, Springer, 2014, pp. 1-16.

[52] Zhang, Kaipeng et al. "Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks." *IEEE Signal Processing Letters*, vol. 23, no. 10, 2016, pp. 1499-1503.

[53] Zhang, Shifeng et al. "Faceboxes: A Cpu Real-Time Face Detector with High Accuracy." *2017 IEEE International Joint Conference on Biometrics (IJCB)*, IEEE, 2017, pp. 1-9.

[54] Zhang, Zhifei et al. "Age Progression/Regression by Conditional Adversarial Autoencoder." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5810-5818.