國立臺灣大學電機資訊學院資訊工程學系

碩士論文

Department of Computer Science and Information Engineering

College of Electrical Engineering and Computer Science

National Taiwan University

Master Thesis

透過重新排序與近似最鄰近搜尋以改進基於殘差網路之人臉特徵擷取器

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

蕭勝興

Sheng-Hsing Hsiao

指導教授：張智星 教授

Advisor: Prof. Jyh-Shing Roger Jang

中華民國 108 年 7 月

July, 2019

Abstract

Deep residual network (ResNet) is one of the state-of-the-art architectures in image classification and object detection, which can serve as a robust feature extractor when the last layer is removed. Once the faces are embedded as feature vectors, tasks such as face recognition, verification and identification can be easily implemented using some distance measurements. This paper proposes a framework for face recognition based on feature extractor from ResNet, together with other steps for improving its performance, 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 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 the accuracy from 84.3% to 86.5% in large inter-class variation datasets

(CASIA-WebFace) but improves the Rank-1 accuracy from 86.6% to 87.7% in large intra-class variation datasets (FG-NET).

Keywords: ResNet, feature extractor, face recognition, face verification, face identification, re-ranking, Approximate Nearest Neighbor Search.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Faces, a basic attribute that can distinguish one person to another. Face recognition is a popular area of research. There are many face recognition applications that have been deployed on many areas, such as face biometrics for payments, public security, authentication on devices, self-driving vehicles, etc.

Face recognition is a generic term that could imply either or both face verification and identification. Verification is the process of affirming that a claimed identity is correct by comparing the offered claims of identity with one or more previously enrolled templates. A synonym for verification is authentication, which checks if the identity is in the database or not. Identification takes place when the system tries to determine the identity of the individual. The information of the face is collected and compared to all the identities in a database.

Both face verification identification and identification have three steps in general, 1) face detection 2) face alignment 3) face feature embedding. Face detection is a fundamental step for many facial applications. The essential properties of a face detector

must be detecting faces with a high degree of variability in scale, pose, illumination and expression. Alignment is the module to localize the facial landmarks of eyes, nose, lips, etc. These landmarks are used to align faces such as rotation or scaling. Third, the face feature extractor encodes the facial information to a high-dimension feature. These features are used to measure the similarity between two faces. A good feature extractor should be robust to distinguish one from the other person.

Since 2012, AlexNet [49] has won the ImageNet competition by using some deep learning techniques and large data as input, Convolutional Neural Networks (CNNs) has become the mainstream for computer vision like image classification [93] and object detection [31]. Deep CNNs (DCNNs) now are highly used because they learn hierarchical levels of features that correspond to different levels of abstraction. Every level includes the different features, showing strong invariance to the face pose, scale, and lighting change. Therefore, the datasets need to keep growing to provide a corresponding large amount of faces in large variations like CASIA-WebFace [123], MegaFace [45], VGGFace [77], MSCeleb-1M [34], and WIDER Face [121] . The major contributions of this paper are as follows:

- A comparison and analysis of two different face detectors, MTCNN [126] and FaceBoxes [127] on major face detection benchmark, such as PASCAL Face [26], Annotated Faces in the Wild (AFW) [81], and FDDB dataset [40].
- The usage of CASIA-FaceV5 [14] dataset as the authorized users while Helen [51] dataset as the intruders and evaluate the performance of face verification in different configurations.
- We present an ANNS-based re-ranking method to improve the performance on large inter-class variation datasets (*i.e.* CASIA-FaceV5 and CASIA-WebFace) and large intra-class variation datasets (*i.e.* FG-NET [28] and CACD [15]).

The rest of the paper is organized as follows. First, we discuss recent researches in face detection (section 2.1), face alignment (section 2.2), face recognition (section 2.3) and approximate nearest neighbors search (section 2.4). Then, we present our face recognition system pipeline and re-ranking policies in section 3, and the experimental results are provided in section 4. Finally, the conclusion of this paper and discussion of future works are presented in section 5.

Chapter 2

Related Work

We firstly introduce the pipeline of the face verification/identification and discuss two different distance metrics. In the face recognition pipeline, it contains face detection, face alignment and face identification/verification. After that, we will discuss some approaches for Approximate Nearest Neighbors Search (ANNS).

2.1 Face Detection

The first step in any face recognition/ verification system is face detection. Face detection algorithm will output all the locations of faces from an input image or a frame, most of the outputs are in the form of bounding boxes. A good face detection method should be robust to large variations in pose, facial expression, rotation, illumination, scale, resolution, gender, age, ethnicity, occlusion, make-up, etc. Face detection methods can be divided into two different subcategories, one is based on 1) handcraft features and the other is 2) CNN-based face detector.

Handcraft-based methods are usually used for previous face detection systems. As the pioneering work, Viola-Jones face detector [105] utilized Haar-Like features and

AdaBoost learning to train cascade inference for face detection, which achieves real-time face detection. Many subsequent works are inspired by this detector, such as Liao *et al.* [57] and Yang *et al.* [118] proposed new local features, Pham *et al.* [78] and Brubaker *et al.* [12] designed new boosting algorithms to reduce the training time and Bourdev *et al.* [11] and Li, *et al.* [54] indicated that new cascade structures can speed up the detection process.

In addition to the cascade structure, some researches [69, 81, 116, 117] achieved remarkable performance in deformable part models for face detection tasks. These works need high computational expense and may usually require expensive annotation in the training stage to achieve better detection performance.

CNN-based methods can be traced back to 1994, Vaillant *et al.* [104] as the pioneer to train CNN in the manner of a sliding window to detect faces. Osadchy *et al.*[75] proposed a multi-tasks CNN system for simultaneous face detection and pose estimation. Recently, CNN achieves remarkable progress in many computer vision tasks and consequently, most of detection methods are replaced by CNN to learn deeper features. CNN-based face detectors are derived from object detection approaches and can be divided into two types: 1) region-based, and 2) sliding window-based.

1) Region-based methods firstly generate a set object-proposals and use CNN to classify each proposal as a face or not. R-CNN [31] obtained region proposals by selective search [103]. Some recent face detectors are inspired by this, like multi-task learning algorithm HyperFace [85] and multi-purpose algorithm All-in-One Face [86]. Faster R-CNN [87], R-FCN [19] employed Region Proposal Network (RPN) to generate region proposals. Besides, ROI-pooling [29] and position-sensitive RoI pooling [19] are applied to extract features from each region. Jiang *et al.* [42] use Faster R-CNN to detect faces and Li *et al.* [55] proposed a multi-task face detector based on Faster R-CNN framework. These improvements allow them to speed up by significantly reducing the number of face proposals and get better quality of face proposals.

2) 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 the separate region proposal step and thus faster than region-based approaches. Ranjan *et al.* [84] created an image pyramid at multiple scales to detect faces and Farfade *et al.* [27] also fine-tuned CNN model for face / non-face classification tasks. Yang *et al.* [120] trained a series CNNs for facial attribute recognition to yield candidate windows of occluded faces. CascadeCNN [53] used a

cascade architecture built on CNNs for multiple resolutions. Qin *et al.* [80] proposed joint training to achieve end-to-end optimization for CNN cascade. MTCNN [126] adopted a cascaded CNNs that predict faces and landmarks location in a coarse-to-fine manner. The Single Shot Detector (SSD) [61] is also a multi-scale sliding window-based object detector, instead of using image pyramid, it utilizes the hierarchal structure of a single deep CNN. This one-stage face detection method has attracted more attention due to its higher inference efficiency and straightforward system deployment, like ScaleFace [122], S3FD [128], and Pyramidbox [101]. FaceBoxes [127] also inspired by the RPN in Faster R-CNN [87] and the multi-scale mechanism in SSD [61].

In addition to the development of improved face detection algorithms, the benchmark of face detection has been collected by large annotation datasets. Pascal person layout dataset, which is a subset from Pascal VOC [26]. It contains 1,335 faces from 851 images with large appearance variations. Annotated Faces in the Wild (AFW) [81], it has 205 images with 473 faces. FDDB [40] consists 5,171 faces in 2,845 images. MALF [119] dataset has a similar scale which has 5,250 images with 11,931 faces. WIDER Face [121] is a much larger dataset, it contains 32,203 images and labels 393,703 faces with a high degree of variability in scale, pose and occlusion. This dataset

has many tiny faces, some of face detectors mentioned before still struggle with recognizing small faces. Hu *et al.* [36] indicate that context is crucial and define templates that make use of massively large receptive fields.

2.2 Face Alignment

Face alignment is the process of transforming a face into some canonical view and usually done by facial keypoints localization [44]. Facial keypoints include points around the eyes, nose, and mouth on a face. Bansal *et al.* [7] indicated it is important for face identification or verification.

There are two types of facial keypoints detection methods, model-based and regression-based. Model-based methods create a representation of shape during training and use model to fit facial landmarks during testing, include the classic Active Appearance Model (AMM)[70, 90] and Constrained Local Models (CLM) [5, 46].

Regression-based methods directly fit the image appearance with the target output. Kazemi *et al*. [44] showed an ensemble of regression tree can be used to estimate the face's landmark positions and achieve real-time performance with high quality. CFAN [125] used coarse-to-fine auto-encoder networks which cascades a few successive stacked auto-encoder networks. Kumar *et al*. [50] proposed an algorithm for extracting

key-point descriptors using CNN and also presented a face alignment algorithm base on these descriptors.

The capability of facial landmark detection often evaluates on the 300 Faces In-the-Wild database (300W) [88] which comprises of four sub-datasets: Annotated Faces in the Wild (AFW) [81], IBUG [88], LFPW [8], and Helen [51].

2.3 Face Identification and Verification

In this section, we will introduce some related works base on deep network architecture for face recognition. According to [52], there are two widely used paradigms: identification and verification.

In identification, the system has to learn the information from a specific set of identities. At the testing period, a new image or group of images is presented, and the target is to decide which of the gallery identities. By contrast, verification is to analyze two faces images and decide whether they are the same person or not.

There are two main parts of a face identification/ verification system: 1) Face representation; and 2) a classifier (for identification) or similarity measure (for verification).

2.3.1 Face representation

Deep networks have shown the ability to learn the difference between two different people. Huang *et al.* [37] proposed deep learning as a source and combine with traditional methods, such as Local Binary Patterns (LBP). By combining these representations, their method achieves comparable accuracy on the LFW [38] dataset.

In 2014, DeepFace [100] achieved the state-of-the-art accuracy on the LFW benchmark. They used a proprietary face dataset consisting of four million faces belonging to more than 4,000 identities. Similarly, in 2015, Google's FaceNet [91] used over 200 million training data of 3 million people. It directly optimized the embedding itself by using triplets of roughly aligned matching / non-matching face patches. In the same year, VGGface [77] designed a procedure to collect a large-scale dataset from the Internet, and Parkhi *et al.* [77] trained the VGGNet [93] on this dataset and then fine-tuned the networks via a triplet loss function similar to FaceNet. This model achieved competitive results on both LFW and YTF [112] datasets.

The DeepID series models [94-96] used an ensemble of the smaller number of hidden neurons than DeepFace or FaceNet. There are four layers for feature extraction and their training data, CelebFaces+, is enlarged from CelebFaces [98] dataset, which contains 202,599 face images of 10,177 celebrities. The proposed features are extracted

from various face regions to form complementary and over-complete representations, which help it achieve almost as good as human performance on LFW dataset.

2.3.2 Discriminative metric learning

Learning a classifier or a similarity metric is the next step after getting 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 [49], the most widely used cross-entropy based softmax-loss for feature learning. But in recent years, people noticed that the softmax loss is not sufficient, the softmax loss often biases to the sample distribution. Several modified studies have been proposed to explore discriminative loss functions for more robust face representation. Before 2017, Euclidean-distance-based loss played an important role. After that, some loss functions like Angular/cosine-margin-based loss are designed to facilitate the training procedure.

Euclidean-distance-based Loss: Euclidean-distance-based loss is a metric learning method [110, 114] that maps images into Euclidean space to cluster the same person face representations while separate the different person face representations. The most intuitive approach is contrastive loss[97], using pairs of images to train a feature

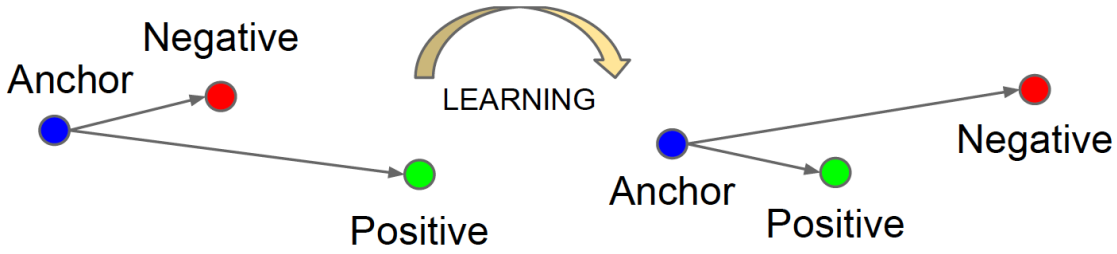


Figure 2.1: Triplet Loss. The Triplet Loss minimizes the distance between an anchor and a positive, both of which have the same identity, and maximizes the distance between the anchor and a negative of a different identity.

embedding where positive pairs are closer and negative pairs are farther apart. But the following problem is the margin parameters of the contrastive loss are often difficult to choose.

The triplet loss, however, tries to enforce a margin between each pair of faces for one identity to all other faces (Fig. 2.1). Along with FaceNet [91] from Google, Parkhi *et al.* [77], Swami *et al.* [89], Liu *et al.* [59], Ding *et al.* [25] are also use triplet loss to embed feature into a discriminative space and got performance improvements on face verification.

No matter contrastive loss or triplet loss will encounter training instability due to the ineffective sampling policy and take a long time to converge. Therefore, Wen *et al.* [113] introduced the Center loss, Center loss learned center for each class and penalized the distances between the deep features and their corresponding class centers. After center

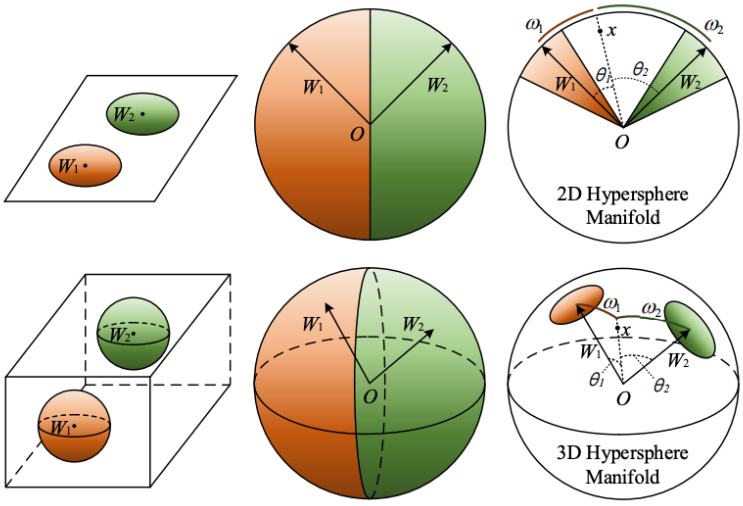


Figure 2.2: Geometry interpretation of A-Softmax loss.

loss was proposed, many of its variations are also presented. Range loss [129] is used to handle long-tailed data and reduce overall intrapersonal variations while enlarging interpersonal differences simultaneously. Wu *et al.* [113] proposed a center invariant loss that penalizes the different between each center of classes. Deng *et al*. [24] proposed marginal loss which simultaneously minimizes the intra-class variances as well as maximizes the inter-class distances by focusing on the marginal samples.

Angular/cosine-margin-based Loss: In 2016, Liu, *et al.*[63] proposed the large-margin softmax (L-Softmax) loss, which is reformulated from original softmax loss. L-softmax makes the classification more rigorous in order to produce a decisive margin. At the next year, SphereFace [62] proposed angular softmax (A-Softmax) loss which further normalized the weight by its L2 norm such that the normalized vector will lie on a

hypersphere, and then the discriminative face features can be learned on a hypersphere manifold with an angular margin (Figure 2.2). To solve the difficulty of optimizing L-Softmax and A-Softmax, ArcFace [23] and CosFace [109] also added the angular margin, $cos(\theta + m)$ while Wang *et al.* [107] added cosine margin, $cos\theta - m$. They are easy to implement and able to converge without the supervision of softmax.

In addition to the two losses mentioned above, there are many other approaches to modify the original softmax loss. Normface [108] explained the necessity of feature and weight normalization. Ranjan *et al.* [83] regularized the softmax loss with a scaled L2-norm constraint (L2-softmax loss) and achieved the STOA on IJB-A [48]. COCO loss [64] optimized the cosine similarity among data and Ring loss [132] presented a simple soft normalization, where it gradually learns to constrain the norm rather than directly enforcing through a hard normalization operation. Crystal loss [82] restricts the features to lie on a hypersphere of a fixed radius and overcomes the limitations of the regular softmax loss.

2.4 Approximate Nearest Neighbor Search

Nowadays, constantly increasing information has led to the growing of time and computation complexity. K-Nearest Neighbor Search (K-NNS) is a common approach

for information retrieval, such as image feature matching in the large dataset [65] and semantic document retrieval [22]. A naïve brute force approach is computing 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. There are several solutions to speed up the retrieval time when the dimension is small, such as Voronoi diagrams [79], kd-trees [9], metric trees [102], ball-trees [74]. However, many real-world applications are facing the high dimensional data and also demanded to achieve sub-linear efficiency, for example in computer vision.

To overcome the "*curse of dimensionality*" [39], Approximate Nearest Neighbors Search (ANNS) was proposed, which instead of finding the exact nearest element but providing a good approximation that is close enough to the query. ANNS can be performed efficiently and sufficiently useful for many practical problems and therefore attracting many studies. There are two ANNS benchmarks, [72] and ANN-benchmarks [6]. In addition, ANNS algorithms can be classified into three categories: **Hashing-based**, **Partition-based** and **Graph-based**.

Hashing-based: The approaches belonging this category project data to low-dimensional representation. Thus, each element could be encoded to hash code. Locality

sensitive hashing (LSH) is a basic hashing-based approach, which refers to a family of functions (known as LSH families) to hash similar data (distance$< r$) to the same bucket with high probability, while dissimilar data points (distance$> cr$) are likely to be in different buckets. It is vital for the LSH-based method to design a good locality sensitive hash function. In Euclidian distance measurement, many hash functions are proposed, such as [21], [2] and [3]. Random linear projections [30, 76] are the most commonly used hash functions to generate hash code, and the parameters are chosen from Gaussian distribution.

If we apply many hash functions, the hash table can be constructed and the collision probability from dissimilar points will decrease. However, it also reduces the collision probability of nearby points, the common solution is creating multiple hash tables but causing time-consuming query time and excessive memory usage. Some methods [66] , [43] , are proposed to search more hash buckets that may contain the nearby neighbors of the query point so that can improve the quality of query and decrease the number of hash tables.

Recently, Product Quantization (PQ) [41] methods become popular for ANNS, which decomposes the original high-dimensional space into the Cartesian product of a

finite number of low-dimensional subspaces that are then quantized separately. There are many approaches base on PQ, one of them is Optimized Product Quantization (OPQ) [29], which uses pre-rotation to further minimize the quantization distortion.

Partition-based: Methods in this category are hierarchical structure based models, namely, the high dimensional space is partitioned into several disjoint regions in a hierarchical manner. Let query q be located in a region r_q, then its nearby neighbors should be in region r_q or near r_q.

According to the way to partition, there are three subcategories, pivoting, hyperplane and compact partitioning schemes. Pivoting approaches partition the vector space relying on the distance from the data point to pivots, such as VP-Tree [124] , ball-tree [60] , M-Tree [18] ,etc. Hyperplane partitioning methods recursively divide the space by the hyperplane with random direction, such as Annoy [4] , Random-Projection Tree [20] , FLANN [71] . Lastly, compact partitioning methods either divide the data points into clusters or create possibly approximate Voronoi partitions to use its locality [73] , [10].

Graph-based: The most common type of graphs used for complex data retrieval using a similarity search is the Proximity Graphs, which define as $G = (V, E)$. A proximity graph is a graph in which each pair of vertices $(u, v) \in V \times V$ is connected

by an edge $e = (u, v)$, $e \in E$, if and only if u and v satisfy the neighbor-relationship. The core idea of graph-based methods is "*a neighbor's neighbor also be a neighbor*", and thus they can explore neighbor's neighbor efficiently by following the edges.

Recently, new types of graphs have been proposed for ANNS. A representative of this type of graph is the Navigable Small World graph (NSW) [67]. The NSW is based on an approximation of the Delaunay graph and 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 is based on iteratively inserting new vertices to the graph. For each new vertex, locate the position and then search its k nearest neighbors in the current graph. The edges connected the new vertex and its k nearest neighbors are defined as *short-range links*. At the i iteration, the *short-range links* of $i - 1$ iteration become *long-range links*. HNSW [68] is an extension of NSW and can be seen as a multi-layer structure consisting of a hierarchical set of proximity graphs for nested subsets of the stored elements. As far as we know, HNSW is one of the state-of-the-art ANNS algorithms so far.

Chapter 3

Methods

In this section, we discuss the pipeline in our system, including face detection, face alignment, face encoding, and re-ranking stage. An overview of the pipeline is displayed in Figure 3.1. First, the introduction of two different face detectors in section 3.1. Then the way how to use 2D face alignment to obtain canonical representations of faces is described in section 3.2. Section 3.3 is a fine description of the ResNet-based face descriptor which pre-trained on MS-Celeb-1M [34] and fine-tuned on VGGFace2 [13] .

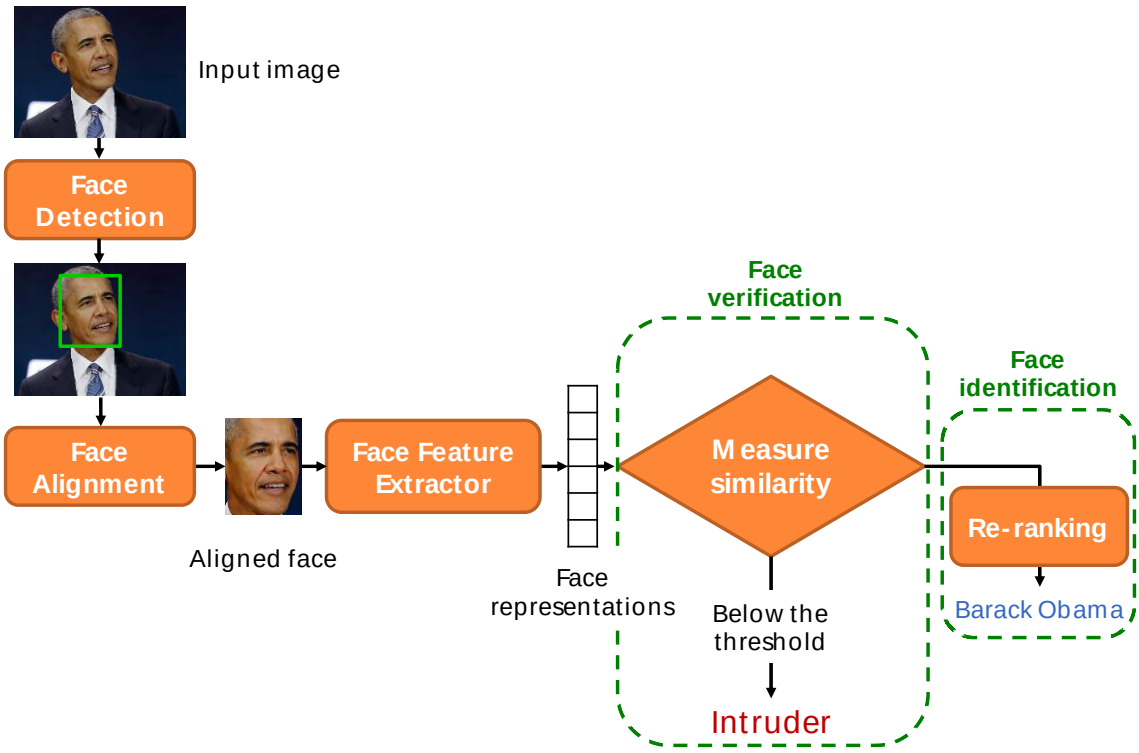


Figure 3.1: System flowchart

Lastly, for the purpose of accelerating and re-ranking the retrieval of face recognition, we apply ANNS and explain our proposed re-ranking method in section 3.4.

3.1 Face Detection

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

3.1.1 MTCNN

Multi-task cascaded CNN (MTCNN) [126] includes three-stage multi-task CNNs trained on WIDER Face [121] . As can be seen in Figure 3.2, MTCNN build a coarse-to-fine detection approach. Before passing an image to the first CNN stage, initially resize it to different scales to build an image pyramid. **Stage 1**: The first stage is the Proposal Network (P-Net), to obtain the candidate windows. After that, use the estimated bounding box regression vectors to calibrate the candidates, then apply non-maximum suppression (NMS) to eliminate highly overlapped windows. **Stage 2**: This is a Refine Network (R-Net) to reject false candidates and also perform bounding box calibration and NMS. **Stage 3**: Output Network (O-Net) is similar to R-Net but aim to locate five facial landmarks' positions. This cascading CNN architecture achieves superior accuracy on the several challenging benchmarks [40] while keeps real-time performance (99fps) on GPU.

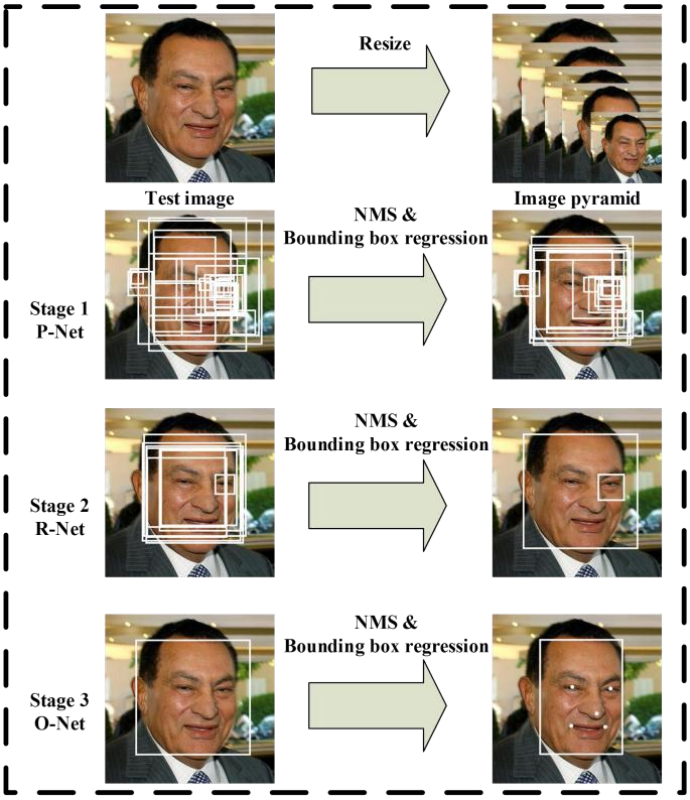
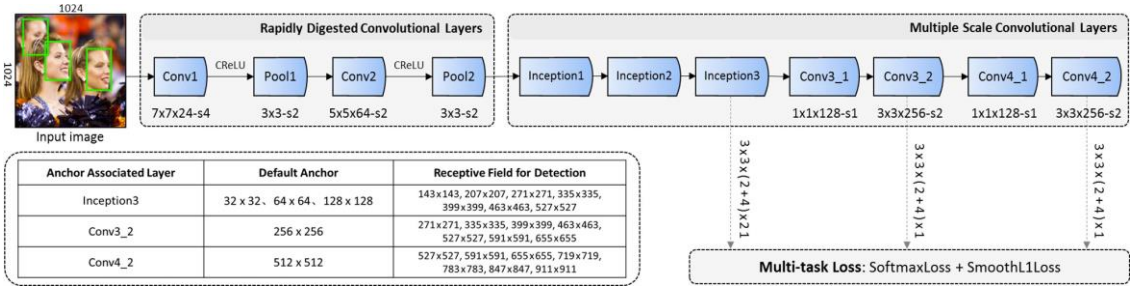


Figure 3.2: The pipeline of MTCNN that includes three-stage multi-task deep convolutional networks. [126]

3.1.2 FaceBoxes

As illustrated in Figure 3.3 , FaceBoxes [127] consists of the Rapidly Digested Convolutional Layers (RDCL), the Multiple Scale Convolutional Layers (MSCL) and the anchor densification strategy. **RDCL** is designed to shrink the input spatial size rapidly by choosing a suitable kernel size. Furthermore, utilize the CReLU [92] activation function can double the number of output channels by simply concatenating negated outputs while significantly increases the speed with a negligible decline in accuracy. **MSCL** is similar to SSD [61] , consisting of several layers, which decrease in size



Anchor Associated Layer	Default Anchor	Receptive Field for Detection
Inception3	32 x 32、64 x 64、128 x 128	143x143, 207x207, 271x271, 335x335, 399x399, 463x463, 527x527
Conv3_2	256 x 256	271x271, 335x335, 399x399, 463x463, 527x527, 591x591, 655x655
Conv4_2	512 x 512	527x527, 591x591, 655x655, 719x719, 783x783, 847x847, 911x911

Figure 3.3: Architecture of the FaceBoxes. [127]

progressively and form the multi-scale feature maps in order that can naturally handle faces of various sizes. Also, the Inception modules [99] are engaged in not only to learn visual patterns for different scales of faces but enrich the receptive fields. **Anchor densification strategy:** the author proposes a new anchor densification strategy to eliminate the imbalance of the density of anchor. This strategy guarantees that different scales of anchor have the same density (i.e., 4) on the image. so that various scales of faces can match almost the same number of anchors. As a consequence, FaceBoxes runs at 20 FPS on a single CPU core and 125 FPS using a GPU for VGA-resolution images.

3.2 Face Alignment

This section we describe the intuitive 2D face alignment procedure of our system. The Ensemble of Regression Trees [44] is used to find 68 facial landmarks, and implementation of the algorithm is available in dlib [47] toolkit. The core of the algorithm is based on the gradient boosting tree for learning an ensemble of regression trees that optimizes the sum of square error loss and naturally handles missing or partially labeled

data. Fig. 4 shows the Landmark estimates at different levels of the cascade initialized with the mean shape centered at the output of a basic Viola & Jones [105] face detector. After the first level of the cascade, the error is already greatly reduced.

After predicting the facial landmarks, we can use these landmarks to rotate the image such that the eyes lie on a horizontal line. As seen in Figure 3.4, firstly, use the eyes region landmarks to compute the center of each eye and compute the angle between two eyes. After that, calculate the midpoint between two eyes, which will serve as the coordinates in which we rotate the face. After rotating around the midpoint between the left and right eyes, we get the canonical view of a face (see Figure 3.5).

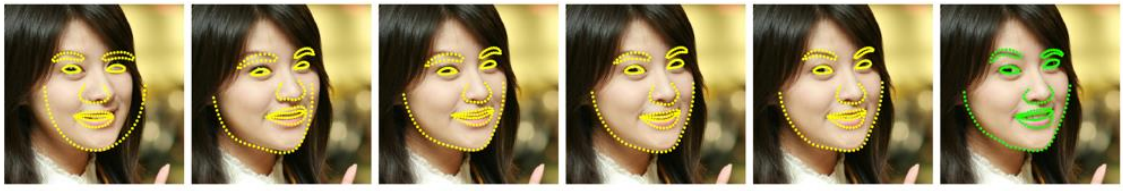


Figure 3.4: Landmark estimates at different levels of the cascade of regressors.

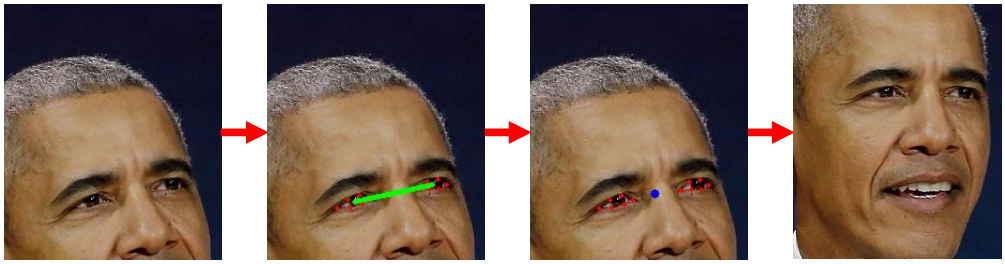


Figure 3.5: The process of rotating a face into some canonical view.

3.3 Face Feature Extractor

In this section, we discuss the detail of the face feature extractor in our system for face verification and identification. We employ VGGFace2 [13] as the face descriptor, which is a ResNet-50 [35] architecture pre-trained on MS-Celeb-1M [34] and then fine-tuned on VGGFace2 dataset.

VGGFace [77] is the previous version, different from VGGFace2, VGGFace initially trains a VGG16 [93] based N-ways classifier on the VGGFace dataset. After that, remove the classifier layer to fine-tune the score vector in Euclidean space using the triplet-loss training scheme. The performance of them is illustrated in section 4.

The VGGFace2 dataset contains 3.31 million images from 9131 celebrities (8631 for training, 500 for evaluation) spanning a wide range of different ethnicities, accents, professions, and ages. The Images were downloaded from Google Image Search and show large variations in pose, age, illumination, and background. Face distribution for different identities is varied, from 87 to 843, with an average of 362 images for each subject. Examples are shown in Figure 3.6.

As discussed above, the training scheme is removing the fully connected layer from a pre-trained model on MS-Celeb-1M which is trained for classification, and then fine-

Figure 3.6: VGGFace2 template examples. **Left**: examples of three different viewpoints (arranged by row) – frontal, three-quarter, profile. **Right**: examples of two subjects for young and mature ages (arranged by row). [13]

tuning on the VGGFace2 dataset as an 8631-classes classifier. Finally, we removed the classifier of the model as the face feature extractor. Note that all the networks are using softmax loss function. During training and testing, the preprocessing of the images is subtracting the mean value of each channel for each pixel.

3.4 Re-ranking via ANNS

Approximate Nearest Neighbors Search (ANNS) is a popular and fundamental method in various applications, such as database, multimedia, and computer vision. As far as we know, Hierarchical Navigable Small World (HNSW) [68] algorithm is one of the most efficient ANNS algorithms; hence we apply HNSW to our system.

3.4.1 Hierarchical Navigable Small World (HNSW)

HNSW can be seen as a coarse-to-fine hierarchical NSW, the ground layer has all data points while the higher layer contains fewer points. Similar to NSW, HNSW is constructed via inserting a new data point, one-by-one. For each inserting, an integer maximum layer l is randomly and there are two phases of the insertion process. The first step starts from the top layer to $l+1$ by greedily traversing the graph in order 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 is adding the new data point to all layers from layer l down to layer ground. M nearest neighbors are found and connected with the new point. The number of enter-points ef also controls the searching quality of HNSW. The search can be seen as a $l=0$ insertion, starting from the upper

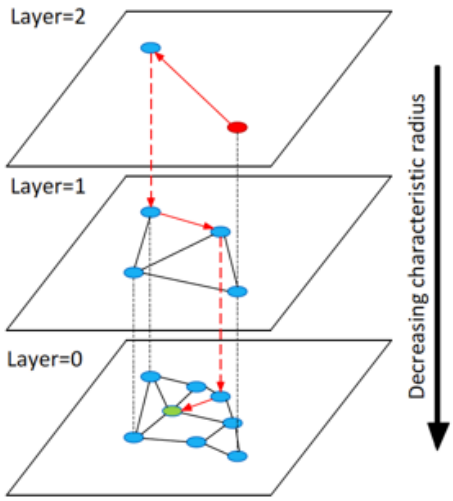


Figure 3.7: Illustration of the Hierarchical NSW idea. The search starts from an element from the top layer (shown red). Red arrows show direction of the greedy algorithm from the entry point to the query (shown green). [68]

layer which only the longest links and greedily traverse the upper layer until a local minimum is reached. After that, the search switch to the lower layer (which has shorter links), restarting the traversal to the local minimum (see Figure 3.7 for illustration).

3.4.2 Re-ranking policy

In this section, we will introduce our re-ranking approach in two different configurations, Fusion-free and Fusion. **Fusion-free** means we compare the probe image to every image in gallery and **Fusion** is averaging the image of each identity in the gallery.

Fusion-free: 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 $\mathcal{G} = \{\ g_i^j \big|\ i = 1,2, \dots N;\ j = 1,2 \dots n_i\}$, where each identity has n_i images in the gallery, the distance between p and g_i^j is measured by Cosine distance,

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

where $V_p = f(p)$ and $V_{g_i^j} = f(g_i^j)$ represent the face vector after of probe person p and gallery image g_i^j, respectively.

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 re-ranking the original $L(p, G)$, so that the more positive samples move top in 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 the p, while $N(p',k)$ is the top-k elements of the p'. The union of them can be defined as $U(p,k) = N(p,k) \cup N(p',k)$, and then we average the distance of each identity who appears in $U(p,k)$. By this operation, renewing the ranking list is more comprehensive because more information could be brought in.

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

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

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 situation of individual, we merge them according average the distance of each identity who is in $N(p,k)$ or in $N(p',k)$.

Figure 3.8: Example of the Re-ranking policy in fusion-free configuration. The search starts from an element from the top layer (shown red). Red arrows show direction of the greedy algorithm from the entry point to the query (shown green). [68]

Chapter 4

Experiments

In section 4.1, we firstly compare the performance of two different face detection algorithms, MTCNN and FaceBoxes, then analyze the runtime efficiency. After analyzing face detection, we evaluate the performance of face verification in section 4.2 and face identification in section 4.3. In the following sections, we describe the evaluation datasets and our experimental protocols. We also describe the changes to the system we made if there are any. All the experiments were conducted on Ubuntu 16.04 LTS operation system using GeForce GTX 1080 Ti and cuDNN v5 with Intel Xeon E5-2630v4 @2.20GHz and 128GB RAM.

4.1 Evaluation of Face Detection

In this section, we firstly evaluate MTCNN and FaceBoxes on the common face detection benchmarks, then compare the runtime efficiency between them.

4.1.1 Evaluate on benchmarks

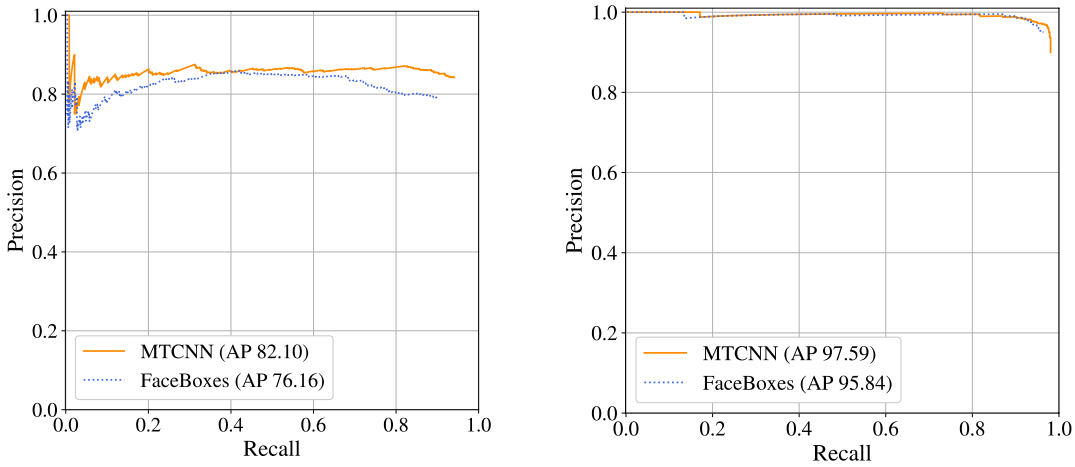
We evaluate the FaceBoxes [127] and MTCNN [126] on three common face detection benchmark datasets, including PASCAL Face [26], Annotated Faces in the Wild (AFW) [81] and Face Detection Data Set and Benchmark (FDDB) [40].

PASCAL Face dataset is collected from the test set of PASCAL person layout dataset, consisting of 1335 faces with large face appearance and pose variations from 851 images. Some examples are shown in Figure 4.1 and Figure 4.3 (a) shows the precision-recall curves on this dataset, MTCNN significantly outperforms FaceBoxes.

Figure 4.1: Some examples of PASCAL Face dataset [26] .

AFW dataset is built using Flickr images. It has 205 images with 473 faces. Some sample images are shown in Figure 4.2. For each face, annotations include a rectangular bounding box, 6 landmarks, and the pose angles. According to the precision-recall curves

Figure 4.2: Some examples of AFW dataset [81] .



(a) PASCAL Face dataset (b) AFW dataset

Figure 4.3: Precision-recall curves on the PASCAL and AFW benchmarks.

in Figure 4.3 (b), no significant differences in performance between MTCNN and FaceBoxes on this dataset.

FDDB dataset contains 5,171 faces in 2,845 images taken from news articles websites. FDDB provides the bounding ellipse, so we converted the ellipses to minimum bounding rectangles. The process is shown in Figure 4.4. The results are shown in Figure 4.4. MTCNN outperforms FaceBoxes a large margin on discontinuous ROC curve.

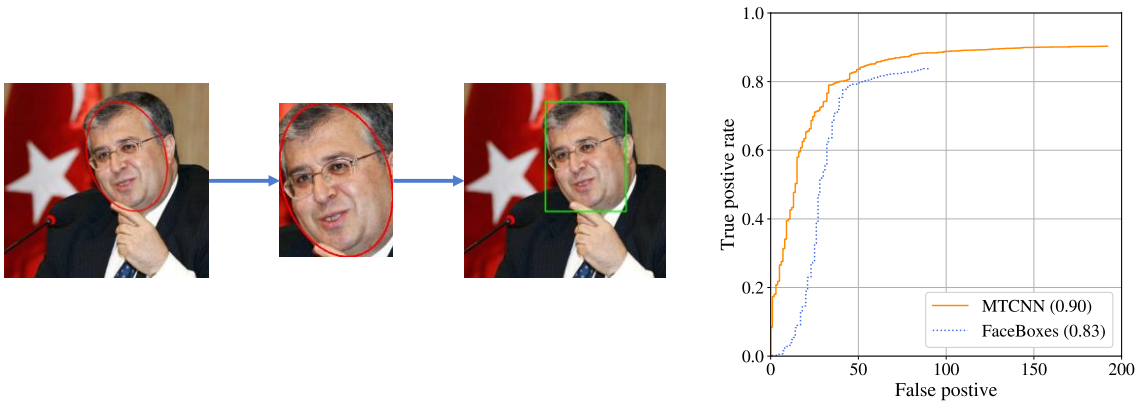


Figure 4.4: Evaluate on FDDB benchmark. **Left**: The processes of converting bounding ellipse to bounding box. **Right**: Discontinuous ROC curves on the FDDB dataset.

4.1.2 Face Detection Runtime Efficiency

We measure the speed on the UTKFace [130] dataset. During inference, we filter the boxes by a confidence threshold of 0.5 before applying NMS, then we perform NMS with IOU of 0.5.

UTKFace is a large-scale face dataset and only a single face in one image. The dataset consists of over 20,000 face images with annotations of age, gender, and ethnicity. The images cover large variation in pose, facial expression, illumination, occlusion, resolution, etc. Some samples are displayed in Figure 4.5.

The result is listed in Table 4.1, MTCNN gets better performance on mean average precision (mAP), which is 95%. However, the FaceBoxes can achieve real-time face detection at 42FPS on the GPU and 20FPS on the CPU with good enough mean average precision.

Figure 4.5: Some examples of UTKFace [130] .

Table 4.1: FPS and mAP compared on different detection approach.

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

4.1.3 Face Detection Conclusion

Face detection in convolutional neural networks (CNN) based methods have achieved remarkable progress but always been accused of its runtime efficiency. In different scenarios, we can choose the corresponding approach to meet the requirements. Since high accuracy is concerned in the preprocessing stage of the face feature extraction, we can apply MTCNN to detect most of the faces as much as possible. In addition, FaceBoxes is fast enough to meet the most of practical applications, we apply it on the user interface to get the face location fluently. Note that, if not mentioned otherwise we applied MTCNN as the face detection algorithm in the following section.

4.2 Evaluation of Face Verification

In section 4.2.1, we evaluate two different backbone feature extractors, VGG16 and ResNet-50 on LFW [38] dataset to select better architecture. Then, we simulate a scenario with the CASIA-FaceV5 [14] dataset as the authorized user dataset while Helen [51]

dataset as the intruder dataset. Lastly, we discuss the performance of different system configurations, including the integration of user features and alignment of the faces.

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 set has 300 matched pairs and 300 mismatched pairs, some samples are shown in Figure 4.6. Nine sets are used to select the cosine similarity threshold. Verification (same or different) is then performed on the tenth set. The selected optimal thresholds for VGG16 and ResNet-50 are respectively 0.694 and 0.463.

Figure 4.7 shows the mean ROC curve on LFW, it can be observed that ResNet-50 architecture is significantly outperformed VGG16. Table 4.2 shows the accuracy and time consumption and it can be seen that ResNet-50 can achieve verification accuracy of 99.33%; thus, we choose ResNet-50 as our face descriptor backbone.

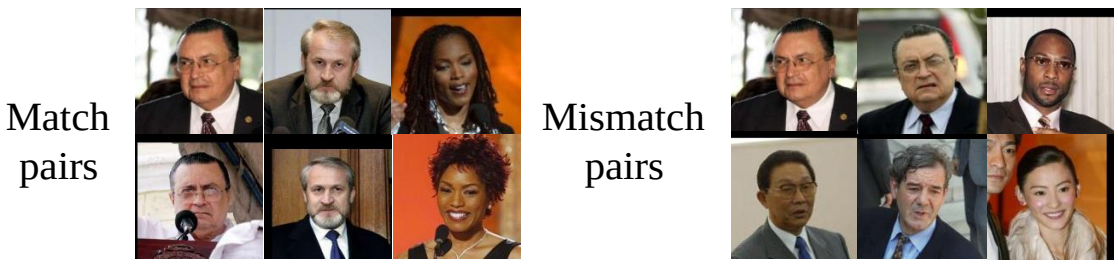


Figure 4.6: Some examples of LFW [38] .

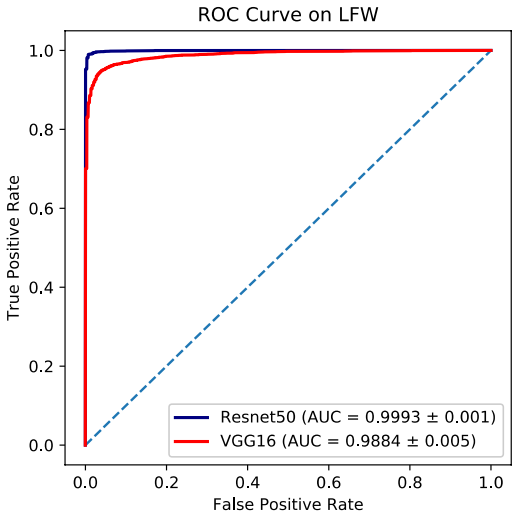


Figure 4.7: ROC curves on LFW

Table 4.2: Performance evaluation on LFW dataset

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

4.2.2 Performance on CASIA-FaceV5 and Helen dataset

In order to evaluate the capability of ResNe-t50 backbone model in discriminating between authorized users and intruders in the practical scenarios, we use CASIA-FaceV5 dataset [14] as the authorized user dataset and Helen [51] dataset as the intruder dataset.

CASIA-FaceV5 (CASIA Face Image Database Version 5.0) contains 2,500 color facial images of 500 subjects. That is, there are 5 images for each identity, we randomly chose one image for test and the others for the gallery images. All face images are 16-bit color BMP files and the resolution is 640*480, Figure 4.8 shows some examples.

Helen dataset is a high-resolution dataset originally built for facial feature localization, and it contains a broad range of appearance variation, including pose,

Figure 4.8: Some examples of CASIA-FaceV5 and Helen dataset. **Left**: CASIA-FaceV5 [14] dataset. **Right**: Helen [51] dataset.

lighting, expression, occlusion, and individual differences. This dataset consists of 2000 training and 330 test images, we use training images as the intruder dataset. If there are multiple faces in the picture, we select the one closest the image center as the test face. Some samples are shown in Figure 4.8.

Inheriting from the confusion matrix, let 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 rejects that person as an intruder. We defined the face verification penalty as:

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

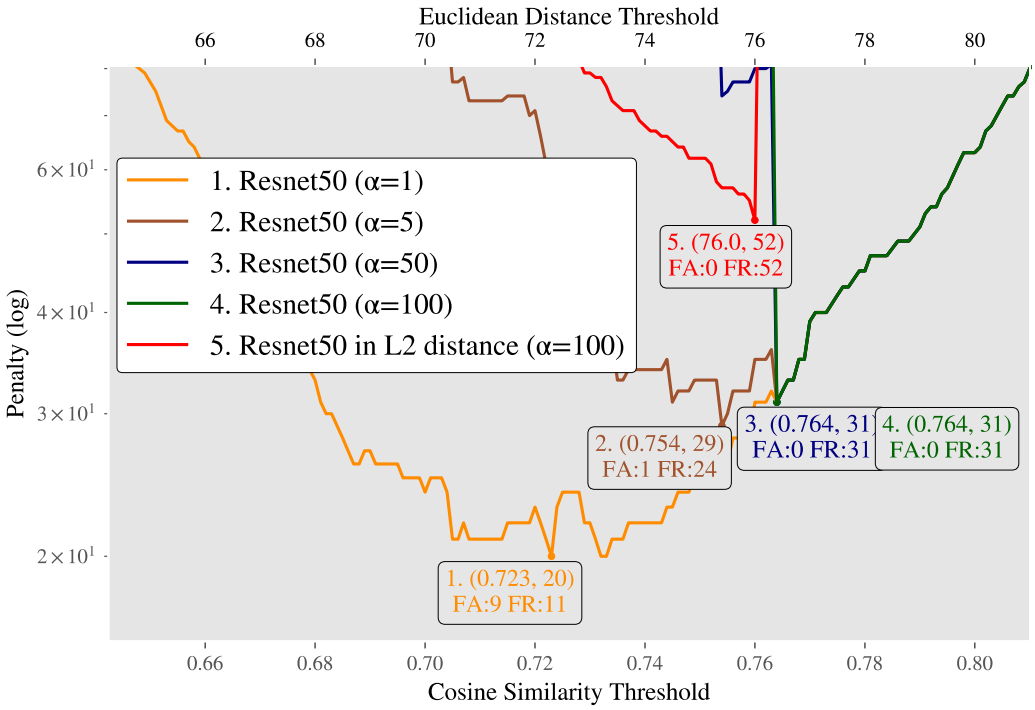


Figure 4.9: Penalty curves of different False Accept penalties and different distance measurements.

α and β are respectively the penalty for FA and FR. A good face verification system is more concerned about the penalty of FA, then we set the FR penalty $\beta = 1$ and alter the FA penalty α. Fig 4.9 shows the minimum FVP in different α, we observed that the penalty increases as the α increases. From Figure 4.9, it can be seen that in the condition of no alignment and average each authorized user's 4 faces in the database, we got the lowest penalty. It also shows the performance of two different distance measurements between Cosine similarity and Euclidean distance, which is Cosine similarity can get the lower penalty in the same α.

4.2.3 Aggregation and Alignment

This section evaluates the effect of the different configurations when building the face verification system using the same authorized user database and intruder dataset mentioned in section 4.2.2. Note that the penalty of FA α is 100.

Aggregation: (i) **Fusion-free**: compared every image of the authorized user in the dataset, i.e. we selected the closest face in the database and return the name. (ii) **Fusion**: for each authorized user, we averaged their face representations which were embedded by the ResNet-50.

Alignment: According to the component analysis of [77], we performed 2D alignment in the process of building the database and also aligned the test images.

Figure 4.10 shows the penalty curve of different configurations, as can be seen, the performance of aggregating each user's face representations is better than comparing

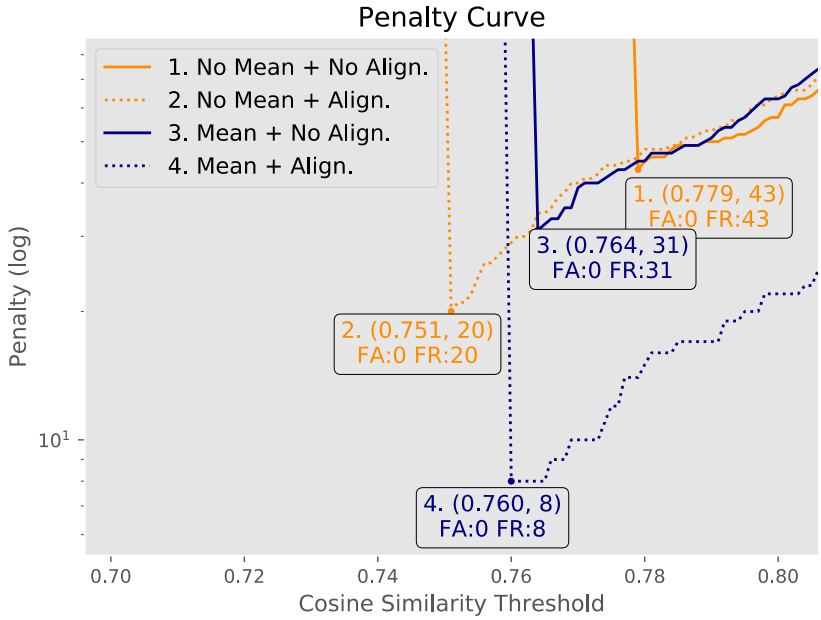


Figure 4.10: Compare the performance of different configurations in face verification.

every image in the database (orange and blue line). The probable reason for this is aggregation the representations may get more robust and general features from a person's face that can avoid the effect of angle, hairstyle or glasses. Furthermore, perform 2D alignment on both training and test images gives a boost in performance (solid and dotted line), comes to the same conclusion as [7]. In conclusion, the best configuration of face verification is the fusion of user face representations and performing alignment on all images.

4.3 Evaluation of Face Identification

This section reports experimental results for face identification on different datasets. Section 4.3.1 and 4.3.2 discuss the performance on large inter-class variation datasets, which are CASIA-FaceV5 [14] and CASIA-WebFace [123] respectively. Section 4.3.2 and section 4.3.3 show the result on two large intra-class age-invariant datasets, FG-NET [28] and Cross-Age Celebrity Dataset (CACD) [15]. If not mentioned otherwise we use HNSW to speed up the 1: N retrieval time and present the Rank-1 accuracy.

4.3.1 Experiments on CASIA-FaceV5

As mentioned previously, we randomly select one image for each identity as the test image and do 5-fold validation. Table 4.3 compares the performance of different

components of face identification. As can be seen, in spite of taking double time, re-ranking the test image does improve the accuracy (rows 1 and 2, rows 5 and 6). In addition, aggregation of user images shows the capability of dealing with the factors of poses, illumination, and hairstyle (rows 1 and 3, rows 5 and 7). In this dataset, alignment does not provide an additional boost on performance.

Table 4.3: Performance of different configurations on CASIA-FaceV5.

No.	Align.	Fusion	Re-rank	Avg. Time (s)	Mean Acc.
1	×	×	×	27	0.9840 ± 0.008
2	×	×	✓	65	0.9880 ± 0.002
3	×	✓	×	22	0.9900 ± 0.006
4	×	✓	✓	48	0.9980 ± 0.002
5	✓	×	×	43	0.9847 ± 0.011
6	✓	×	✓	87	0.9853 ± 0.008
7	✓	✓	×	50	0.9920 ± 0.005
8	✓	✓	✓	92	**0.9987 ± 0.001**

4.3.2 Experiments on CASIA-WebFace

This section discusses the performance on a larger dataset, CASIA-WebFace, which consists of 0.5M images of 10K celebrities, collected from the IMDb website. The authors use a semi-automatically clustering step to construct the dataset, Wang *et al.* [106] analyzed the noise of the dataset, shown in Figure 4.11. We use the washed CASIA-WebFace [1] , which removes 27,703 wrong images and select the identities those images count is more than 5.

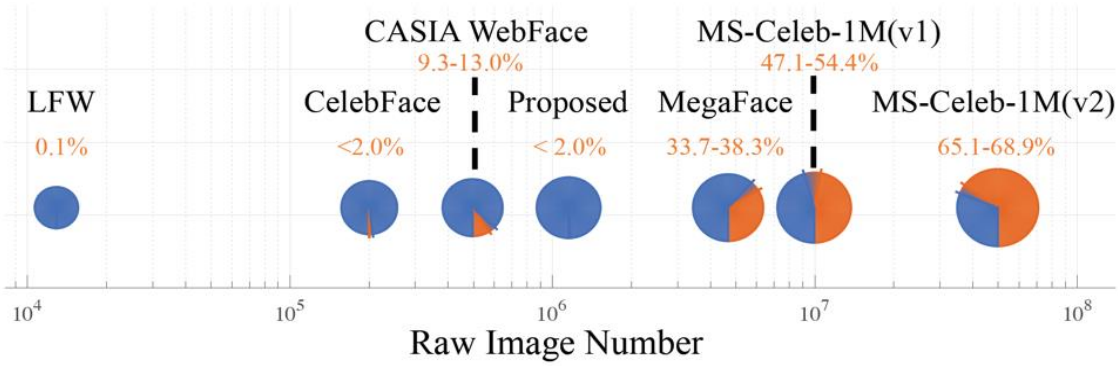


Figure 4.11: A visualization of size and estimated noise percentage of datasets. [106]

We follow the same scheme mentioned before, one image for testing, and the other for gallery images. As observed in Table 4.4, it can be seen that aggregation and re-ranking both significantly improve accuracy. It is also worth noting that alignment is not conducive for this dataset.

Table 4.4: Performance of different configurations on CASIA-WebFace.

No.	Align.	Fusion	Re-rank	Avg. Time (s)	Mean Acc.
1	×	×	×	766	0.8447 ± 0.0014
2	×	×	✓	1479	0.8490 ± 0.0011
3	×	✓	×	749	0.8648 ± 0.0039
4	**×**	**✓**	**✓**	**1445**	**0.8675 ± 0.0023**
5	✓	×	×	1344	0.8297 ± 0.0023
6	✓	×	✓	2636	0.8351 ± 0.0015
7	✓	✓	×	1369	0.8526 ± 0.0030
8	✓	✓	✓	2654	0.8614 ± 0.0018

4.3.3 Experiments on FG-NET

FG-NET [28] database contains 1002 color or grayscale face images of 82 subjects, with ages from 0 to 69. Figure 4.12 shows some examples.

Figure 4.12: Some examples of FG-NET [28] .

Following the testing scheme in [56] , the leave-one-out strategy is used. Table 4.5 shows our result on FG-NET, it reveals that the best configuration on this dataset is contrary to previous, comparing each image and re-ranking can get better performance. The possible explanation is that single image comparison can handle the significant intra-personal variation and find the closest age image of the person. Combine all gallery image vectors of a person might be led to the representation biases toward the middle year of the person. We also compared our result with some state-of-the-art approaches [32, 33, 111, 115] on this dataset, the gap between our proposed method and specialized age-invariant face recognition methods is not significant. The comparative results are reported in Table 4.6.

Table 4.5: Performance of different configurations on FG-NET.

No.	Align.	Fusion	Re-rank	Rank-1 Acc. (%)
1	×	×	×	86.62
2	×	×	✓	86.82
3	×	✓	×	83.43
4	×	✓	✓	83.63
5	✓	×	×	87.32
6	✓	×	✓	**87.72**
7	✓	✓	×	84.13
8	✓	✓	✓	85.03

Table 4.6: Performance of different methods on FG-NET.

Method	Rank-1 Acc. (%)
HFA [32]	69.0
MEFA [33]	76.2
CAN [115]	86.5
LF-CNNs [111]	88.1
Ours	**87.72**

4.3.4 Experiments on CACD

Cross-Age Celebrity Dataset (CACD) [15] is a dataset for age-invariant face recognition, containing 163,446 images from 2,000 celebrities with ages ranging from 16 to 62. The images in this dataset have varied illumination, different poses, different makeup and better simulate the practical scenario.

We followed the experimental setting in [15] , choose 120 celebrities with rank 3-5 as test sets where images taken at 2013 are used as query images. The remaining images

are split into three subsets respectively taken in 2004-2006, 2007-2009 and 2010-2012 as database images.

In our experiment on CACD, we use mean average precision (MAP) as evaluation metrics. Cosine distance is used to compute the similarity of two images. Specifically, let $q_i \in Q$ is the query image and Q is the query dataset. For each q_i, the positive images are expressed as $Y_1, Y_2, Y_3, \dots, Y_m$ and we define E_{ic} as the retrieval results of q_i in descending order from the top to Y_c. Therefore, the average precision (AP) of q_i can be computed as below:

$$AP(q_i) = \frac{1}{m_i} \sum_{c=1}^{m_i} Precision(E_{ic}),$$

where $Precision(E_{ic})$ means the ratio of relevant images in retrieval results E_{ic}. After that, the Mean Average Precision (MAP) of all query images can be denoted as below:

$$MAP(Q) = \frac{1}{|Q|} \sum_{i=1}^{|Q|} AP(q_i).$$

Using this evaluation metric, Figure 4.13 reports the result that we compare our method with some state-of-the-art Cross-Age face recognition algorithms including hidden factor analysis (HFA) [32] , cross-age reference coding (CARC) [16] , generalized similarity model [58] (GSM-2 use more training data), coupled auto-encoder networks (CAN) [115] and age estimation guided convolutional neural network

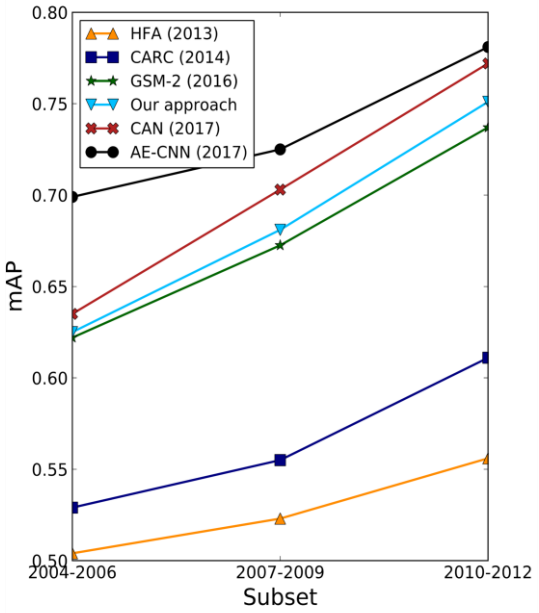


Figure 4.13: The performance of mAP of our approach compared with SOTA algorithms on CACD.

Table 4.7: The Rank-1 accuracy of different methods on three subsets in CACD.

Methods	Rank-1 accuracy (%)		
	04-06	07-09	10-12
LBP [17]	78.0	80.3	85.5
CARC [16]	88.8	88.5	92.2
Ours	82.6	85.1	86.4
Ours + Re-rank	**85.7**	**86.9**	**88.2**

(AE-CNN) [131]. The result shows that our approach still yields good performance compared with age-invariant face recognition algorithms. We also show the rank-1 recognition accuracy of our approach and other methods [16, 17] in Table 4.7, it can be seen that our approach obtains competitive results.

Chapter 5

Conclusions

In this paper, we provided an overview of modern face recognition systems based on DCNNs and discussed each part in the pipeline of face recognition. We also presented the result of two different face detectors so that we can apply the most suitable method in different requirements. Moreover, our system shows superior performance on identity authentication with the setting of fusion and alignment. As for face identification, our re-ranking method not only achieves comparable results on the dataset with large inter-class variation but also on the dataset with large intra-class variation. On the basis of the kind of the dataset, the configurations are intuitive, which applies segregation setting on intra-class variation datasets while use feature fusion strategy leads to more robust representations and better performance on inter-class variation dataset.

Future work will focus on a better understanding of the error cases, further improving the policy of re-ranking, and also developing more efficient architectures inasmuch as the increasing demand for the deployment on edge devices.

Reference

[1] "Washed CASIA-WebFace," https://github.com/cmusatyalab/openface/issues/119.

[2] A. Andoni, and P. Indyk, “Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions,” *Communications of the ACM,* vol. 51, no. 1, pp. 117, 2008.

[3] A. Andoni, and I. Razenshteyn, "Optimal data-dependent hashing for approximate near neighbors." pp. 793-801.

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

[5] A. Asthana, S. Zafeiriou, S. Cheng, and M. Pantic, "Robust discriminative response map fitting with constrained local models." pp. 3444-3451.

[6] M. Aumüller, E. Bernhardsson, and A. Faithfull, “ANN-benchmarks: A benchmarking tool for approximate nearest neighbor algorithms,” *Information Systems*, 2019.

[7] A. Bansal, C. Castillo, R. Ranjan, and R. Chellappa, "The do's and don'ts for cnn-based face verification." pp. 2545-2554.

[8] P. N. Belhumeur, D. W. Jacobs, D. J. Kriegman, and N. Kumar, “Localizing parts of faces using a consensus of exemplars,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 12, pp. 2930-2940, 2013.

[9] J. L. Bentley, “Multidimensional binary search trees used for associative searching,” *Communications of the ACM*, vol. 18, no. 9, pp. 509-517, 1975.

[10] A. Beygelzimer, S. Kakade, and J. Langford, "Cover trees for nearest neighbor." pp. 97-104.

[11] L. Bourdev, and J. Brandt, "Robust object detection via soft cascade." pp. 236-243.

[12] S. C. Brubaker, J. Wu, J. Sun, M. D. Mullin, and J. M. Rehg, “On the design of cascades of boosted ensembles for face detection,” *International Journal of Computer Vision*, vol. 77, no. 1-3, pp. 65-86, 2008.

[13] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and A. Zisserman, "Vggface2: A dataset for recognising faces across pose and age." pp. 67-74.

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

[15] B.-C. Chen, C.-S. Chen, and W. H. Hsu, "Cross-age reference coding for age-invariant face recognition and retrieval." pp. 768-783.

[16] B.-C. Chen, C.-S. Chen, and W. H. Hsu, “Face recognition and retrieval using cross-age reference coding with cross-age celebrity dataset,” *IEEE Transactions on Multimedia,* vol. 17, no. 6, pp. 804-815, 2015.

[17] D. Chen, X. Cao, F. Wen, and J. Sun, "Blessing of dimensionality: High-dimensional feature and its efficient compression for face verification." pp. 3025-3032.

[18] P. Ciaccia, M. Patella, and P. Zezula, "M-tree: An E cient Access Method for Similarity Search in Metric Spaces." pp. 426-435.

[19] J. Dai, Y. Li, K. He, and J. Sun, "R-fcn: Object detection via region-based fully convolutional networks." pp. 379-387.

[20] S. Dasgupta, and Y. Freund, "Random projection trees and low dimensional manifolds." pp. 537-546.

[21] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni, "Locality-sensitive hashing scheme based on p-stable distributions." pp. 253-262.

[22] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, “Indexing by latent semantic analysis,” *Journal of the American society for information science,* vol. 41, no. 6, pp. 391-407, 1990.

[23] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, “Arcface: Additive angular margin loss for deep face recognition,” *arXiv preprint arXiv:1801.07698,* 2018.

[24] J. Deng, Y. Zhou, and S. Zafeiriou, "Marginal loss for deep face recognition." pp. 60-68.

[25] C. Ding, and D. Tao, “Robust face recognition via multimodal deep face representation,” *IEEE Transactions on Multimedia,* vol. 17, no. 11, pp. 2049-2058, 2015.

[26] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, “The pascal visual object classes (voc) challenge,” *International journal of computer vision,* vol. 88, no. 2, pp. 303-338, 2010.

[27] S. S. Farfade, M. J. Saberian, and L.-J. Li, "Multi-view face detection using deep convolutional neural networks." pp. 643-650.

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

[29] T. Ge, K. He, Q. Ke, and J. Sun, "Optimized product quantization for approximate nearest neighbor search." pp. 2946-2953.

[30] A. Gionis, P. Indyk, and R. Motwani, "Similarity search in high dimensions via hashing." pp. 518-529.

[31] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation." pp. 580-587.

[32] D. Gong, Z. Li, D. Lin, J. Liu, and X. Tang, "Hidden factor analysis for age invariant face recognition." pp. 2872-2879.

[33] D. Gong, Z. Li, D. Tao, J. Liu, and X. Li, "A maximum entropy feature descriptor for age invariant face recognition." pp. 5289-5297.

[34] Y. Guo, L. Zhang, Y. Hu, X. He, and J. Gao, "Ms-celeb-1m: A dataset and benchmark for large-scale face recognition." pp. 87-102.

[35] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition." pp. 770-778.

[36] P. Hu, and D. Ramanan, "Finding tiny faces." pp. 951-959.

[37] G. B. Huang, H. Lee, and E. Learned-Miller, "Learning hierarchical representations for face verification with convolutional deep belief networks." pp. 2518-2525.

[38] G. B. Huang, M. Mattar, T. Berg, and E. Learned-Miller, "Labeled faces in the wild: A database forstudying face recognition in unconstrained environments."

[39] P. Indyk, and R. Motwani, "Approximate nearest neighbors: towards removing the curse of dimensionality." pp. 604-613.

[40] V. Jain, and E. Learned-Miller, *Fddb: A benchmark for face detection in unconstrained settings*, UMass Amherst Technical Report, 2010.

[41] H. Jegou, M. Douze, and C. Schmid, "Product quantization for nearest neighbor search," *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, no. 1, pp. 117-128, 2010.

[42] H. Jiang, and E. Learned-Miller, "Face detection with the faster R-CNN." pp. 650-657.

[43] A. Joly, and O. Buisson, "A posteriori multi-probe locality sensitive hashing." pp. 209-218.

[44] V. Kazemi, and J. Sullivan, "One millisecond face alignment with an ensemble of regression trees." pp. 1867-1874.

[45] I. Kemelmacher-Shlizerman, S. M. Seitz, D. Miller, and E. Brossard, "The megaface benchmark: 1 million faces for recognition at scale." pp. 4873-4882.

[46] J. Kim, C. Liu, F. Sha, and K. Grauman, "Deformable spatial pyramid matching for fast dense correspondences." pp. 2307-2314.

[47] D. E. King, "Dlib-ml: A machine learning toolkit," *Journal of Machine Learning Research*, vol. 10, no. Jul, pp. 1755-1758, 2009.

[48] B. F. Klare, B. Klein, E. Taborsky, A. Blanton, J. Cheney, K. Allen, P. Grother, A. Mah, and A. K. Jain, "Pushing the frontiers of unconstrained face detection and recognition: Iarpa janus benchmark a." pp. 1931-1939.

[49] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks." pp. 1097-1105.

[50] A. Kumar, R. Ranjan, V. Patel, and R. Chellappa, “Face alignment by local deep descriptor regression,” *arXiv preprint arXiv:1601.07950*, 2016.

[51] V. Le, J. Brandt, Z. Lin, L. Bourdev, and T. S. Huang, "Interactive facial feature localization." pp. 679-692.

[52] E. Learned-Miller, G. B. Huang, A. RoyChowdhury, H. Li, and G. Hua, "Labeled faces in the wild: A survey," *Advances in face detection and facial image analysis*, pp. 189-248: Springer, 2016.

[53] H. Li, Z. Lin, X. Shen, J. Brandt, and G. Hua, "A convolutional neural network cascade for face detection." pp. 5325-5334.

[54] S. Z. Li, L. Zhu, Z. Zhang, A. Blake, H. Zhang, and H. Shum, "Statistical learning of multi-view face detection." pp. 67-81.

[55] Y. Li, B. Sun, T. Wu, and Y. Wang, "Face detection with end-to-end integration of a convnet and a 3d model." pp. 420-436.

[56] Z. Li, U. Park, and A. K. Jain, “A discriminative model for age invariant face recognition,” *IEEE transactions on information forensics and security,* vol. 6, no. 3, pp. 1028-1037, 2011.

[57] S. Liao, A. K. Jain, and S. Z. Li, “A fast and accurate unconstrained face detector,” *IEEE transactions on pattern analysis and machine intelligence,* vol. 38, no. 2, pp. 211-223, 2015.

[58] L. Lin, G. Wang, W. Zuo, X. Feng, and L. Zhang, “Cross-domain visual matching via generalized similarity measure and feature learning,” *IEEE transactions on pattern analysis and machine intelligence,* vol. 39, no. 6, pp. 1089-1102, 2016.

[59] J. Liu, Y. Deng, T. Bai, Z. Wei, and C. Huang, “Targeting ultimate accuracy: Face recognition via deep embedding,” *arXiv preprint arXiv:1506.07310,* 2015.

[60] T. Liu, A. W. Moore, and A. Gray, “New algorithms for efficient high-dimensional nonparametric classification,” *Journal of Machine Learning Research,* vol. 7, no. Jun, pp. 1135-1158, 2006.

[61] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector." pp. 21-37.

[62] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song, "Sphereface: Deep hypersphere embedding for face recognition." pp. 212-220.

[63] W. Liu, Y. Wen, Z. Yu, and M. Yang, "Large-margin softmax loss for convolutional neural networks." p. 7.

[64] Y. Liu, H. Li, and X. Wang, "Rethinking feature discrimination and polymerization for large-scale recognition," *arXiv preprint arXiv:1710.00870*, 2017.

[65] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91-110, 2004.

[66] Q. Lv, W. Josephson, Z. Wang, M. Charikar, and K. Li, "Multi-probe LSH: efficient indexing for high-dimensional similarity search." pp. 950-961.

[67] Y. Malkov, A. Ponomarenko, A. Logvinov, and V. Krylov, "Approximate nearest neighbor algorithm based on navigable small world graphs," *Information Systems*, vol. 45, pp. 61-68, 2014.

[68] Y. A. Malkov, and D. A. Yashunin, “Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs,” *IEEE transactions on pattern analysis and machine intelligence*, 2018.

[69] M. Mathias, R. Benenson, M. Pedersoli, and L. Van Gool, "Face detection without bells and whistles." pp. 720-735.

[70] I. Matthews, and S. Baker, “Active appearance models revisited,” *International journal of computer vision,* vol. 60, no. 2, pp. 135-164, 2004.

[71] M. Muja, and D. G. Lowe, “Scalable nearest neighbor algorithms for high dimensional data,” *IEEE transactions on pattern analysis and machine intelligence,* vol. 36, no. 11, pp. 2227-2240, 2014.

[72] B. Naidan, L. Boytsov, and E. Nyberg, “Permutation search methods are efficient, yet faster search is possible,” *Proceedings of the VLDB Endowment,* vol. 8, no. 12, pp. 1618-1629, 2015.

[73] G. Navarro, “Searching in metric spaces by spatial approximation,” *The VLDB Journal*, vol. 11, no. 1, pp. 28-46, 2002.

[74] S. M. Omohundro, “Efficient algorithms with neural network behavior,” *Complex Systems*, vol. 1, no. 2, pp. 273-347, 1987.

[75] M. Osadchy, Y. L. Cun, and M. L. Miller, “Synergistic face detection and pose estimation with energy-based models,” *Journal of Machine Learning Research,* vol. 8, no. May, pp. 1197-1215, 2007.

[76] R. Panigrahy, "Entropy based nearest neighbor search in high dimensions." pp. 1186-1195.

[77] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition." p. 6.

[78] M.-T. Pham, and T.-J. Cham, "Fast training and selection of haar features using statistics in boosting-based face detection." pp. 1-7.

[79] F. P. Preparata, and M. I. Shamos, *Computational geometry: an introduction*: Springer Science & Business Media, 2012.

[80] H. Qin, J. Yan, X. Li, and X. Hu, "Joint training of cascaded CNN for face detection." pp. 3456-3465.

[81] D. Ramanan, and X. Zhu, "Face detection, pose estimation, and landmark localization in the wild." pp. 2879-2886.

[82] R. Ranjan, A. Bansal, H. Xu, S. Sankaranarayanan, J.-C. Chen, C. D. Castillo, and R. Chellappa, “Crystal loss and quality pooling for unconstrained face verification and recognition,” *arXiv preprint arXiv:1804.01159*, 2018.

[83] R. Ranjan, C. D. Castillo, and R. Chellappa, “L2-constrained softmax loss for discriminative face verification,” *arXiv preprint arXiv:1703.09507*, 2017.

[84] R. Ranjan, V. M. Patel, and R. Chellappa, "A deep pyramid deformable part model for face detection." pp. 1-8.

[85] R. Ranjan, V. M. Patel, and R. Chellappa, “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, pp. 121-135, 2019.

[86] R. Ranjan, S. Sankaranarayanan, C. D. Castillo, and R. Chellappa, "An all-in-one convolutional neural network for face analysis." pp. 17-24.

[87] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks." pp. 91-99.

[88] C. Sagonas, G. Tzimiropoulos, S. Zafeiriou, and M. Pantic, "300 faces in-the-wild challenge: The first facial landmark localization challenge." pp. 397-403.

[89] S. Sankaranarayanan, A. Alavi, C. D. Castillo, and R. Chellappa, "Triplet probabilistic embedding for face verification and clustering." pp. 1-8.

[90] J. Saragih, and R. Goecke, "A nonlinear discriminative approach to AAM fitting." pp. 1-8.

[91] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering." pp. 815-823.

[92] W. Shang, K. Sohn, D. Almeida, and H. Lee, "Understanding and improving convolutional neural networks via concatenated rectified linear units." pp. 2217-2225.

[93] K. Simonyan, and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.

[94] Y. Sun, Y. Chen, X. Wang, and X. Tang, "Deep learning face representation by joint identification-verification." pp. 1988-1996.

[95] Y. Sun, D. Liang, X. Wang, and X. Tang, “Deepid3: Face recognition with very deep neural networks,” *arXiv preprint arXiv:1502.00873*, 2015.

[96] Y. Sun, X. Wang, and X. Tang, "Deep learning face representation from predicting 10,000 classes." pp. 1891-1898.

[97] Y. Sun, X. Wang, and X. Tang, "Deeply learned face representations are sparse, selective, and robust." pp. 2892-2900.

[98] Y. Sun, X. Wang, and X. Tang, "Hybrid deep learning for face verification." pp. 1489-1496.

[99] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions." pp. 1-9.

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

[101] X. Tang, D. K. Du, Z. He, and J. Liu, "Pyramidbox: A context-assisted single shot face detector." pp. 797-813.

[102] J. K. Uhlmann, “Satisfying general proximity/similarity queries with metric trees,” *Information processing letters*, vol. 40, no. 4, pp. 175-179, 1991.

[103] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders, “Selective search for object recognition,” *International journal of computer vision*, vol. 104, no. 2, pp. 154-171, 2013.

[104] R. Vaillant, C. Monrocq, and Y. Le Cun, “Original approach for the localisation of objects in images,” *IEE Proceedings-Vision, Image and Signal Processing*, vol. 141, no. 4, pp. 245-250, 1994.

[105] P. Viola, and M. J. Jones, “Robust real-time face detection,” *International journal of computer vision*, vol. 57, no. 2, pp. 137-154, 2004.

[106] F. Wang, L. Chen, C. Li, S. Huang, Y. Chen, C. Qian, and C. Change Loy, "The devil of face recognition is in the noise." pp. 765-780.

[107] F. Wang, J. Cheng, W. Liu, and H. Liu, “Additive margin softmax for face verification,” *IEEE Signal Processing Letters*, vol. 25, no. 7, pp. 926-930, 2018.

[108] F. Wang, X. Xiang, J. Cheng, and A. L. Yuille, "Normface: l 2 hypersphere embedding for face verification." pp. 1041-1049.

[109] H. Wang, Y. Wang, Z. Zhou, X. Ji, D. Gong, J. Zhou, Z. Li, and W. Liu, "Cosface: Large margin cosine loss for deep face recognition." pp. 5265-5274.

[110] K. Q. Weinberger, and L. K. Saul, “Distance metric learning for large margin nearest neighbor classification,” *Journal of Machine Learning Research*, vol. 10, no. Feb, pp. 207-244, 2009.

[111] Y. Wen, Z. Li, and Y. Qiao, "Latent factor guided convolutional neural networks for age-invariant face recognition." pp. 4893-4901.

[112] L. Wolf, T. Hassner, and I. Maoz, *Face recognition in unconstrained videos with matched background similarity*: IEEE, 2011.

[113] Y. Wu, H. Liu, J. Li, and Y. Fu, "Deep face recognition with center invariant loss." pp. 408-414.

[114] E. P. Xing, M. I. Jordan, S. J. Russell, and A. Y. Ng, "Distance metric learning with application to clustering with side-information." pp. 521-528.

[115] C. Xu, Q. Liu, and M. Ye, “Age invariant face recognition and retrieval by coupled auto-encoder networks,” *Neurocomputing,* vol. 222, pp. 62-71, 2017.

[116] J. Yan, Z. Lei, L. Wen, and S. Z. Li, "The fastest deformable part model for object detection." pp. 2497-2504.

[117] J. Yan, X. Zhang, Z. Lei, and S. Z. Li, “Face detection by structural models,” *Image and Vision Computing,* vol. 32, no. 10, pp. 790-799, 2014.

[118] B. Yang, J. Yan, Z. Lei, and S. Z. Li, "Aggregate channel features for multi-view face detection." pp. 1-8.

[119] B. Yang, J. Yan, Z. Lei, and S. Z. Li, "Fine-grained evaluation on face detection in the wild." pp. 1-7.

[120] S. Yang, P. Luo, C.-C. Loy, and X. Tang, "From facial parts responses to face detection: A deep learning approach." pp. 3676-3684.

[121] S. Yang, P. Luo, C.-C. Loy, and X. Tang, "Wider face: A face detection benchmark." pp. 5525-5533.

[122] S. Yang, Y. Xiong, C. C. Loy, and X. Tang, “Face detection through scale-friendly deep convolutional networks,” *arXiv preprint arXiv:1706.02863*, 2017.

[123] D. Yi, Z. Lei, S. Liao, and S. Z. Li, “Learning face representation from scratch,” *arXiv preprint arXiv:1411.7923*, 2014.

[124] P. N. Yianilos, "Data structures and algorithms for nearest neighbor search in general metric spaces." pp. 311-21.

[125] J. Zhang, S. Shan, M. Kan, and X. Chen, "Coarse-to-fine auto-encoder networks (cfan) for real-time face alignment." pp. 1-16.

[126] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, “Joint face detection and alignment using multitask cascaded convolutional networks,” *IEEE Signal Processing Letters*, vol. 23, no. 10, pp. 1499-1503, 2016.

[127] S. Zhang, X. Zhu, Z. Lei, H. Shi, X. Wang, and S. Z. Li, "Faceboxes: A CPU real-time face detector with high accuracy." pp. 1-9.

[128] S. Zhang, X. Zhu, Z. Lei, H. Shi, X. Wang, and S. Z. Li, "S3fd: Single shot scale-invariant face detector." pp. 192-201.

[129] X. Zhang, Z. Fang, Y. Wen, Z. Li, and Y. Qiao, "Range loss for deep face recognition with long-tailed training data." pp. 5409-5418.

[130] Z. Zhang, Y. Song, and H. Qi, "Age progression/regression by conditional adversarial autoencoder." pp. 5810-5818.

[131] T. Zheng, W. Deng, and J. Hu, "Age estimation guided convolutional neural network for age-invariant face recognition." pp. 1-9.

[132] Y. Zheng, D. K. Pal, and M. Savvides, "Ring loss: Convex feature normalization for face recognition." pp. 5089-5097.